CS 6316 Machine Learning

Introduction to Learning Theory

Yangfeng Ji

Department of Computer Science
University of Virginia

UNIVERSITY of VIRGINIA | ENGINEERING

Overview

1. A Toy Example
2. A Formal Model
3. Empirical Risk Minimization
4. Finite Hypothesis Classes
5. PAC Learning
6. Agnostic PAC Learning

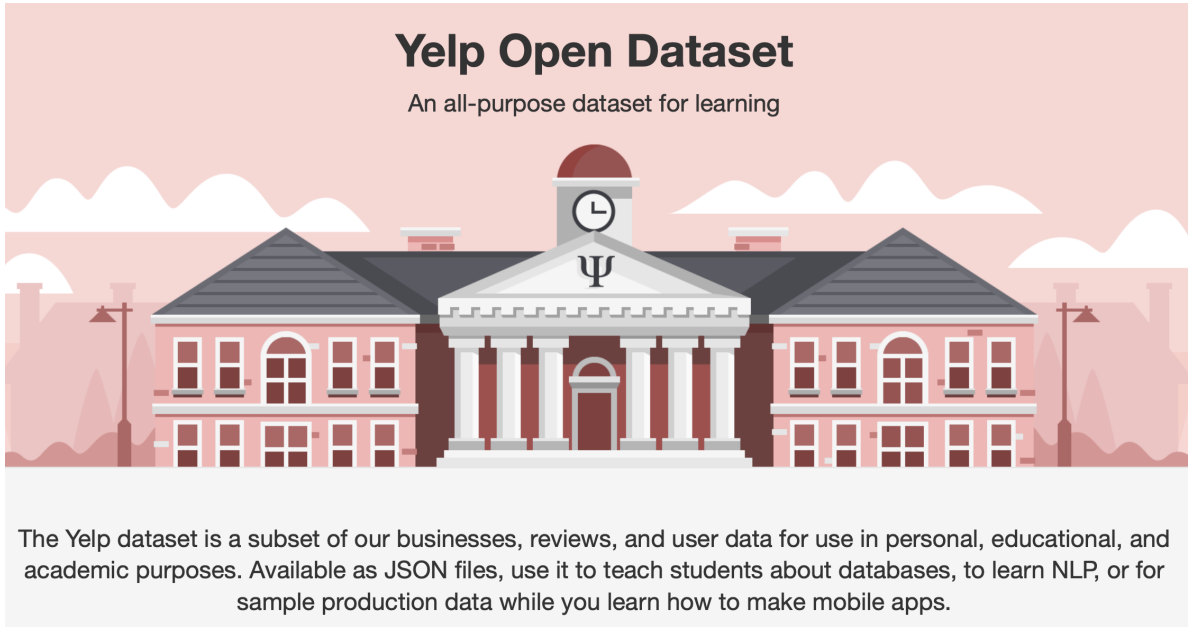
Real-world Classification Problem

Image classification

14M images, 20K categories

Real-world Classification Problem (II)

Sentiment classification

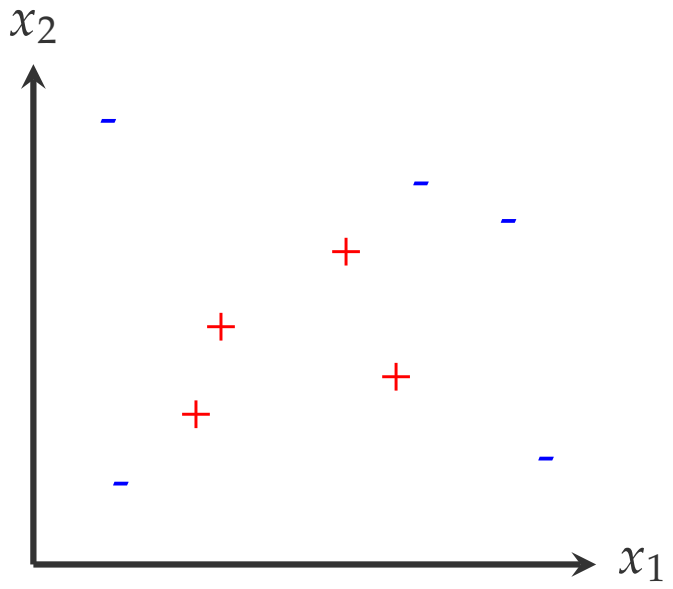


192K businesses, 6.6M user reviews

A Toy Example

Question

Based on the following observations, try to find out the shape/size of the area where the positive examples come from

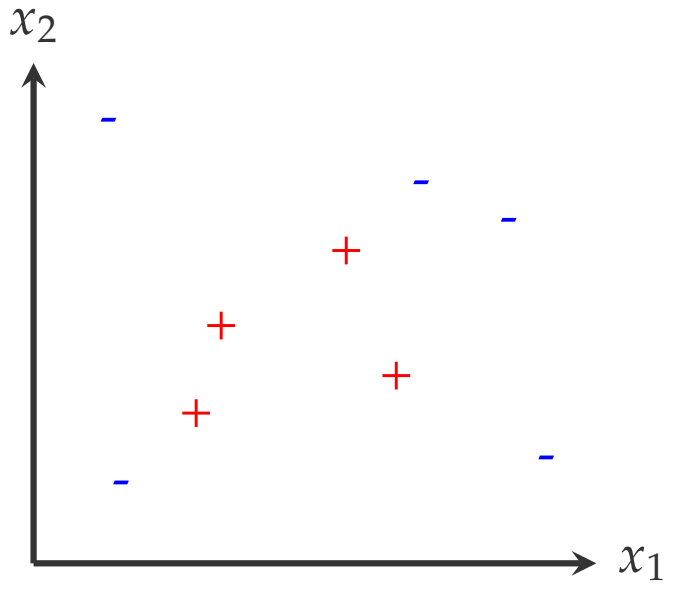


We have to make certain assumptions, otherwise there is no way to answer this question.

Hypotheses

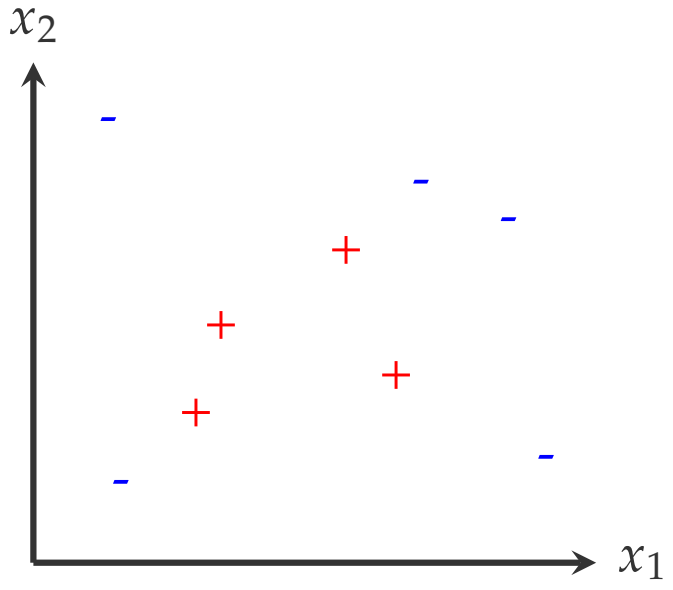
Given these data points, answer the following two questions:

1. Which shape is the underlying distribution of red points?
   - A triangle
   - A rectangle
   - A circle
2. What is the size of that shape?



Basic Concepts (I)

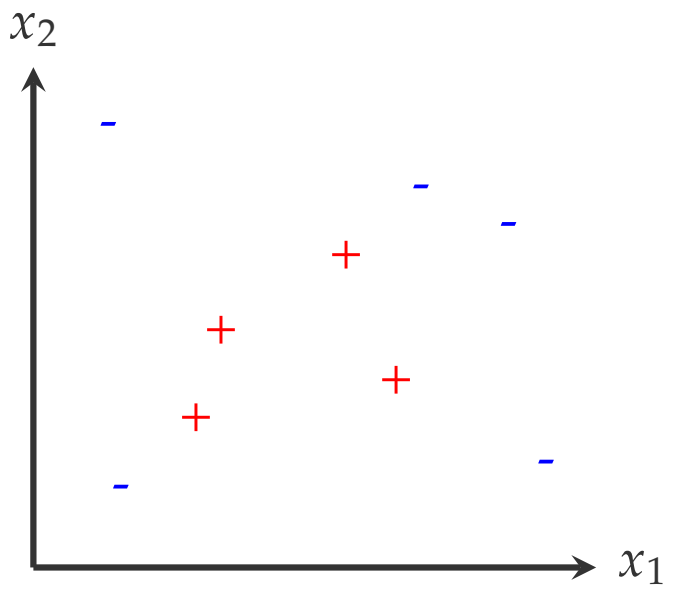
Domain set or **input space** $\mathcal{X}$: the set of all possible examples



- In the example, $\mathcal{X} = \mathbb{R}^2$
- Each point $\boldsymbol{x}$ in $\mathcal{X}$, $\boldsymbol{x} \in \mathcal{X}$, is called one *instance*.

Basic Concepts (II)

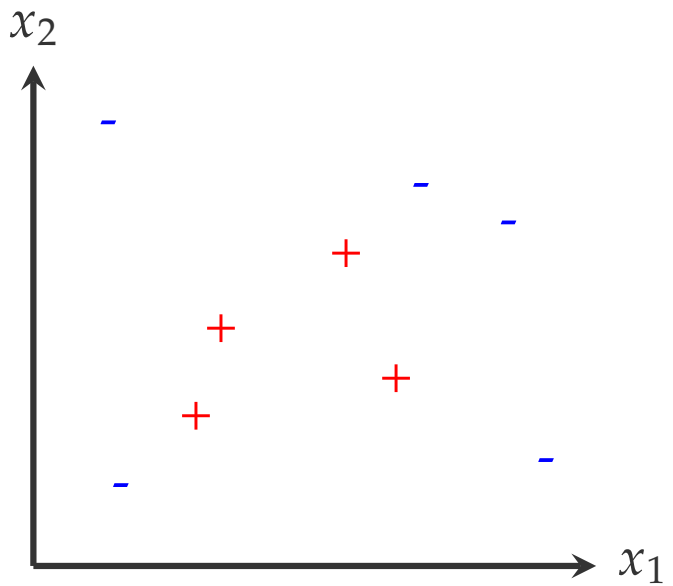
Label set or **output space** $\mathcal{Y}$: the set of all possible labels



- In this toy example, $\mathcal{Y} \in \{+, -\}$
- In this course, we often restrict the label set to be a two-element set, such as $\{+1, -1\}$

Basic Concept (III)

Training set S: a finite sequence of pairs in $\mathcal{X} \times \mathcal{Y}$, represented as $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$ with size m



Basic Concept: Hypothesis Space

- **Hypothesis class** or **hypothesis space** $\mathcal{H}$: a set of functions that map instances to labels
- Each element h in this hypothesis class is called a *hypothesis*

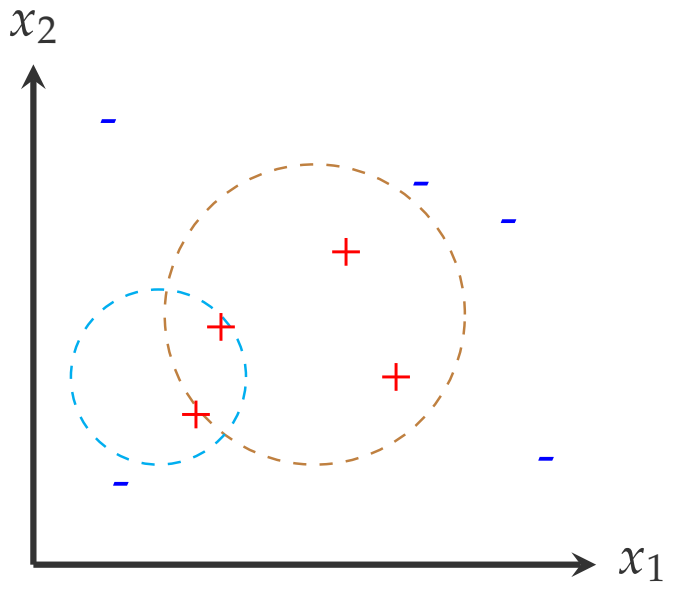


Figure: Two hypotheses from the Circle class.

Basic Concept: Hypothesis Space (Cont.)

If we represent a hypothesis by its parameter value, then each hypothesis corresponds one point in the hypothesis space.

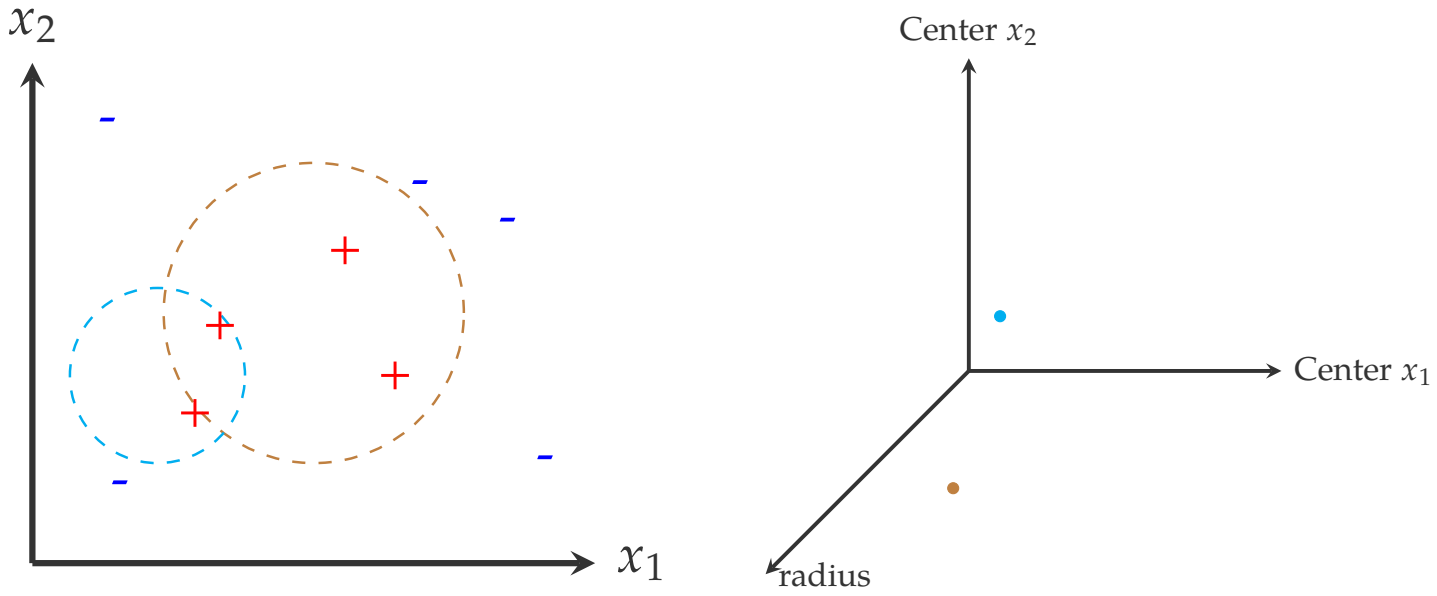
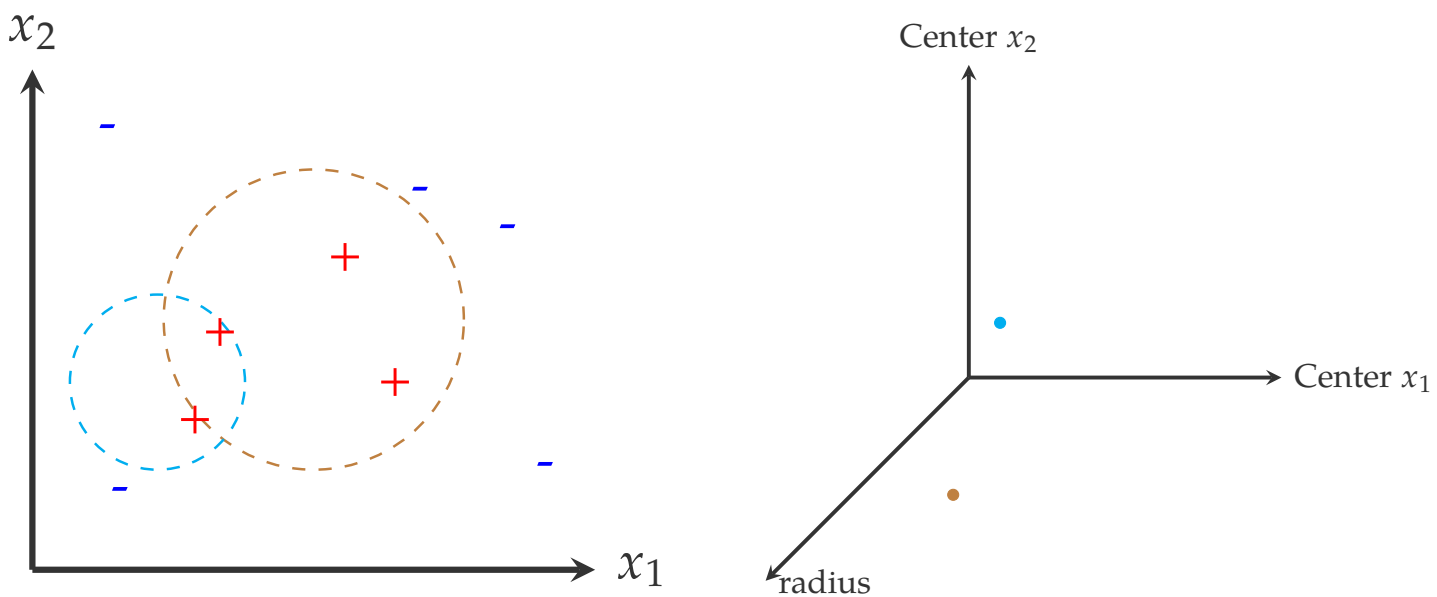


Figure: Visualizing the Circle hypothesis class.

Basic Concept: Machine Learners

- A (machine) learner is an *algorithm* A that can find an *optimal* hypothesis from $\mathcal{H}$ based on the training set S
- This optimal hypothesis is represented as $A(S)$



- A hypothesis space $\mathcal{H}$ is learnable if such an algorithm A exists[1]

[1] A precise definition will be provided later in this lecture.

Why a Toy Problem?

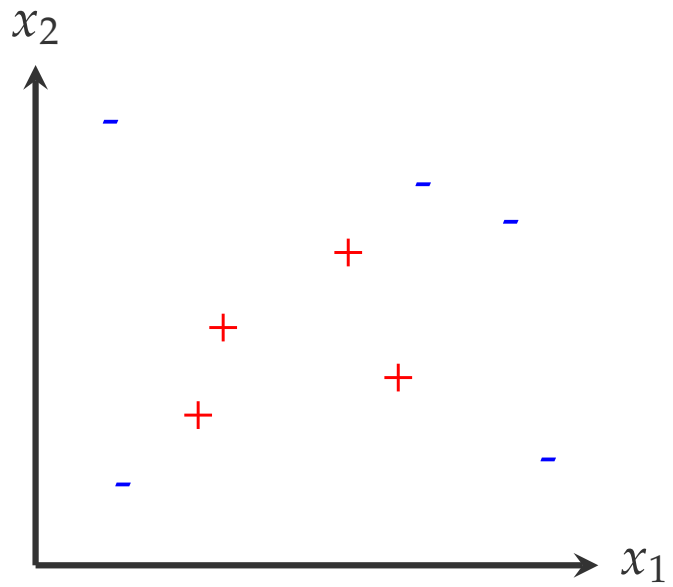
With a toy problem, we can have the following conveniences that we usually do not have with real-world problem,

- Do not need data pre-processing
- Do not need feature engineering
- Make some *unrealistic* assumptions, e.g.,
  - Assume we know the underlying data distribution
  - Assume at least one of the classifiers we pick will completely solve the problem

A Formal Model

Basic Concepts: Summary

- Domain set $\mathcal{X}$
- Label set $\mathcal{Y}$
- Training data S: the observations
- Hypothesis class $\mathcal{H}$: rectangle class
- A learner A: an algorithm that finds an *optimal* hypothesis



Data generation process

An idealized process to illustrate the relations among domain set $\mathcal{X}$, label set $\mathcal{Y}$, and the training set S

1. the probability distribution $\mathcal{D}$ over the domain set $\mathcal{X}$
2. sample an instance $\boldsymbol{x} \in \mathcal{X}$ according to $\mathcal{D}$
3. annotate it using the labeling function f as $y = f(\boldsymbol{x})$

Example

Assume the data distribution $\mathcal{D}$ over the domain set $\mathcal{X}$ is defined as

$$p(x) = \underbrace{\frac{1}{2}\mathcal{N}(x; 2, 1)}_{\text{component 1}} + \underbrace{\frac{1}{2}\mathcal{N}(x; -2, 1)}_{\text{component 2}} \tag{1}$$

The specific data generation process: for each data point

1. Randomly select a Gaussian component
2. Sample x from the corresponding component
3. Label x based on which component was selected at step 1
   - Component 1: positive
   - Component 2: negative

Example (Cont.)

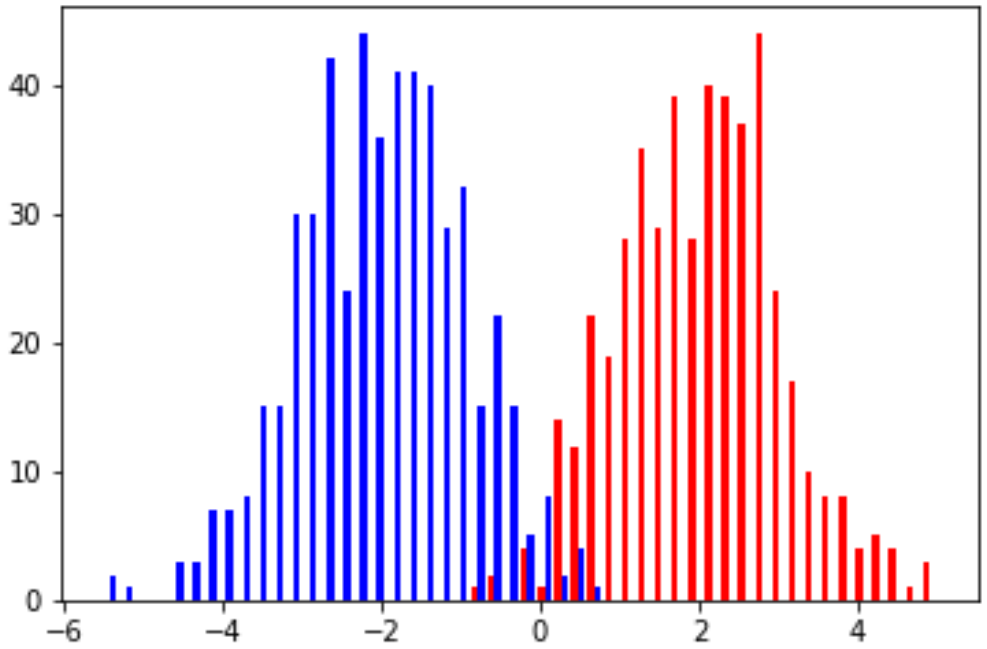


Figure: 1K examples generated with the previous process.

Measures of success

- The error of a classifier as the probability that it does not predict the correct label on a randomly generated instance x
- Definition

$$L_{\mathcal{D},f}(h) = \mathbb{P}_{x \sim \mathcal{D}}[h(x) \neq f(x)] \tag{2}$$

  - $x \sim \mathcal{D}$: an instance generated following the distribution $\mathcal{D}$
  - $h(x) \neq f(x)$: prediction from hypothesis h does not match the labeling function output
  - $L_{\mathcal{D},f}(h)$: the error of h is measured with respect to $\mathcal{D}$ and f

True Error/Risk

Other names (used interchangably):

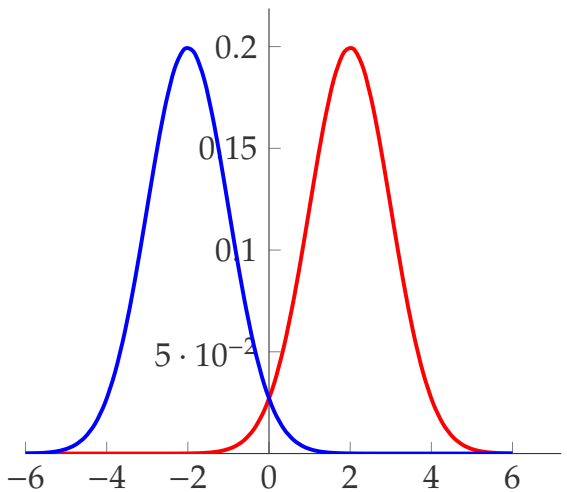
- the *generalization* error
- the true error
- the risk

$$L_{\mathcal{D},f}(h) = \mathbb{P}_{\boldsymbol{x}\sim\mathcal{D}}[h(\boldsymbol{x}) \neq f(\boldsymbol{x})] \tag{3}$$

Example

Assume we have the data distribution $\mathcal{D}$ and the labeling function f as following

$$
\begin{aligned}
p(y = +1) = p(y = -1) &= \frac{1}{2} \\
p(x \mid y = +1) &= \mathcal{N}(x; 2, 1) \\
p(x \mid y = -1) &= \mathcal{N}(x; -2, 1)
\end{aligned}
\tag{4}
$$



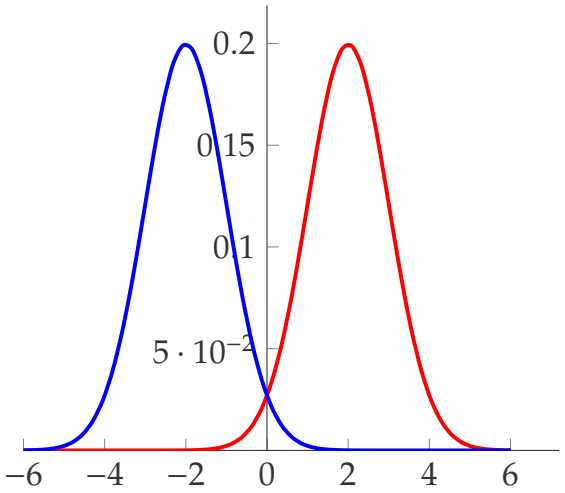
Note that, $p(x)$ is the same as in the example of data

Example (Cont.)

If h is defined as

$$h(x) = \begin{cases} +1 & p(+1 \mid x) \geq p(-1 \mid x) \\ -1 & \text{otherwise} \end{cases} \tag{5}$$

then what is $L_{\mathcal{D},f}(h) = \mathbb{P}_{\boldsymbol{x} \sim \mathcal{D}}[h(\boldsymbol{x}) \neq f(\boldsymbol{x})]$?

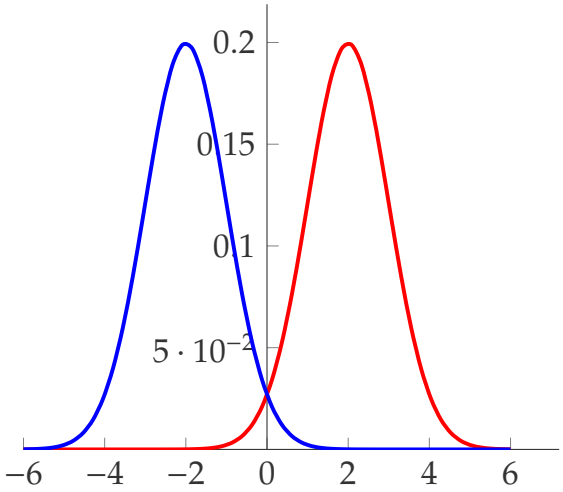


Example (Cont.)

If h is defined as

$$h(x) = \begin{cases} +1 & p(+1 \mid x) \geq p(-1 \mid x) \\ -1 & \text{otherwise} \end{cases} \tag{5}$$

then what is $L_{\mathcal{D},f}(h) = \mathbb{P}_{\boldsymbol{x}\sim\mathcal{D}}[h(\boldsymbol{x}) \neq f(\boldsymbol{x})]$?



The Bayes predictor: the best predictor if we know the data distribution (more detail will be discussed later)

Comments

Recall the definition of true risk with the data distribution $\mathcal{D}$ and the labeling function f

$$L_{\mathcal{D},f}(h) = \mathbb{P}_{x \sim \mathcal{D}}[h(x) \neq f(x)] \tag{6}$$

It impossible to compute $L_{\mathcal{D},f}(h)$ in practice, since we do not know

- the distribution of data generation $\mathcal{D}$
- the labeling function f

Alternative option: Empirical Risk

Empirical Risk Minimization

Empirical Risk

The definition of the **empirical risk** (*or*, empirical error, training error):

$$L_S(h) = \frac{|\{i \in [m] : h(\boldsymbol{x}_i) \neq y_i\}|}{m} \tag{7}$$

Explanations

- $[m] = \{1, 2, \ldots, m\}$ where m is the total number of instances in S
- $\{i \in [m] : h(\boldsymbol{x}_i) \neq y_i\}$: the set of instances that h predicts wrong
- $|\{i \in [m] : h(\boldsymbol{x}_i) \neq y_i\}|$: the size of the set
- $L_S(h)$ defines with respect to the set S

Example

Empirical risk is defined on the training set S:

$$L_S(h) = \frac{|\{i \in [m] : h(\boldsymbol{x}_i) \neq y_i\}|}{m} \tag{8}$$

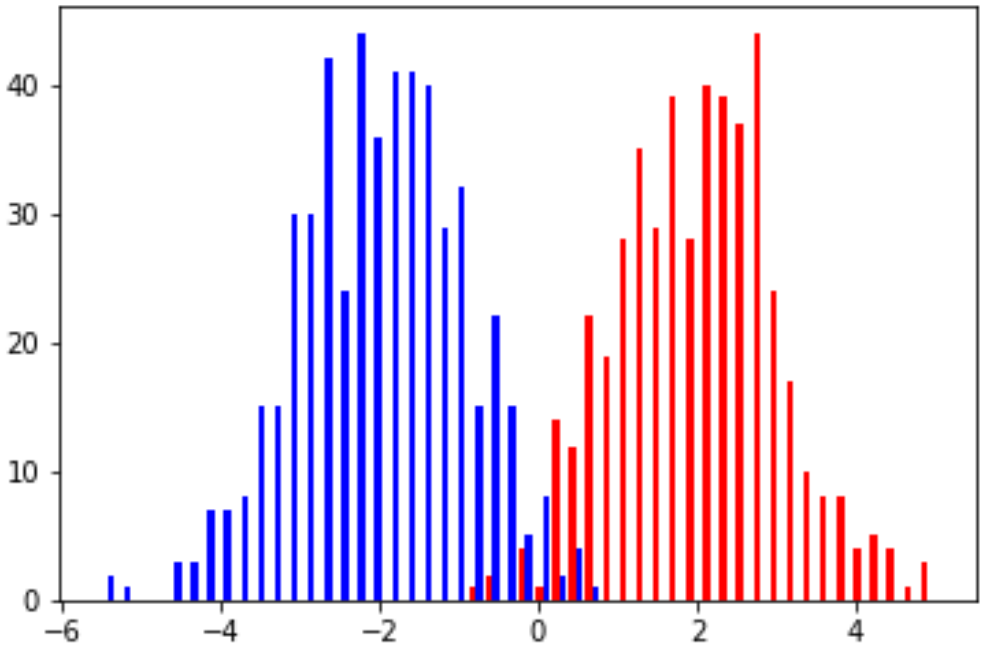


Figure: 1K examples generated with the previous process.

Empirical Risk Minimization: Definition

Empirical Risk Minimization (ERM): given the training set S and the hypothesis class $\mathcal{H}$

$$h \in \operatorname*{argmin}_{h \in \mathcal{H}} L_S(h) \tag{9}$$

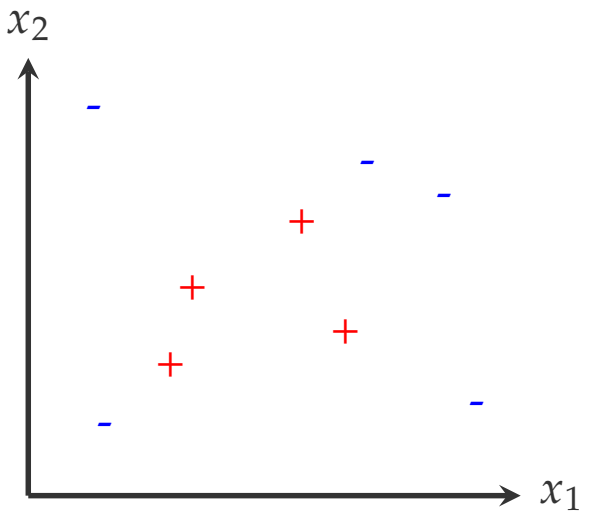
- argmin stands for the set of hypotheses in $\mathcal{H}$ that achieve the minimum value of $L_S(h)$ over $\mathcal{H}$
- In general, there is always at least one hypothesis that makes $L_S(h) = 0$ with an *unrealistically* large $\mathcal{H}$

Empirical Risk Minimization: Limitation

For example, with an *unrealistically* large hypothesis class $\mathcal{H}$, we can always minimize the empirical error and make it zero

$$h_S(\boldsymbol{x}) = \begin{cases} y_i & \text{if } (\boldsymbol{x} = \boldsymbol{x}_i) \wedge (\boldsymbol{x}_i \in S) \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

no matter how many instances in S

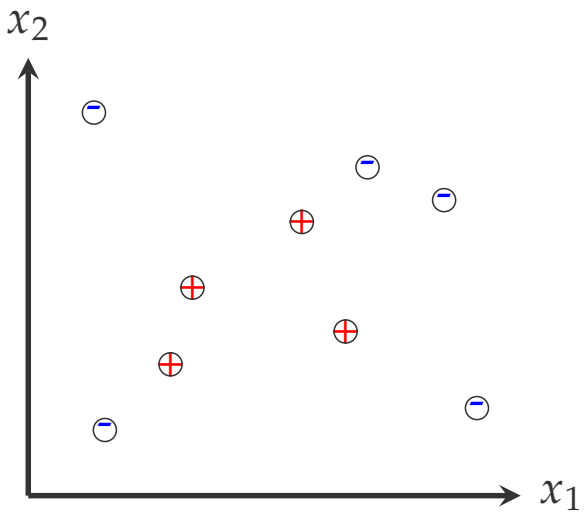


Empirical Risk Minimization: Limitation

For example, with an *unrealistically* large hypothesis class $\mathcal{H}$, we can always minimize the empirical error and make it zero

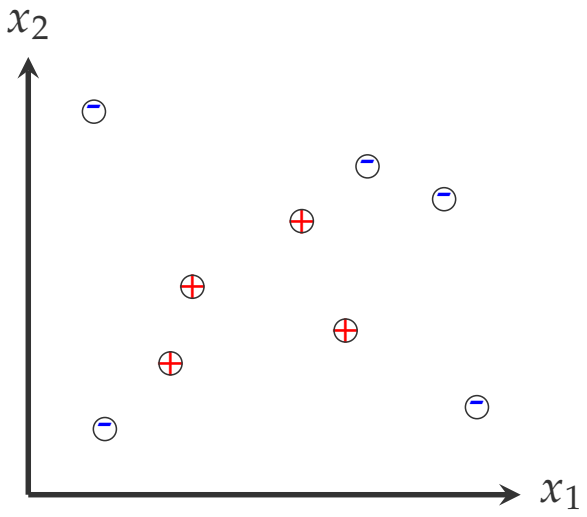
$$h_S(\boldsymbol{x}) = \begin{cases} y_i & \text{if } (\boldsymbol{x} = \boldsymbol{x}_i) \wedge (\boldsymbol{x}_i \in S) \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

no matter how many instances in S



Overfitting

Although this is just an extreme case, it illustrates an important phenomenon, called *overfitting*



- The performance on the training set is excellent; but on the whole distribution was very poor
- Continue our discussion on lecture 6: model selection and validation

Inductive Bias

"A learner that makes no a priori assumptions regarding the identity of the target concept[2] has no rational basis for classifying any unseen instances."

[Mitchell, 1997, Page 42]

[2]labeling function, in the context of our discussion

Finite Hypothesis Classes

A Learning Problem

Assume we know the following information:

- Domain set $\mathcal{X} = [0, 1]$
- Distribution $\mathcal{D}$: the uniform distribution over $\mathcal{X}$
- Label set $\mathcal{Y} = \{-1, +1\}$
- Labeling function f

$$f(x) = \begin{cases} -1 & 0 \leq x < b \\ +1 & b \leq x \leq 1 \end{cases} \tag{11}$$

with b is unknown

A Learning Problem

Assume we know the following information:

- Domain set $\mathcal{X} = [0, 1]$
- Distribution $\mathcal{D}$: the uniform distribution over $\mathcal{X}$
- Label set $\mathcal{Y} = \{-1, +1\}$
- Labeling function f

$$f(x) = \begin{cases} -1 & 0 \leq x < b \\ +1 & b \leq x \leq 1 \end{cases} \tag{11}$$

with b is unknown

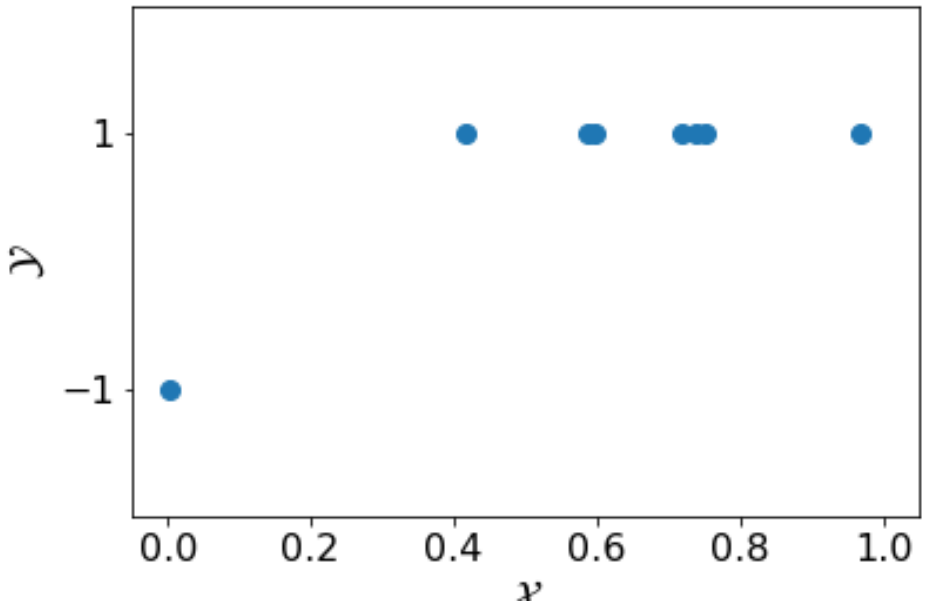
The learning problem is defined as

- Given a set of observations $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, is there a learning algorithm that can find f (or identify b)?

A Training Set S

Consider the following training sets, each of them contains 8 data points, can a learning algorithm find the dividing point?

Training set S[3]



[3]Please refer to the demo code for more examples

Finite Hypothesis Class

- The finite hypothesis class of dividing points

$$\mathcal{H}_f = \{h_i : i \in [10]\} \tag{12}$$

with each h_i defined as

$$f(x) = \begin{cases} -1 & 0 \leq x < \frac{i}{10} \\ +1 & \frac{i}{10} \leq x \leq 1 \end{cases} \tag{13}$$

The Realizability Assumption

The Realizability Assumption:

$$\text{There exists } h^* \in \mathscr{H} \text{ such that } L_{(\mathscr{D},f)}(h^*) = 0$$

[Shalev-Shwartz and Ben-David, 2014, Definition 2.1]

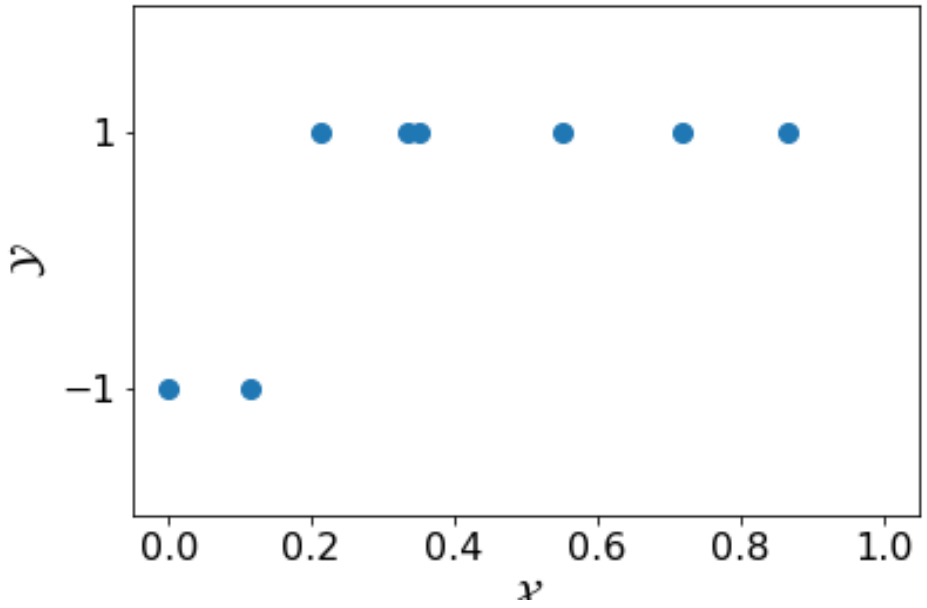
Comments

- $L_{(\mathscr{D},f)}$ indicates this is the true error
- this assumption implies $L_S(h_S) = 0$,

where L_s is the empirical risk based on the training set S and h_S is the hypothesis found by minimizing the empirical risk based on S

A Learning Algorithm

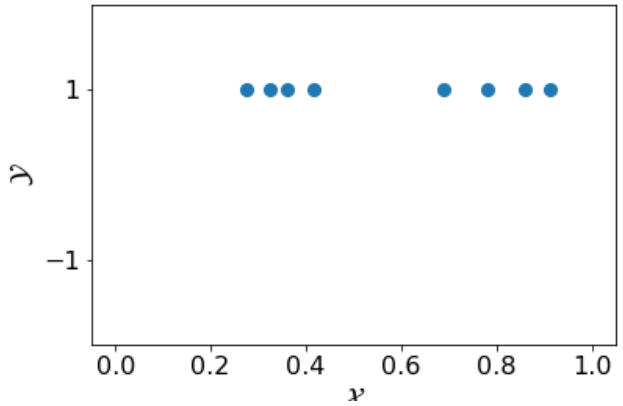
- A learner: the brute force algorithm



  - try the hypotheses one by one and find the best
  - time complexity $\mathcal{O}(|\mathscr{H}_f|)$
- better algorithms exist, such as binary search algorithm

Nonrepresentative Training Set

- Consider the following training set (no negative example)[4]

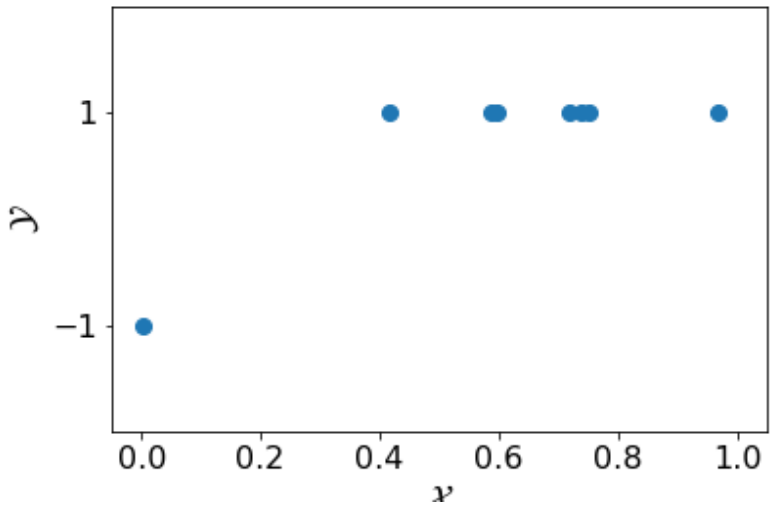


- Introduce $\delta \in (0, 1)$ to capture nonrepresentative cases. With probability $(1 - \delta)$, we have representative cases
  - Loosely speaking, in the running example, at least S has both positive and negative instances
- $(1 - \delta)$ is called confidence parameter

[4]Run the demo code about ten times, you may be able to see this

Nonperfect Predictors

Consider the following training instances

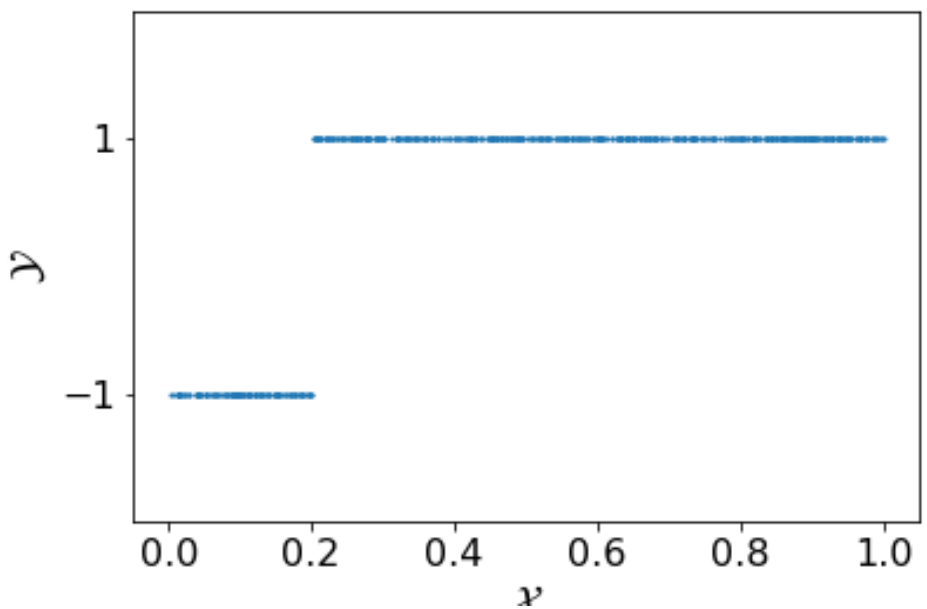


- Follow the realizability assumption, there exists $L_S(h_S) = 0$
- But there is no guarantee that $L_{(\mathcal{D},f)}(h_S) = 0$
- Relax the constraint as

$$L_{(\mathcal{D},f)}(h_S) \leq \epsilon \tag{14}$$

Sample Complexity

- In the running example, we use $m = 8$
- Intuitively, if we increase the size of S, we will have a better chance to identify the labeling function f. For example, when $m = 691$



Summary of the Issues

1. Nonrepresentative training Set
   - Missing critical information about the data distribution $\mathcal{D}$
2. Nonperfect predictors
   - $L_S(h_S) = 0$, but $L_{\mathcal{D}}(h_S) \neq 0$
3. Mismatch of the hypothesis space
   - The realizability assumption is unrealistic for practical applications

The first two issues are considered in the PAC learning model, and the last issue is considered in the agnostic PAC learning model.

PAC Learning

The Realization Assumption

Let keep this assumption in this section

$$\text{There exists } h^* \in \mathcal{H} \text{ such that } L_{(\mathcal{D},f)}(h^*) = 0$$

Comments

- $L_{(\mathcal{D},f)}(h^*)$ is the true error
- It implies, with probability 1, every ERM hypothesis $L_S(h_S) = 0$
- It is a *strong* assumption for theoretical analysis purpose. In practice, we do not have a such guarantee

A Oversimplified Definition of PAC Learnability

A hypothesis class $\mathcal{H}$ is PAC learnable if there exists a learning algorithm with the following property:

- for every distribution $\mathcal{D}$ over $\mathcal{X}$ and
- for every labeling function $f : \mathcal{X} \to \{0, 1\}$

with *enough* training examples, the algorithm returns a hypothesis h such that with a *large* probability that

$$L_{(\mathcal{D},f)}(h) \tag{15}$$

is *arbitrarily small*.

Distribution $\mathcal{D}$ over $\mathcal{X}$

Consider the distribution over $[0, 1]$

- Uniform distribution
- Beta distribution

- Many other distributions

$$p(x; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{16}$$

We expect that, if there exists a learning algorithm A, it should work with all kinds of different distributions.

Labeling Function $f : \mathcal{X} \rightarrow \{0, 1\}$

For the problem of finding the dividing point, the labeling function is defined as

$$f(x) = \begin{cases} -1 & 0 \leq x < b \\ +1 & b \leq x \leq 1 \end{cases} \tag{17}$$

- b can be any number here, as long as it follows the realization assumption. In other words, the labeling function is in the hypothesis space $f \in \mathcal{H}$
- We will discuss the scenario of $f \notin \mathcal{H}$ in next section

A Simplified Definition of PAC Learnability

A hypothesis class $\mathcal{H}$ is PAC learnable if there exists a learning algorithm with the following property:

- for every distribution $\mathcal{D}$ over $\mathcal{X}$
- for every labeling function $f : \mathcal{X} \to \{0, 1\}$, and
- for every $\epsilon, \delta \in (0, 1)$

with *enough* training examples, the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$,

$$L_{(\mathcal{D},f)}(h) \leq \epsilon \tag{18}$$

Accuracy Parameter ϵ

The accuracy parameter ϵ determines how far the output classifier can be from the optimal one

A Simplified Definition

...

$$L_{(\mathcal{D},f)}(h) \leq \epsilon \tag{19}$$

Accuracy Parameter ϵ

The accuracy parameter ϵ determines how far the output classifier can be from the optimal one

A Simplified Definition

...

$$L_{(\mathcal{D},f)}(h) \leq \epsilon \tag{19}$$

Approximately Correct

Confidence Parameter δ

The confidence parameter δ indicates how likely the classifier is to meet the accuracy requirement

A Simplified Definition

. . . the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$ (over the choice of the examples),

$$L_{(\mathcal{D},f)}(h) \leq \epsilon \tag{20}$$

Confidence Parameter δ

The confidence parameter δ indicates how likely the classifier is to meet the accuracy requirement

A Simplified Definition

. . . the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$ (over the choice of the examples),

$$L_{(\mathcal{D},f)}(h) \leq \epsilon \tag{20}$$

Probably Approximately Correct (PAC)

Is It Necessary to Have Both Parameters?

Cen we remove either ϵ or δ?

- We need δ
  - Because the training set is randomly generated, which can be non-representative
- We need ϵ
  - Because we can only finite number of training examples, even though the training set is representative

PAC Learnability

A hypothesis class $\mathcal{H}$ is PAC learnable if there exist a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property:

- for every distribution $\mathcal{D}$ over $\mathcal{X}$,
- for every labeling function $f : \mathcal{X} \to \{0,1\}$, and
- for every $\epsilon, \delta \in (0,1)$,

if the realizable assumption holds wrt $\mathcal{H}, \mathcal{D}, f$, then when running the learning algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$ and labeled by f, the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$,

$$L_{(\mathcal{D},f)}(h) \leq \epsilon \tag{21}$$

Sample Complexity

- Sample complexity function: a function of ϵ and δ

$$m_{\mathcal{H}}(\epsilon, \delta) : (0, 1)^2 \rightarrow \mathbb{N} \tag{22}$$

- How many examples are required to guarantee a probably approximately correct solution
  - many different options
- To be precise, $m_{\mathcal{H}}(\epsilon, \delta)$ is defined to the minimal function that satisfies the requirements of PAC learning with ϵ and δ

Finite Hypothesis Class

Let $\mathcal{H}$ be a finite hypothesis class. Let $\delta \in (0,1)$ and $\epsilon > 0$ and let m be an integer that satisfies

$$m \geq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \tag{23}$$

Then, for any labeling function f, and for any distribution $\mathcal{D}$, for which the realizability assumption holds, with probability $1 - \delta$ over the choice of an i.i.d. sample S of size m, we have that for every ERM hypothesis, h_S, it holds that

$$L_{(\mathcal{D},f)}(h_S) \leq \epsilon. \tag{24}$$

[Shalev-Shwartz and Ben-David, 2014, Corollary 2.3]

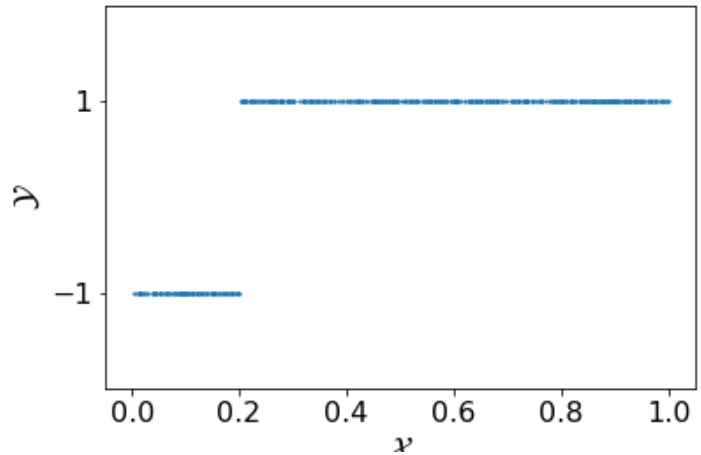
Example: Finding the Dividing Points

The sample complexity of finite hypothesis space

$$m \geq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \tag{25}$$

- The size of the hypothesis space: $|\mathcal{H}| = 100$
- Confidence parameter: $\delta = 0.1$
- Accuracy parameter: $\epsilon = 0.01$

$$m_0 = \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \approx 691$$



Agnostic PAC Learning

Reconsider the Realizability Assumption

The Realizability Assumption

There exists $h^* \in \mathcal{H}$ such that

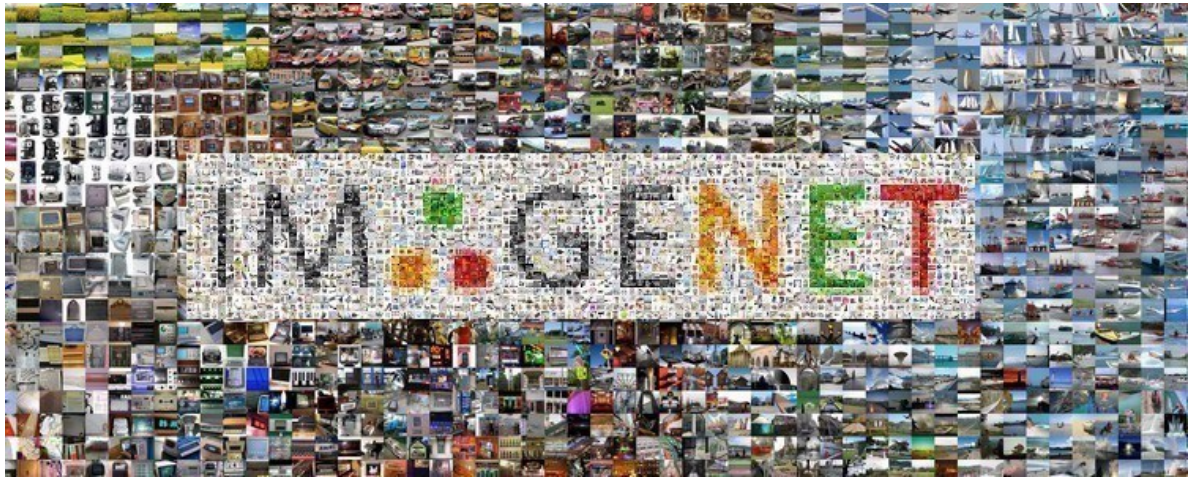
$$L_{(\mathcal{D},f)}(h^*) = \mathbb{P}_{x\sim\mathcal{D}}[h^*(x) = f(x)] = 0 \tag{26}$$

Comment: a strong assumption

- Do we really know f?
- Does equation 26 also holds?

Example: Unrealistic assumption

Image classification



14M images, 20K categories

Notation Revision

- Remove the labeling function f from the framework of PAC learning
- Modify the definitions
  - Revise $\mathcal{D}$ as a *joint* distribution over $\mathcal{X} \times \mathcal{Y}$
  - Revise the *true* risk of a prediction rule h to be

$$L_{\mathcal{D}}(h) = \mathbb{P}_{(x,y)\sim\mathcal{D}}[h(x) \neq y] \tag{27}$$

  - Revise the empirical risk remains the same

$$L_S(h) = \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m} \tag{28}$$

- No fundamental changes, just for the convenience of notations
- all other things remain the same

Agnostic PAC Learnability

A hypothesis class $\mathcal{H}$ is agnostic PAC learnable if there exist a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property:

- for every distribution $\mathcal{D}$ over $\mathcal{X} \times \{-1, +1\}$ and
- for every $\epsilon, \delta \in (0,1)$,

when running the learning algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$,

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon \tag{29}$$

Comments

- In general, we have

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon \tag{30}$$

- If the realizability assumption holds, by the definition we have

$$\min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') = 0 \tag{31}$$

and then,

$$\begin{aligned} L_{\mathcal{D}}(h) &\leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon \\ &= \epsilon \end{aligned}$$

which is a special case of agnostic PAC learning

The Bayes Optimal Predictor

If we know the underlying data distribution $\mathcal{D}$, what will be the best hypothesis in agnostic PAC learning?

The Bayes Optimal Predictor

If we know the underlying data distribution $\mathcal{D}$, what will be the best hypothesis in agnostic PAC learning?

- The Bayes optimal predictor: given a probability distribution $\mathcal{D}$ over $\mathcal{X} \times \{-1, +1\}$, the predictor is defined as

$$f_{\mathcal{D}}(x) = \begin{cases} 1 & \text{if } \mathbb{P}[y = 1|x] \geq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

The Bayes Optimal Predictor

If we know the underlying data distribution $\mathcal{D}$, what will be the best hypothesis in agnostic PAC learning?

- The Bayes optimal predictor: given a probability distribution $\mathcal{D}$ over $\mathcal{X} \times \{-1, +1\}$, the predictor is defined as

$$f_{\mathcal{D}}(x) = \begin{cases} 1 & \text{if } \mathbb{P}[y = 1|x] \geq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

- No other predictor can do better: for any predictor h

$$L_{\mathcal{D}}(f_{\mathcal{D}}) \leq L_{\mathcal{D}}(h) \tag{33}$$

- *Exercise*: The Bayes predictor defined in Eq. 32 is optimal

Example

Consider the following data distribution

$$\mathcal{D} = \underbrace{\frac{1}{2}\mathcal{B}(x;4,1)}_{f(x)=+1} + \underbrace{\frac{1}{2}\mathcal{B}(x,1,4)}_{f(x)=-1} \tag{34}$$

where $\mathcal{B}(x,\alpha,\beta)$ is a Beta distribution with parameters α and β

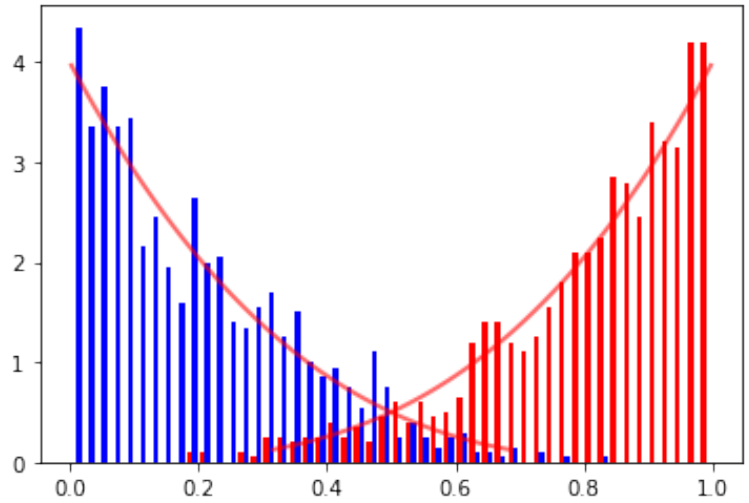
Example

Consider the following data distribution

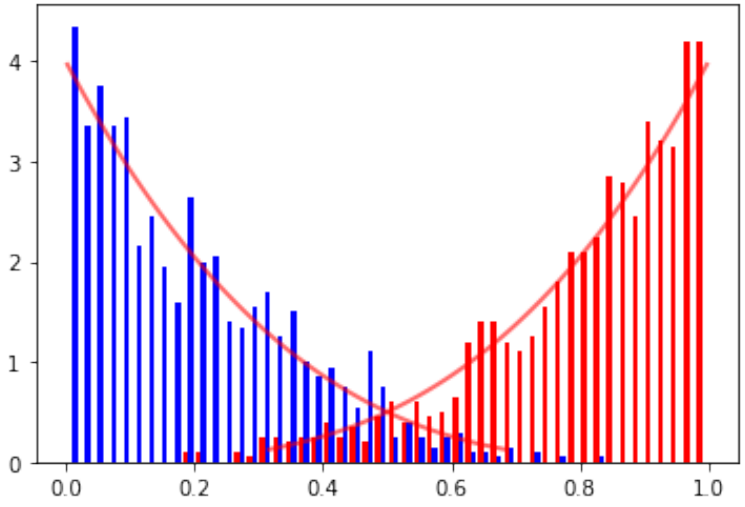
$$\mathcal{D} = \underbrace{\frac{1}{2}\mathcal{B}(x;4,1)}_{f(x)=+1} + \underbrace{\frac{1}{2}\mathcal{B}(x,1,4)}_{f(x)=-1} \tag{34}$$

where $\mathcal{B}(x,\alpha,\beta)$ is a Beta distribution with parameters α and β

The true error of the Bayes predictor is $L_{\mathcal{D}}(f_{\mathcal{D}}) = 0.0625$



Example (Cont.)



With 2K training examples, we can find h_S by minimizing the empirical risk $L_S(h)$

- the empirical risk of h_S, $L_S(h_S) = 0.0535$ (threshold $b = 0.4996$)
- the true risk of h_S, $L_{\mathcal{D}}(h_S) = 0.06250018$

Reference

Mitchell, T. M. (1997).
Machine learning.
McGraw-Hill.

Shalev-Shwartz, S. and Ben-David, S. (2014).
Understanding machine learning: From theory to algorithms.
Cambridge university press.