A General Model

- Input/instance space $X$ (e.g. $\mathbb{R}^2$)

- Target concept/function $c \subseteq X$ (positive ex's)
  $c : X \rightarrow \{0, 1\}$ or $\{+3, -3\}$

- Concept class $C$,
  (quite) restricted
  E.g. rectangles in $\mathbb{R}^2$
- We assume \( c \in C \):
  - \( C \) Known to learner
  - \( C \) Unknown
- Input distribution \( P \) over \( X \)
  - \( X \) unknown and arbitrary
- Learner given access to labeled samples \( \{ (x, c(x)) \mid x \in P \} \)
Definition \( C \) is PAC learnable if \( \exists \) learning algo \( L \) s.t.

\[ \forall c \in C \text{ (target)} \]

\[ \forall \rho \text{ over } X \text{ (dist.)} \]

\[ \forall \varepsilon, \delta > 0: \]

- With prob. \( \geq 1-\delta \), \( L \) outputs

\[ h \in C \text{ s.t. } E(h) \leq 3 \]

\[ (E(h) \triangleq \Pr_{x \sim \rho}[h(x) \neq c(x)]) \]

- Sample size & runtime of \( L \) are "efficient", e.g. polynomial in \( \frac{1}{\varepsilon}, \frac{1}{\delta} \) and...
..."complexity" of X and C:

- e.g. $\mathbb{R}^2$ vs $\mathbb{R}^d$, must depend on d, ideally polynomially or better

- e.g. "size" of C:
  - # nodes in decision tree
  - # weights in neural net

- We'll be precise as needed
• What classes \( C \) are PAC-learnable?
• What classes are (provably) not PAC, and why?
• What are general algo tools/reductions?
• What are interesting variations on model?
Theorem The class $C$ of axis-aligned rectangles in $\mathbb{R}^2$ is PAC learnable.
Let's look at another C: conjunctions of Boolean features.

- Domain $X = X_n = \{0, 1\}^n$

- Conjunctions: e.g.,
  
  $c(x) = x_1 \land x_2 \land x_4$, $n = 6$
  
  $c(110100) = 1$
  
  $c(111111) = 0$

- Generalize & specialize rectangles in $\mathbb{R}^2$:
  
  $2 \rightarrow n$

  $x_i \in [a, b] \rightarrow x_i = 0, 1, \ast$
Let \( C \) be class of conjunctions over \( \{0,13\} \).

- What is \( |C| \)?
- Is \( C \) PAC learnable in time polynomial in \( \frac{1}{\varepsilon}, \frac{1}{\delta}, \text{and } n \)?
- Algorithm?
• Initial hypothesis:
  \[ h \leftarrow x_1 \neg x_1 x_2 \neg x_2 \cdots x_n \neg x_n \]

• Given \( \langle x, y \rangle \), \( x \in P \):
  \[
  y = 0 \rightarrow \text{ignore}
  
  y = 1 \rightarrow \text{delete contradictions from } h
  
  \]

  E.g., on 1 1 0 1 \ldots, \( y = 1 \):
  \[
  \text{delete } x_1, x_2, x_3 \ldots
  
  \]
• Every deletion proven
  → most specific h
  → consistent with data so far

• Only mistake: fail to delete some \( x_i, T_x_i \) that is harmful ("bad event")

• Let's analyze for some \( z \in \mathcal{C} \)
  \( (z = x_i \text{ or } \neg x_i) \)
• Define

\[ q(z) = \Pr_{x \sim p} \left[ c(x) = 1 \text{ if } z = 0 \text{ in } x \right] \]

= deletion prob. of \( z \)

• \( e(h) \leq \forall \, q(z) \) 

• call \( z \) bad if \( q(z) \geq \frac{1}{2n} \)

• \( h \) has no bad \( z \) \( \Rightarrow \exists \, e(h) \leq \frac{1}{3} \)
For fixed bad $z$:

- Prob. $z$ not deleted
  \[
  \Pr(z \text{ not deleted}) \leq (1 - \frac{3}{2n})^m \text{ indep.}
  \]

- Prob. some/any bad $z$ not deleted
  \[
  \leq 2n(1 - \frac{3}{2n})^m \text{ union bound}
  \]

Set $\delta$, solve for $m$
Algo is PAC for
\[ m \geq \frac{2^n}{\epsilon} (\ln(2n) + \ln(1/\delta)) \]
Running time \( O(m \cdot n) \)

Q: Even stronger property of algo?
A (slight?) generalization: \( C = 3\text{-term DNF} \)

Now target \( C = T_1 \lor T_2 \lor T_3 \)

Each \( T_i \): a conjunction over \( \{0,1\}^n \)

e.g. \( C = x_1 \lor x_5 \lor x_1 x_2 x_7 \lor \neg x_1 \lor x_2 \)

\( c(x) = T_1(x) \lor T_2(x) \lor T_3(x) \)

Claim: if 3-term DNF is PAC learnable, then \( \text{NP} = \text{RP} \).
The Graph 3-Coloring Problem

Input: undirected graph/network \( G \); e.g.

every edge connects different colors

Output: "yes" if \( G \) 3-colorable, "no" else.

An NP-complete problem.
Encode as 3-term DNF learning problem: create labeled sample $S$.

\[ (\text{vertices}) \]
\[ + \text{ex's} \]
\[ 0111111, + \]
\[ 1011111, + \]
\[ 1101111, + \]
\[ \vdots \]
\[ 1111110, + \]

\[ (\text{edges}) \]
\[ -\text{ex's} \]
\[ 0011111, - \]
\[ 1001111, - \]
\[ 1010111, - \]
\[ 1011011, - \]
\[ \vdots \]
\[ 1111100, - \]
Let's show the following:

A 3-term DNF consistent with $S$ consistent with $S$ $\iff$ $G$ is 3-colorable
Suppose \( G \) is 3-colorable:

\[
TR = \text{all vars/vert3 not red}
\]
\[
TB = x_1x_2x_4x_6x_7
\]
\[
TG = x_1x_3x_4x_5x_7
\]

Claim: \( TR \lor TB \lor TG \) consistent with \( S \).
Now suppose some $T_R \lor T_B \lor T_C$ consistent with $S$.

- Define color of van/ver/kx $i$ to be the $T$ that satisfies $<11.101\ldots 1,+,i>$. 

- If $i \neq j$ both $R$ and $(i,j) \in G$:

$$-0-1 \rightarrow \overset{+}{\Rightarrow} \text{ sat. } T_R$$

$$\Rightarrow \text{ none of } x_i, x_j, x_i, x_j \in T_R$$

$$\Rightarrow -0-0-\text{ sat's } T_R$$

$\Leftarrow$ with consistency!

$\therefore G$ is 3-colorable.
So $G$ is 3-colorable $\iff$ 
$S = S(G)$ is consistent with some 3-term DNF.

So what?

What does this have to do with PAC?

Where are our friends $P, E, \delta$?

Need to simulate them.
Let $A$ be a black-box PAC algo.

Given $G \rightarrow S = S(G)$

Let $P$ be uniform over $S$

$|S| = \#\text{vertices} + \#\text{edges} = \text{size of } G$

Choose $\epsilon < \frac{1}{|S|}$

and any small $\delta > 0$

Run $A$ on $P, \epsilon, \delta$

→ test output $T_1 \lor T_2 \lor T_3$

for consistency
\( G \text{ 3-colorable } \Rightarrow \)
\( \text{w.p. } \geq 1 - \delta, A \text{ output consistent hypothesis.} \)

\( G \text{ not 3-colorable } \Rightarrow \)
\( \text{w.p. } 1, A \text{ fails to output consistent hypo.} \)

\( \therefore \text{PAC learning 3-term DNF } \)
\( \Rightarrow \text{NP = RP.} \)
Moral: As mysterious & powerful as ML can sometimes seem, it obeys same "computation laws" as any other algorithmic problem/framework.

But now let's weasel out of this result.
A little Boolean algebra.

\[ T_1 \lor T_2 \lor T_3 = \bigwedge (u \lor v \lor w) \]

(3-term DNF) \( u \in T_1 \) \( v \in T_2 \) \( w \in T_3 \)

\( (3\text{CNF}) \)

\[ \text{e.g. } T_1 = 1 \Rightarrow \text{ each } u \in T = 1 \]

\[ \Rightarrow \text{ RHS} = 1 \]

\[ \text{LHS} = 0 \Rightarrow \text{ some } u, v, w = 0 \]

\[ \Rightarrow \text{ RHS} = 0 \]

\[ \therefore 3\text{-term DNF} \leq 3\text{CNF} \]

(\( \not\equiv \))
Create meta-features: ("linearization")

- \( \exists (u, v, \omega) \triangleq u \lor v \lor \omega \)

- \# meta-features

\[ \sim (2^n) = O(n^3) \]

- RHS on last page is a conjunction over \( \exists \)'s

- given \( x \in \{0, 1\}^n \)

expand:

\[ x \rightarrow \exists(x) \]

\[ n \rightarrow n^3 \]
So: 3-term DNF is PAC-learnable "by" 3CNF.

We have circumvented the hardness result by enlarging our hypothesis class.
Notes:

• We are using HP part of PAC defn!

• E.g. P uniform over \{0,1\}^n \Rightarrow P' uniform over \{0,1\}^n

• Output of conjuncts algo may not bc = any 3cnf

• Hypo. representation matters

• "Overcompleteness"
Definition. $C$ is PAC learnable if there exists a learning algorithm $H$ such that:

$$\forall c \in C \land \forall P \text{ over } X \text{ (dist.)} \land \forall \delta, \epsilon > 0:\n$$

- With probability $\geq 1 - \delta$, $H$ outputs $h \in H$
  s.t. $\exists \delta(h) \leq \epsilon$
  s.t. $\Pr_{x \sim P} [h(x) \neq c(x)]$

- Sample size and runtime of $L$ are "efficient", e.g. polynomial in $\frac{1}{\epsilon}, \frac{1}{\delta}$ and...
Recap:

• PAC learning 3-term DNF by 3-term DNF is NP-hard.

• 3-term DNF is PAC learnable by 3CNF
Q: Can we ever be sure any C is truly hard to learn?