1. **Probably Approximately Correct**

Valiant’84 “Theory of the Learnable”; Turing Award’14
Average case performance wrt a fixed instance distribution
Assume instances $x \in X$ are drawn from a distribution $\mathcal{D}$ (unknown and arbitrary)
(Training phase) Given independent samples $(x, c(x))$, all labelled by an unknown concept $c \in \mathcal{C}$
Goal: Output hypothesis $h \subseteq X$ s.t. $\text{err}_{\mathcal{D}}(h, c) \stackrel{\text{def}}{=} \mathbb{P}_{x \sim \mathcal{D}}[h(x) \neq c(x)]$ is small
Equivalently $\text{err}_{\mathcal{D}}(h, c) = \mathbb{P}_{x \sim \mathcal{D}}[x \in h \triangle c]$
Recall $h \triangle c \stackrel{\text{def}}{=} (h \setminus c) \cup (c \setminus h)$ (symmetric difference)
Error cannot always be small: if unlucky, training samples may be useless
New goal: With high probability over training samples and internal randomness (probably), output hypothesis $h \subseteq X$ with small error (approximately correct)

Algorithm $A$ **PAC learns** $\mathcal{C}$ if
for any concept $c \in \mathcal{C}$
for any distribution $\mathcal{D}$ over $X$
  for any confidence parameter $\delta > 0$ and accuracy parameter $\varepsilon > 0$
  when $A$ takes $m$ samples from $\text{EX}(c, \mathcal{D})$
  with probability $\geq 1 - \delta$ over the samples and $A$’s randomness
  output hypothesis $h \subseteq X$ such that $\text{err}_{\mathcal{D}}(h, c) \leq \varepsilon$

$A$ is **efficient** if runs in $\text{poly}(1/\delta, 1/\varepsilon)$ time (plus two more conditions below)$\,$
  $\text{poly}(1/\delta, 1/\varepsilon)$ means at most polynomial in $1/\delta$ and $1/\varepsilon$ (e.g. at most $\varepsilon^{-2}\delta^{-1}$)
  or $\text{poly}(n, 1/\delta, 1/\varepsilon)$ time if $X = \{0, 1\}^n$ or $\mathbb{R}^n$

Run time always $\geq m$ (just to read the samples)

Algorithm $A$ only knows $\mathcal{C}, \delta, \varepsilon$
$A$ doesn’t know $\mathcal{D}$ (distribution independent learning)
$A$ works under any $\mathcal{D}$ (strong assumption!), but error is also evaluated under $\mathcal{D}$

2. **PAC Learning Rectangles**

$X = \text{the plane } \mathbb{R}^2 \quad \mathcal{C} = \text{axis-aligned rectangles } = \{R(x_1, y_1, x_2, y_2) \mid x_1, y_1, x_2, y_2 \in \mathbb{R}\}$
where $R(x_1, y_1, x_2, y_2) = \{(x, y) \in \mathbb{R}^2 \mid x_1 \leq x \leq x_2 \text{ and } y_1 \leq y \leq y_2\}$
$\mathcal{D} = \text{fixed distribution over } \mathbb{R}^2$ (unknown)

Algorithm

Hypothesis $h = \text{smallest rectangle containing all positive samples}$ (0 if no positive samples)

Claim 2.1. Given any $c \in \mathcal{C}$, if $m \geq (4/\varepsilon) \ln(4/\delta)$, with probability $\geq 1 - \delta$, the Algorithm outputs hypothesis $h$ with $\text{err}_{\mathcal{D}}(h, c) \leq \varepsilon$.

Proof. $h \subseteq c$ always
Want to show $h \triangle c = c \setminus h$ small under $\mathcal{D}$
Can decompose $c \setminus h$ as union of four strips: top, left, bottom, right
top strip $T = \text{rectangle sharing top \\ & left \\ & right sides with } c$, has weight $\varepsilon/4$ under $D$

\begin{center}
\begin{tikzpicture}
\draw[red] (0,0) rectangle (1,1);
\draw (0.5,0) -- (0.5,1);
\draw (0,0.5) -- (1,0.5);
\draw (0,0) -- (1,1);
\end{tikzpicture}
\end{center}

left, bottom, right strips defined analogously
$c' = c$ with top, left, bottom, right strips removed
$(c' = \emptyset$ if $c$ has probability mass $\leq \varepsilon/4$ under $D$, so that top strip is bigger than $c$)

**Claim:** $c' \subseteq h$ with probability $\geq 1 - \delta$

Reason: if each strip contains a sample, then $c' \subseteq h$

top strip has no sample with probability $(1 - \varepsilon/4)^m$
same for other strips, union bound:

$$P[\text{some strip has no sample}] \leq 4(1 - \varepsilon/4)^m \leq 4(e^{-\varepsilon/4})^m \leq \delta$$

c' \subseteq h implies $\text{err}_D(h,c) \leq \varepsilon$

because each strip has probability mass $\varepsilon/4$ under $D$  

---

3. Hypothesis size

some concepts $c(x)$ have a natural **size** (e.g. #bits needed to describe $c$)
e.g. $C = \text{DNF formulae over } X = \{0,1\}^n$
every boolean function $f : X \to \{0,1\}$ can be represented as a DNF

some as a 2-term DNF (e.g. $f(x) = (\overline{x}_1 \land \overline{x}_2 \land x_6) \lor (x_9 \land \overline{x}_4 \land x_2)$)
some requires $\geq 2\sqrt{n}$ terms

$\text{size}(f) = \text{size of the smallest representation of } f$ in $C$
e.g. when $C = \{\text{DNF}\}$, sometimes $\text{size}(f)$ may be #terms

Redefinition: PAC learning Algorithm $A$ is **efficient** if runs in time $\text{poly}(1/\delta, 1/\varepsilon, \text{size}(c))$
or $\text{poly}(n, 1/\delta, 1/\varepsilon, \text{size}(c))$ if $X = \{0,1\}^n$ or $\mathbb{R}^n$
c = target concept

in particular, $A$ cannot output $h$ with large $\text{size}(h)$
Algorithm knows $C, \delta, \varepsilon, \text{size}(c)$

Some $C$ may not have interesting size measure; size can be ignored
e.g. monotone conjunctions have size $\leq n$

---

4. Efficient hypothesis

Often PAC learning Algorithm $A$ outputs hypothesis $h : X \to \{0,1\}$ that is itself a program

Not useful if $h$ too slow

If $X = \{0,1\}^n$ or $\mathbb{R}^n$, hypothesis $h$ is **polynomially evaluable** if $h$ runs in $\text{poly}(n)$ time

PAC learning Algorithm $A$ is **efficient** if it additionally outputs polynomially evaluable hypothesis
e.g. inefficient $A$:

stores all training samples in $h$
then $h$ exhaustively searches for smallest DNF consistent with all training samples