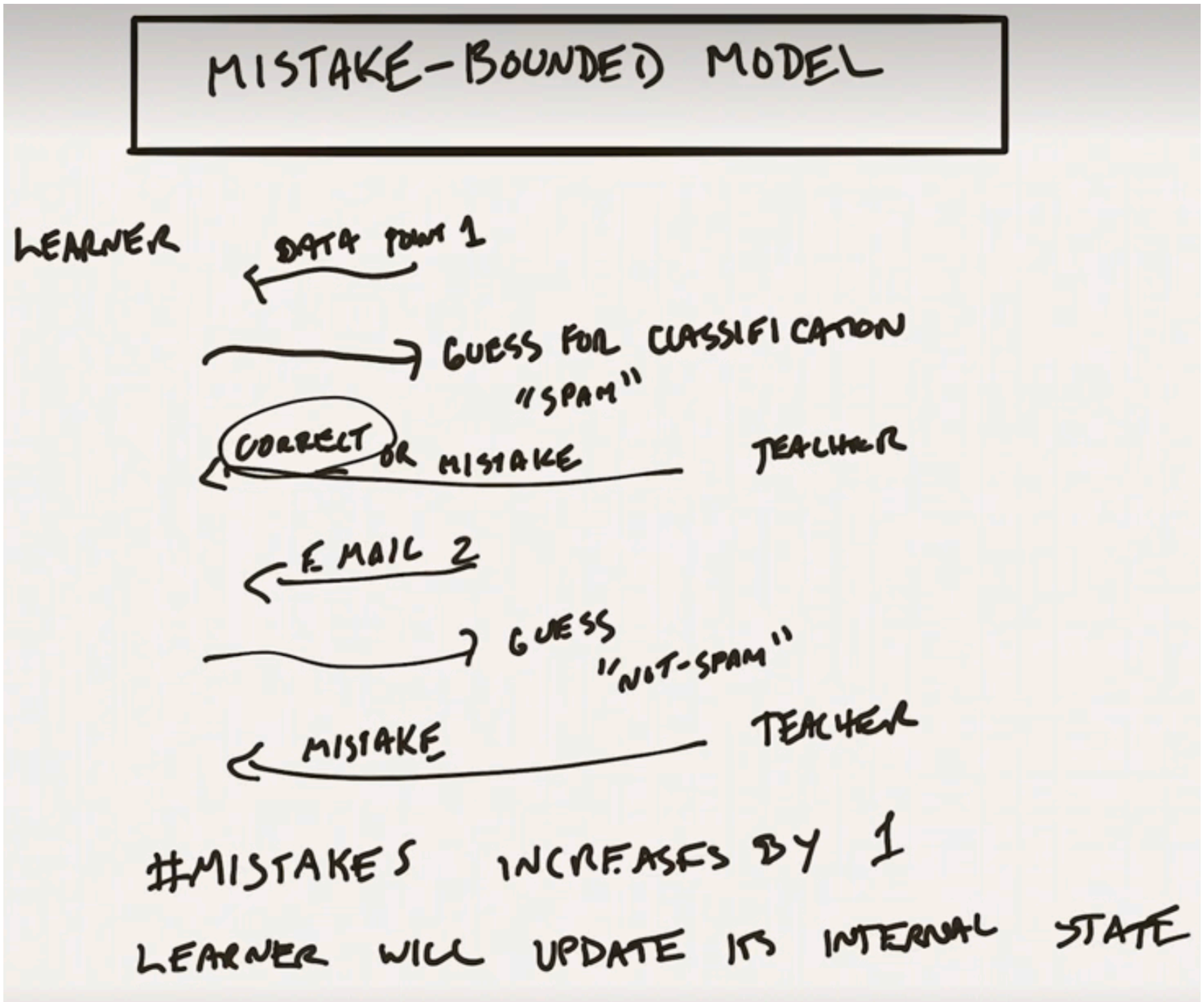
Mistake bounded Learning



WE SAY A LEARNER HAS MISTAKE-BOUND $\underline{t}$

IF FOR EVERY SEQUENCE OF CHALLENGES, LEARNER MAKES AT MOST $\underline{t}$ MISTAKES.

$C = \{$ MONOTONE DISJUNCTIONS ON n VARIABLES $\}$

DOMAIN $= \{0,1\}^n$

e.g. $x_1 \vee x_3$ $\qquad$ $f(x) = x_1 \vee x_7 \vee x_9$

Disjunction is typically OR combination
So our domain in above screenshot is 0 and 1

And monotone disjunction can be X1 v X3 means in bit string if 1 is at 1st or 3rd place that means learner should evaluate to true
Another example is X1 V X7 v X9 so our bit wise string has 1 at 1st or 7th or 9th that means learner should return true.

$f \in C$ $\qquad$ f IS A MONOTONE DIST

"TRYING TO LEARN f"

LEARNER

$\xleftarrow{x \in \{0,1\}^n}$

$\xrightarrow{0 \text{ or } 1}$

TEACHER SAYS
"CORRECT" IF GUESS $= f(x)$
"MISTAKE" OTHERWISE

$\xleftarrow{x' \in \{0,1\}^n}$

EVERY TIME WE MAKE A MISTAKE
AT LEAST ONE LITERAL IS ELIMINATED.

AT MOST n LITERALS

$\Rightarrow$ # MISTAKES $\leq n$.

$\mathcal{C} = \{$ DISJUNCTIONS $\}$

$$f = x_1 \vee \bar{x}_2 \vee x_7 \vee \bar{x}_9$$

QUESTION: HOW CAN WE USE THE ALGORITHM FOR MONOTONE DISJUNCTIONS TO LEARN DISJUNCTIONS?

Above question answer

$x_1 \dots x_n \in \{0,1\}^n$

"FEATURE EXPANSION"

$\underbrace{x_1 \dots x_n}_{n} \longrightarrow \underbrace{x_1, \dots, x_n, y_1, \dots, y_n}_{2n}$

EACH $y_i = \bar{x}_i$

$\underline{0110}$ $(n=4)$ $\longrightarrow$ $\underline{01101001}$

$f(x_1, \dots, x_n) = \underline{x_2 \vee \bar{x}_4 \vee x_7} \equiv$

$\underline{f(x_1, \dots, x_n, y_1, \dots, y_n)} = \underline{x_2 \vee y_4 \vee x_7}$

USE ALGO
FOR MONOTONE
DIST WE HAVE
A NEW ALGO
WITH MISTAKE BOUND
$\leq 2n$

•

Explanation : first we going to take a string say x = 0110 so we going to negate x to generate y 1001
So our combined string become 01101001
Now say we have disjunction like x2 V x4dash so will replace like x2 v y4 (where y4 is negation of x4) in this way we can use current algorithm to find out correct result.

Let's go step by step with **concrete examples** to understand the formula:

$$M(A, c) \leq B(C)$$

where $M(A, c)$ is the number of mistakes made by algorithm A while learning concept c, and $B(C)$ is the upper bound on mistakes for concept class C.

Since every mistake **eliminates** a wrong threshold, the number of mistakes is at most **$\log_2$ of the number of possible thresholds**.
If the robot had **N possible temperature values**, the mistake bound is:

$$B(C) = \log_2 N$$

Thus, even if there are **1,000 possible temperatures**, the robot only makes at most **10 mistakes** (since $\log_2 1000 \approx 10$) before it learns the correct rule.

Example 3: Spam Email Classification

Scenario:

You are training an AI to classify emails as **Spam** (S) or **Not Spam** (N).

- At first, it guesses randomly.
- Each mistake helps it **remove wrong rules** (hypotheses).

Suppose it starts with **16 possible rules** (e.g., checking keywords, sender info, attachments, etc.).
Each mistake eliminates **half** the wrong rules.
After at most **$\log_2(16) = 4$ mistakes**, the AI has **learned the correct rule**.

Thus, the mistake bound is:

$$B(C) = \log_2 16 = 4$$

Even if there were **1,000 possible rules**, the mistake-bound formula ensures that at most:

$$B(C) = \log_2 1000 \approx 10$$

mistakes are needed before **perfect classification**.