# Concept learning:

For most of this course, we will focus on the problem of concept learning, here are some useful definitions:

- **Instances:** an instance $x$ is a single observation / measurement, typically represent by its value on some set of attributes or variables.

  For example:

  $$x = (0, 1, 1, 0, 1) \quad \text{or} \quad x = (0.6, -3.4, 2.8, 7.2)$$

  Note: A lot of the time, instance $x$ is a vector and its attributes are the dimension.

- **Instance space:** An instance space $X$ is a space from which all instances $x$ are drawn. i.e: $x \in X$

  For example:

  $$X = \{0, 1\}^n \quad \text{or} \quad X = \mathbb{R}^n$$

- **Labelled example:** An instance with a label. For example:

  $$(0, 1, 1, 0, 1, +) \quad \text{or} \quad (0.6, -3.4, 2.8, 7.2, -)$$

- **Concept:** Is a function that applies on instance space $X$ and returns binary output (basically an if... else clause)

  For example:

  Apply concept $x_1 \wedge \overline{x_3}$ over the set $\{0,1\}^3$ (or $\mathbb{R}^3$):

  - instance $(1,0,0)$ evaluates to 1 (since $x_1$ is 1, $\overline{x_3}$ is 1)
  - instance $(1,1,1)$ evaluates to 0 (since $x_1$ is 1, but $\overline{x_3}$ is 0)

- **Concept class:** is a collection of concepts. We can understand it as a bigger function with a bunch of if... else (each if... else is a concept)

# 2.1 The Statistical Learning Framework

- Consider example where we are trying to learn if a papaya is tasty or not based of their features (i.e softness, color).
- We build a learner to do this task.
- **The learner input:**
  - **Domain set:**
    - An arbitrary set, $X$, that we wish to label.
    - $X$ is an instance space of instances, where an instance is:

$$x_i = (\underbrace{1}_{\text{feature 1}}, \underbrace{0}_{\text{feature 2}}) \quad \text{where} \begin{cases} 0 < i \leq n \\ n: \text{number of instances} \end{cases}$$

  - **Label set:**
    - For our example, since we are deciding if a papaya is tasty or not. We use a 2 elements set:

$$Y = \{0, 1\} \quad \text{where:} \begin{cases} 0 \text{ means not tasty} \\ 1 \text{ means tasty} \end{cases}$$

    - So an instance that is labelled is:

$$x_{i\,(\text{labelled})} = (\underbrace{x_i}_{\substack{\text{unlabelled} \\ \text{instance}}}, \underbrace{y_i}_{\text{label}}) \quad \text{where:} \quad 0 < i \leq n$$

$n$: number of instances

  - **Training data:**

A sequence of labelled instances (domain points). For example:

$$S = ((x_1, y_1) \dots (x_n, y_n))$$

- **The learner's output:**
  - The learner is requested to output a prediction rule,

$$h: X \rightarrow Y$$

function $h$ is also called: predictor, hypothesis, classifier

- How is the training data is generated ?
  - We first sample instance space $X$ over some probability distribution $D$, denoted $X \sim D$.
    For example: $X \sim \text{Norm}(0, 1)$
  - Then we label the instances with some function $f: X \to Y$, that means $y_i = f(x_i)$ for all $i$

  Why sampling by distribution before labelling ?
  - Helps in the process of determine label. For example, if test scores are Normally distributed, we can label "top student" by getting the top 25% percentile.
  - Generalization, since probability distributions represent real-world variability
  - Scalability: it is impossible to label all data, sampling by distributions allows you to manage a subset of data for labelling, training efficiently.

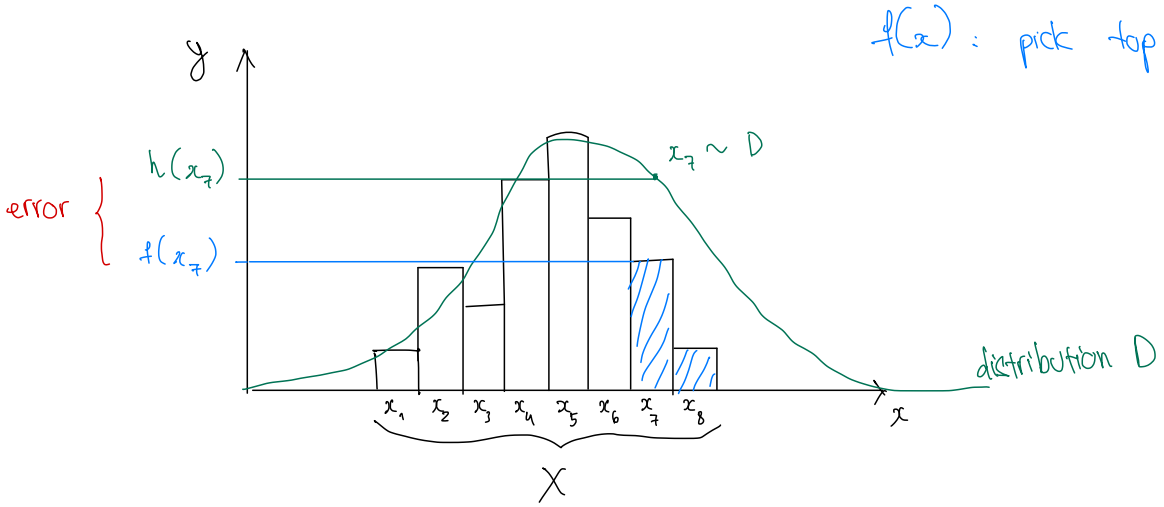
- How do we measure success ?
  - Error of a classifier is the probability that it does not predict the correct label generated by the sampling distribution.
    In other words, the error of $h$ is the probability to draw a random instance $x$ based of the distribution $D$, such that $h(x) \neq f(x)$
  - Formally, given a domain subset $A \subset X$, probability function $D$
    - $D(A)$ determine how likely it is for any instance $x \in A$.
      A can be thought of as event and expressed as:
      $$A = \{ x \in X : \pi(x) = 1 \} \quad \text{where } \pi: X \to \{0, 1\}$$

- $\mathbb{P}_{x \sim D}[\pi(x)]$ is used to express $D(A)$. It means "probability of an arbitrary $x$ drawn based off distribution $D$ to also be in subset $A$"
- Definition of error of a prediction rule, $h: X \to Y$

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

- In words, the error of $h$ is the probability of choosing randomly $x$ for which $h(x) \neq f(x)$
- There are some synonyms for $L_{D,f}(h)$, these are:
  - generalization error
  - risk
  - true error

## Visualize loss (error)




## What information available to the learner?

- The learner is blind to the distribution $D$ and labelling function $f$.
- In our example, we have just arrived at the island and have no clue how papayas are distributed and how should we label one as "tasty".
- The only way the learner can interact with the environment is through observing the training set.

## 2.2 Empirical Risk Minimization

- Since the learner doesn't know what $D$ and $f$ are, the true error is not directly available to the learner.
- What the learner have access to is the training set $S$, so they can calculated training error

**Definition (Training error)**

$$L_S(h) = \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m} \quad \text{or} \quad L_S(h) = \frac{1}{m}\sum_{i=1}^{m} 1_{h(x_i) \neq y_i}$$

where $[m] : \{1, \dots, m\}$

→ Other terms: empirical error, emperical risk

→ Empirical Risk Minimization: is the learning paradigm that coming up with a predictor $h$ that minimizes $L_S(h)$, is also called ERM

### Overfitting

Consider this example:

- Let assume the true error is $L_D(h_S) = \frac{1}{2}$
- Given this predictor:

$$h_S(x) = \begin{cases} y_i & \text{if } \exists i \in [m] \text{ s.t } x_i = x \\ 0 & \text{otherwise} \end{cases}$$

So clearly, the empirical risk is $L_S(h_S) = 0$

- We can see that $L_D(h_S) >> L_S(h_S)$, this is "overfiting"

**Definition (Overfitting)**

Overfitting occurs when our hypothesis fits the training data too well

## 2.3 Empirical Risk Minimization with Inductive Bias

- With the argument of "overfitting", we just prove that ERM paradigm might lead to overfitting.
- To prevent "overfitting", we can use ERM with Inductive Bias

### Definition (ERM with Inductive Bias)

- ERM with Inductive Bias says that a set of predictors $H$ should be chosen in advance (before seing the data), each $h$ inside this $H$ is a function mapping from $X \to Y$, so:

$$H = \{ h_1, h_2, \dots, h_n \}$$

where $h_i : X \to Y$

- The learner, $ERM_H$, use ERM rule to choose a predictor $h \in H$ with the lowest possible error over $S$. So:

$$ERM_H(S) \in \underset{h \in H}{\operatorname{argmin}} L_S(h)$$

where: argmin stands for set of set of hypotheses in $H$ that achieve the minimum values of $L_S(h)$ over $H$

### Why ERM with Inductive Bias ?

By doing ERM with Inductive Bias, we restricting the learner to choosing a predictor from $H$, we "bias" it towards a particular set of predictors

### Trade off of ERM with Inductive Bias

Choosing more restricted hypotheses class $H$ will better prevent us from overfitting but also might cause us a stronger inductive bias

## 2.3.1 Finite Hypothesis Classes

The simplest type of restriction on hypothesis class is imposing an upper bound on its size. In other words, restricting number of predictors h in $\mathcal{H}$.

For example, $\mathcal{H}$ can be a set of all predictors that can be implement by a C++ program written in at most $10^9$ bits of code.

Analyze performance of $ERM_{\mathcal{H}}$ assume that $\mathcal{H}$ is finite:

- Let $S$ be traing sample, $f: X \to Y$ be label function, $h_S$ be result of applying $ERM_{\mathcal{H}}$ to $S$, so:

$$h_S \in \underset{h \in \mathcal{H}}{\operatorname{argmin}} L_S(h)$$

- Under these assumptions:
  - The realizability assumption: implies that for every ERM hypothesis we have that $L_S(h_S) = 0$
  - The i.i.d assumption: the examples in training set $S$ is i.i.d on distribution $D$. Denoted $S \sim D^m$

    where
    - $m$: size of $S$
    - $D^m$: probability over m-tupples induced by applying $D$ to pick each element of the tuple indepently.
- $L_{D,f}(h_S)$ depends on the training set, $S$, proven below:
  - Randomness in training set $S$: the training set is selected through a random process from distribution $D^m$ <i.i.d assumption>
  - Randomness in predictor $h_S$: the predictor $h_S$ is trained / chosen based on training set $S$
  - Therefore, $L_{D,f}(h_S)$ depends on $h_S$ depends on $S$

- However, just because $L_{D,f}(h_S)$ depends on $S$, it is unrealistic to say that having $S$ means having a good learner (there is always a chance that $S$ does not represent distribution $D$ well).

The metric used to assess "how bad a traing set $S$ is" is "probability of picking $S$ s.t $L_{D,f}(h_S)$ larger than certain error threshold"

↳ denoted as: $D^m(\{S|_x : L_{D,f}(h_S) > \varepsilon\})$

where $\varepsilon$, called accuracy parameter, is the error threshold

- In order to analyze the performance of $ERM_H$ with finite $H$, we need to find the upper bound (worst case scenario) of the above probability. 👆 Lets find the upper bound:
  - Use the fact that $H$ is finite to bound the event $L_{D,f}(h_S) > \varepsilon$, let $H_B \subseteq H$ be "bad hypotheses" since it leads to high loss value, denoted as:

$$H_B = \{h \in H : L_{D,f}(h) > \varepsilon\}$$

In addition, let $M$ be set of misleading samples, meaning having low training loss $L_S(h)$, but high true loss $L_{D,f}(h) > \varepsilon$

$$M = \{S|_x : \exists h \in H_B, L_S(h) = 0\}$$

$$\text{or } M = \bigcup_{h \in H_B} \{S|_x : L_S(h) = 0\}$$

  - Recall the cycle of ERM with Inductive Bias:

Pick sample $S$ → Calculate $L_S(h)$ (step 2)

Pick set $H$ (step 1)

↓

Pick predictors $h_S$ with lowest loss (step 3) → Calculate $L_{D,f}(h_S)$ (step 4)

Realizability assumption stated that in step 2, we will always have $L_S(h_S) = 0$, meaning $h_S$ picked in step 3 fits "perfectly" to $S$.

So when error happens in step 4, $L_{D,f}(h_S) > \epsilon$, must be due to some discrepancies not captured by $S$. That mean $h_S$ belongs to the bad hypotheses $H_B$. In short:

$$\begin{aligned} & | \quad h_S \in H_B \\ \Rightarrow \quad & \{S|x : L_{D,f}(h_S) > \epsilon\} \subseteq \bigcup_{h \in H_B} \{S|x : L_S(h) = 0\} \\ \Rightarrow \quad & D^m(\{S|x : L_{D,f}(h_S) > \epsilon\}) \leq D^m\left(\bigcup_{h \in H_B} \{S|x : L_S(h) = 0\}\right) \end{aligned}$$

(equation 1)

- Next, we upper bound the RHS of equation 1 using union bound:

**Lemma 2.2 (Union bound)**

For any 2 set $A, B$ and distribution $D$

$$D(A \cup B) \leq D(A) + D(B)$$

Equation 1's RHS with union bound:

$$D^m\left(\bigcup_{h \in H_B} \{S|x : L_S(h) = 0\}\right) \leq \sum_{h \in H_B} D^m(\{S|x : L_S(h) = 0\})$$

(equation 2)

- Next, we bound the RHS of equation 2 using the fact that if $h$ is fixed, event $L_S(h) = 0$ is equivalent to event $\forall i, h(x_i) = f(x_i)$. And since samples in $S$ are i.i.d. So:

$$\begin{aligned} D^m(\{S|x : L_S(h) = 0\}) &= D^m(\{S|x : \forall i, h(x_i) = f(x_i)\}) \\ &= \prod_{i=1}^{m} D(\{x_i : h(x_i) = f(x_i)\}) \end{aligned}$$

(equation 3)

- Next, we try to quantity the result from equation 3 in terms of $\epsilon$. Consider an individual element in $\prod_{i=1}^{m} D(\{x_i : h(x_i) = f(x_i)\})$:

$$\begin{aligned} D(\{x_i : h(x_i) = f(x_i)\}) &= 1 - L_{D,f}(h) \quad \leftarrow \text{why?} \\ &\leq 1 - \epsilon \quad < h \in H_B, \\ & \qquad\qquad L_{D,f}(h_S) > \epsilon > \end{aligned}$$

(equation 4)

- Combine equation 3 and 4, we have:

$$D^m(\{S|_x : L_S(h) = 0\}) \leq (1-\varepsilon)^m \leq e^{-\varepsilon m} \quad \langle 1-\varepsilon \leq e^{-\varepsilon} \rangle$$

(equation 5)

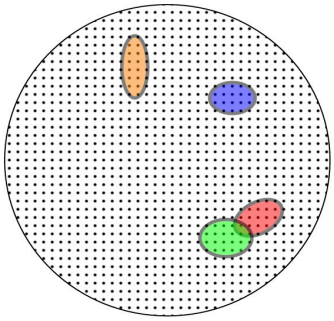
- Using equation 5 and apply to equation 2, we conclude that:

$$D^m(\{S|_x : L_{D,f}(h_S) > \varepsilon\}) \leq |H_B| \cdot e^{-\varepsilon m} \leq |H| \cdot e^{-\varepsilon m} \quad \langle H_B \subseteq H \rangle$$

$\leq \delta$ (our goal)

What this means in human language?

- Consider this picture of large circle representing $m$-tuple of instances



- Each colored-area represent a set of "misleading" instances for some bad hypothesis $h$.
- From the analysis above, we know that $ERM_h$ can leads to "overfit". In other words, for some $h \in H_B$, we have $L_S(h) = 0$.

  Equation 5 says that for each bad hypothesis $h$, we have at most $(1-\varepsilon)^m$ chance of getting "misleading" training sets.

  The larger $m$ is, the smaller the chance of being misleading
- The final result only formalize the point mentioned in above point, with all $h \in H_B$ instead of just 1 fixed $h$.

This result leads to corallary 2.3

## Corollary 2.3

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

$$m \geq \frac{\log(|H|/\delta)}{\varepsilon}$$

Then for any label function $f$, any distribution $D$, for which realizability holds (meaning, for some $h \in H$, $L_{D,f}(h) = 0$), with probability of at least $1-\delta$ over the choice of an i.i.d. sample $S$ of size $m$, we have that for every ERM hypothesis, $h_S$:

$$L_{D,f}(h_S) \leq \varepsilon$$

→ probability of misleading sample

### What this means in human language?

For a sufficiently large $m$, the $ERM_H$ rule over a finite hypothesis class will be probably (with confidence $1-\delta$) approximately (up to an error of $\varepsilon$) correct.

### Another way to reasonate $D^m(\{S|_x : L_S(h) = 0\}) \leq (1-\varepsilon)^m$.

- Here we are fixing $h$, and assume that it is picked by the learner, hence $L_S(h) = 0$ (with realizability assumption)
- Lets assume $h$ is a "bad learner", so it has true error larger than $\varepsilon$:

$$\begin{aligned}
& L_{D,f}(h) > \varepsilon \\
\Leftrightarrow \quad & P_{x\sim D}[h(x) \neq f(x)] > \varepsilon && \text{<definition>} \\
\Leftrightarrow \quad & 1 - P_{x\sim D}[h(x) \neq f(x)] \leq 1-\varepsilon \\
\Leftrightarrow \quad & P_{x\sim D}[h(x) = f(x)] \leq 1-\varepsilon && \text{→ prob } h \text{ correctly classifies a single new example from } D \\
\Leftrightarrow \quad & P_{x\sim D}[\text{all correcets in } S] \leq 1-\varepsilon \\
\Leftrightarrow \quad & D(\{S|_x : L_S(h) = 0\}) \leq 1-\varepsilon
\end{aligned}$$

- Generalize this to m examples:

$$D^m(\{S|_x : L_S(h) = 0\}) \leq (1-\varepsilon)^m$$

→ prob h correctly classifies every example in a set S of m size.

- Generalize this even more, to the case where $h \in H$ is not fixed.

$$D^m\left(\bigcup_{h \in H} \{S|_x : L_S(h) = 0\}\right) \leq |H| \cdot (1-\varepsilon)^m$$

→ prob a random $h \in H$ correctly classifies every example in a set S of m size, but is a "bad learner"

Since $D^m(\{S|_x : L_{D,f}(h) > \varepsilon\}) \leq D^m\left(\bigcup_{h \in H} \{S|_x : L_S(h) = 0\}\right)$:

$$D^m(\{S|_x : L_{D,f}(h) > \varepsilon\}) \leq |H| \cdot (1-\varepsilon)^m$$

failure prob: prob that random $h \in H$ fail to meet the desired accuracy $\varepsilon$.

- We are trying to bound probability of failure by $\delta$:

$$|H| \cdot (1-\varepsilon)^m \leq \delta$$

- Solving this for m, we find the sample complexity:

$$|H| \cdot (1-\varepsilon)^m \leq \delta$$

$$\Leftrightarrow |H| \cdot e^{-\varepsilon m} \leq \delta \qquad \langle\, 1 + x \approx e^x \,\rangle$$

$$\Leftrightarrow e^{-\varepsilon m} \leq \frac{\delta}{|H|}$$

$$\Leftrightarrow -\varepsilon m \leq \log\left(\frac{\delta}{|H|}\right)$$

$$\Leftrightarrow m \geq \frac{\log\left(\frac{|H|}{\delta}\right)}{\varepsilon}$$