CSE 417T
Introduction to Machine Learning

Instructor: Chien-Ju (CJ) Ho

Logistics

- HW 0:
  - http://chienjuho.com/courses/cse417t/hw0.pdf
  - Due by **11am next Tuesday**.
  - Submit via Gradescope.
  - Only waitlisted students need to submit
    - the same question will appear in HW1
  - No late days can be used.
  - The rules on academic integrity apply
    - I'll manually check each submission
    - Suspicions on the violation of academic integrity will be reported to the university. The same rules apply even if not officially enrolled yet
- HW 1: Will be announced by early next week
  - 30% of HW1 will be the content of HW0

Matlab

- Check if you can get access to MATLAB

- Notes:
  - If you are an undergraduate student, follow instructions here. (Portal)
  - If you are an engineering graduate student, you should also have access using instructions above. If not, contact softwarelicensing@wustl.edu.
  - If you are a non-engineering graduate student, check the portal first. If it's not working,
    - Try computer labs or use remote desktop.
    - You can also purchase a student copy here at $53.

Logistics

- Exam Dates
  - Exam 1: March 3 (Tuesday), 2020
  - Exam 2: April 23 (Thursday), 2020

- Each exam covers the content of around half of the semester.
- There will not be a separate final exam.

Logistics: Academic Integrity

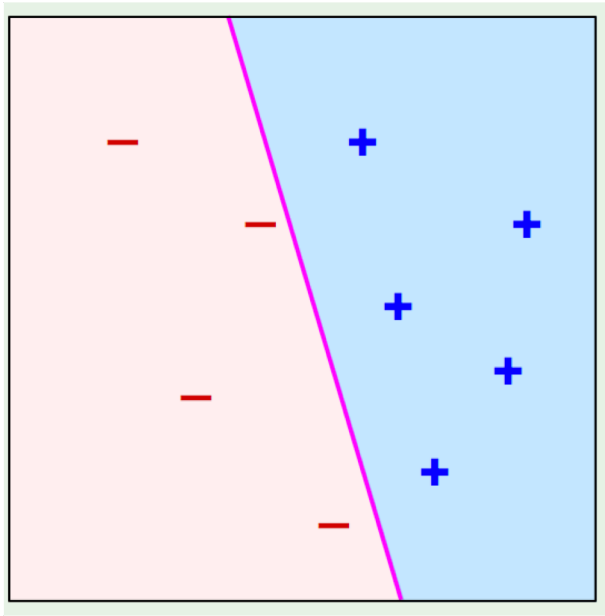
- How to make sure to not violate academic integrity?
- Rule of thumb:
  - You should be able to write down the answers/codes entirely on your own.
  - Can’t look at the write-up / codes by others.

- Ask if you are not sure!

Recap

Linear hypothesis space (Perceptron)

- Input $\vec{x} = (x_1, x_2, \ldots, x_d)$
- Output $y \in \{-1, +1\}$

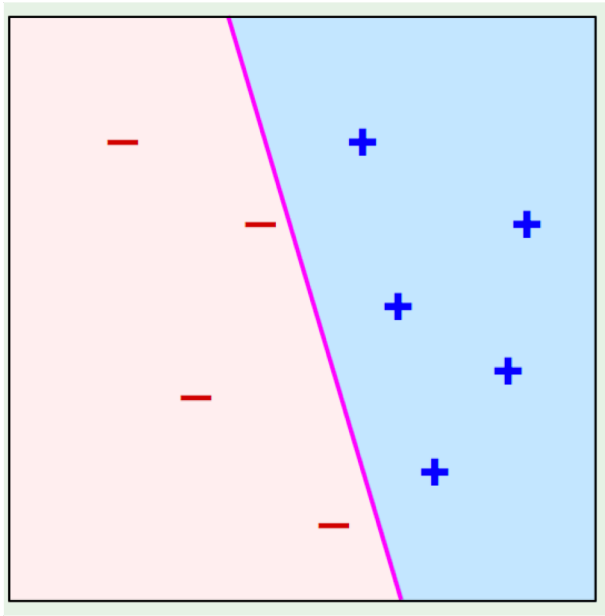


- A hypothesis h is characterized by
  - weight vector $\vec{w} = (w_1, \ldots w_d)$
  - threshold b
  - Linear separator $\vec{w}^T \vec{x} = b$

- $h(\vec{x}) = sign\left(\sum_{i=1}^{d} w_i x_i - b\right) = sign(\vec{w}^T \vec{x} - b)$
  - Predict $+1$ if $\vec{w}^T \vec{x} > b$
  - Predict -1 if $\vec{w}^T \vec{x} < b$

Linear hypothesis space (Perceptron)

- To simplify $h(\vec{x}) = sign(\vec{w}^T\vec{x} - b)$, define
  - $x_0 = 1$
  - $w_0 = -b$



- And we implicitly let
  - $\vec{x} = (\boldsymbol{x_0}, x_1, \ldots, x_d)$
  - $\vec{w} = (\boldsymbol{w_0}, w_1, \ldots, w_d)$

- A hypothesis can then be written as
  - $h(\vec{x}) = sign(\vec{w}^T\vec{x})$
  - We will interchangeably use h and $\vec{w}$ to express a hypothesis in Perceptron

Perceptron Learning Algorithm (PLA)

- Given a dataset $D = \{(\vec{x}_1, y_1), \dots, (\vec{x}_N, y_N)\}$
- Assume the dataset is **linearly separable**

- Perceptron Learning Algorithm
  - Initialize $\vec{w}(0) = \vec{0}$
  - For $t = 0, \dots$
    - Find a misclassified example $(\vec{x}(t), y(t))$ in D
      - That is, $\text{sign}(\vec{w}(t)^T \vec{x}(t)) \neq y(t)$
    - If no such sample exists
      - Return $\vec{w}(t)$
    - Else
      - $\vec{w}(t+1) \leftarrow \vec{w}(t) + y(t)\vec{x}(t)$

Notation:
We use $\vec{w}(t)$ to denote the value of $\vec{w}$ at step t of the algorithm.

Similarly, we use $(\vec{x}(t), y(t))$ to denote the data point found at step t.

Perceptron Learning Algorithm (PLA)

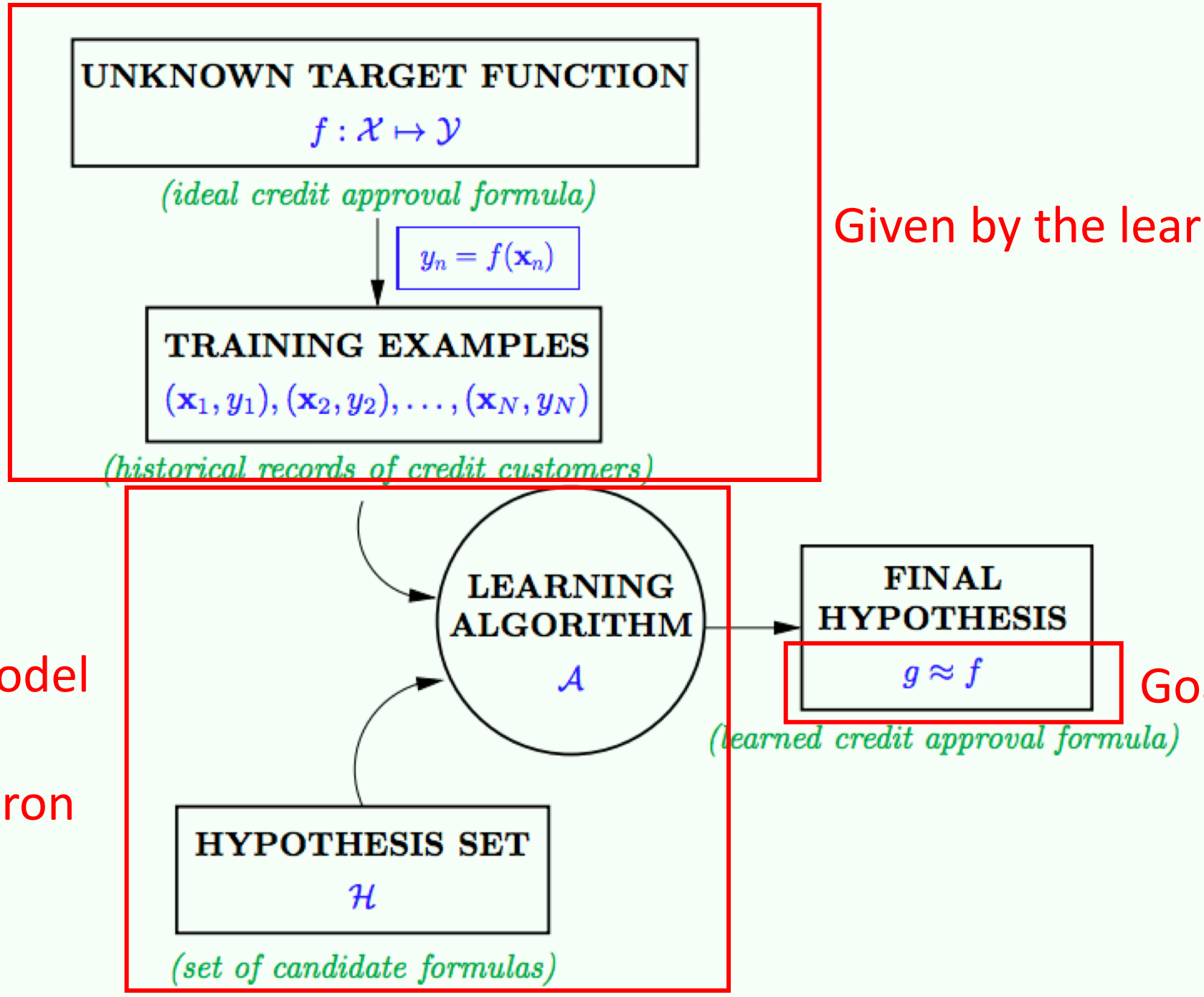
- Theorem (informal):
  - If D is linearly separable, PLA find a linear separator that separates the data in D within a finite number of steps.

- You will need to prove this in the homework

Brief Lecture Notes

The notes are not intended to be comprehensive.

Let me know if you spot errors.



Given by the learning problem

learning model (example:
H: Perceptron
A: PLA)

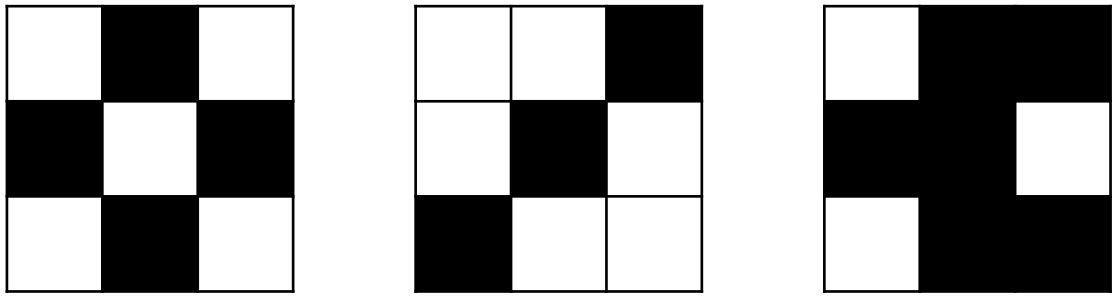
Goal of learning

How Do We Formally Characterize the Goal?

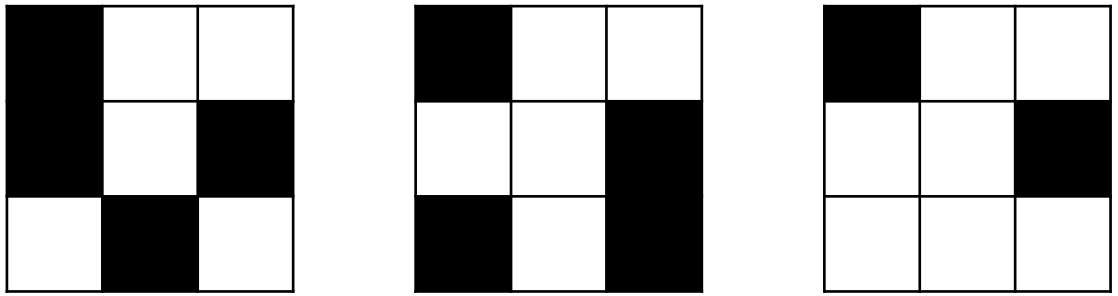
- Goal of learning
  - find $g \approx f$
  - f: unknown target function
  - g: output of the learning algorithm

- Main idea: **Generalization**
  - Want g to make predictions similar to f for **unseen data points**

Training Dataset

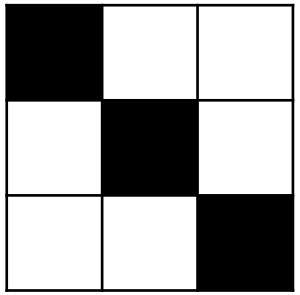


$f(x) = +1$



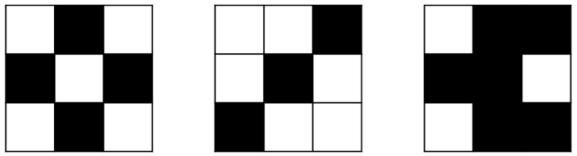
$f(x) = -1$

Predict for unseen points
(Generalization)

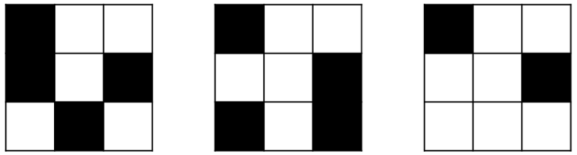


$f(x) = ???$

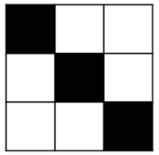
Hypothesis 1



$f(x) = +1$



$f(x) = -1$

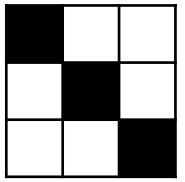


$f(x) = ???$

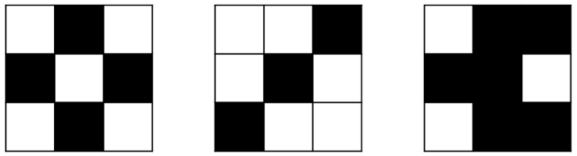
$$h(x) = \begin{cases} +1 \text{ if symmetric} \\ -1 \text{ otherwise} \end{cases}$$

$\downarrow$

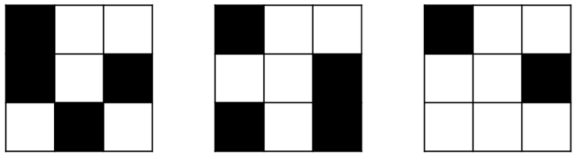
$$h\left(\ \right) = +1$$



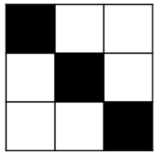
Hypothesis 2



$f(x) = +1$



$f(x) = -1$

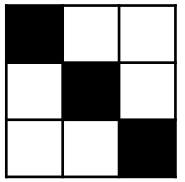


$f(x) = ???$

$$h(x) = \begin{cases} +1 \text{ if top left is white} \\ -1 \text{ otherwise} \end{cases}$$

$\downarrow$

$$h\left(\ \right) = -1$$

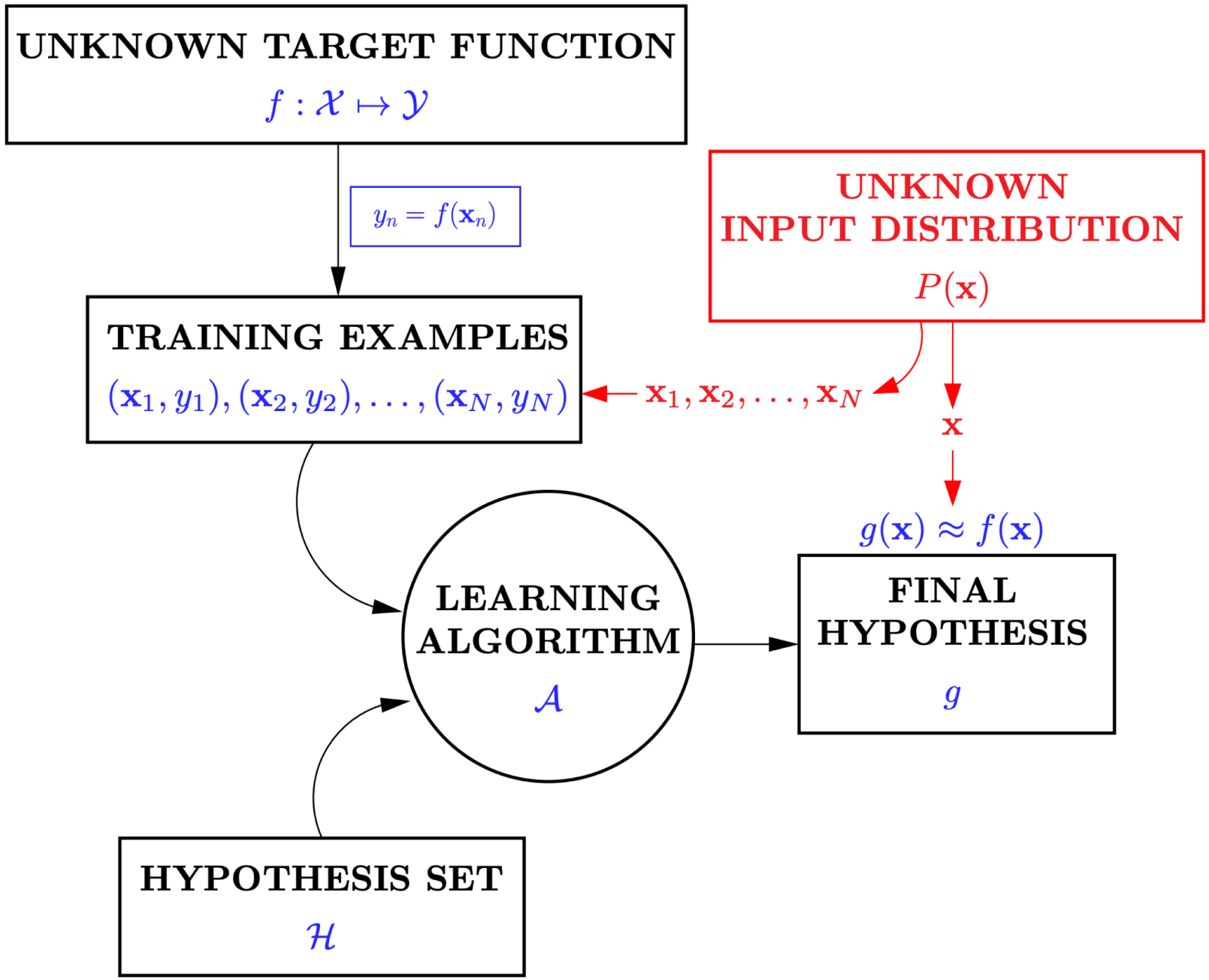


You can come up with many more hypothesis

Feasibility of Learning

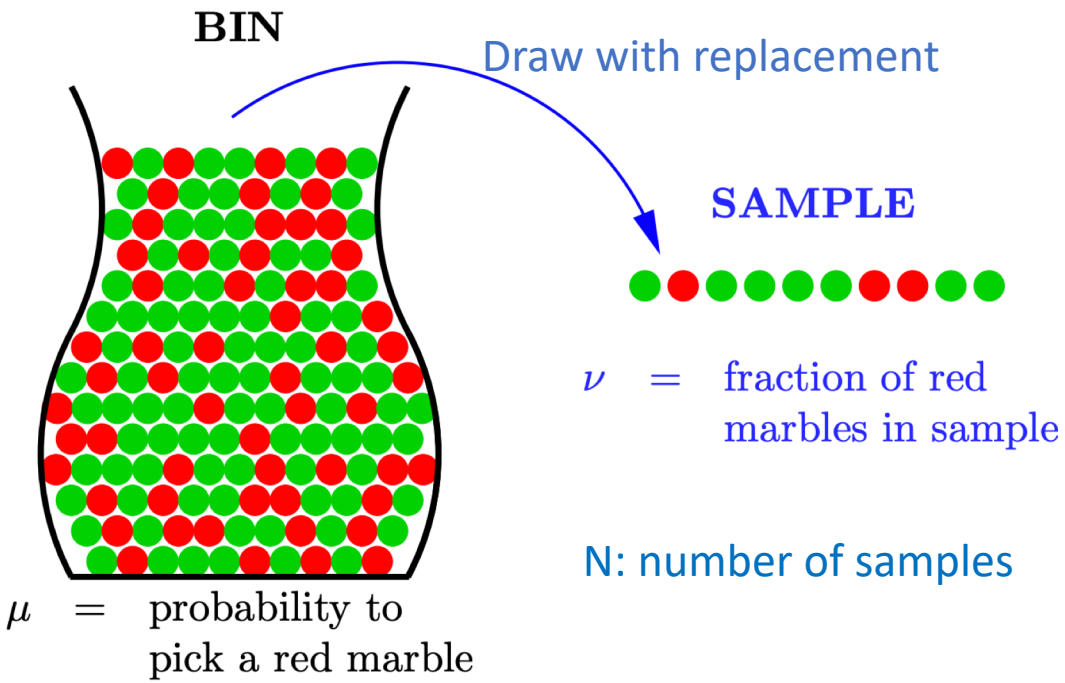
- Is learning feasible?
  - Cannot know anything **for sure** about f outside the data without assumptions
  - We might need to give up the **"for sure"**

- Probability to the rescue:
  - Assume "training" data points and "testing" data points are i.i.d. drawn from some (unknown) distribution
  - Key assumptions of machine learning
    - There have been various discussions on what do do when this assumption is not true, but it's out of scope of this course.


UNKNOWN TARGET FUNCTION
$f : \mathcal{X} \mapsto \mathcal{Y}$
$y_n = f(\mathbf{x}_n)$
UNKNOWN INPUT DISTRIBUTION
$P(\mathbf{x})$
TRAINING EXAMPLES
$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$
$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$
$\mathbf{x}$
$g(\mathbf{x}) \approx f(\mathbf{x})$
LEARNING ALGORITHM
$\mathcal{A}$
FINAL HYPOTHESIS
g
HYPOTHESIS SET
$\mathcal{H}$

Let’s discuss probability first

A Thought Experiment about Probability



What can we say about μ from ν?

Law of large numbers

- When $N \to \infty$, $\nu \to \mu$

Hoeffding's Inequality

- $\Pr[|\mu - \nu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$ for any $\epsilon > 0$

Interpretations

$$\Pr[|\mu - \nu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

Define $\delta = \Pr[|\mu - \nu| > \epsilon]$: Probably of "bad events"

- Fix $\epsilon, \delta = O(e^{-N})$; Fix $N, \delta = O(e^{-\epsilon^2})$; Fix $\delta, \epsilon = O(\sqrt{\frac{1}{N}})$
- N=1000
  - $\mu - 0.05 \leq \nu \leq \mu + 0.05$ with 99% chance
  - $\mu - 0.10 \leq \nu \leq \mu - 0.10$ with 99.9999996% chance

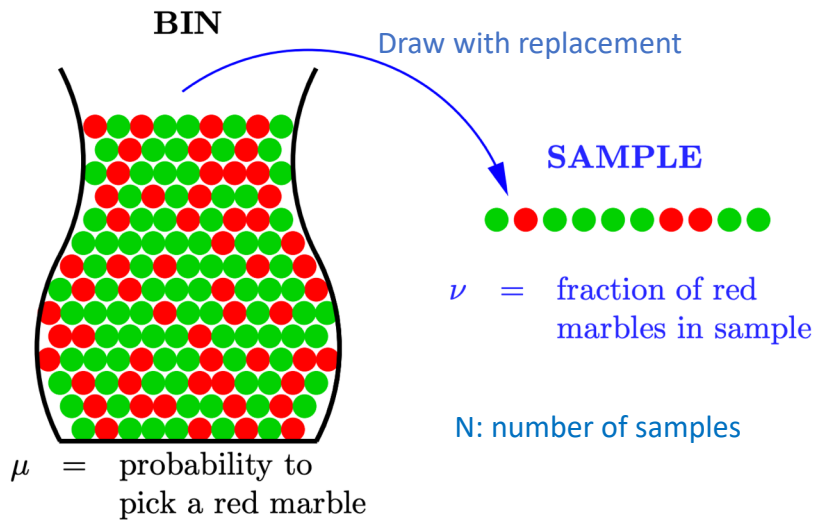
- ν is approximately close to μ with high probability
- ν as an estimate of μ is **p**robably **a**pproximately **c**orrect (P.A.C.)

PAC learning is proposed by Leslie Valiant, who wins the Turing award in 2010.

Connection to Learning

- Let each marble represent a point $\vec{x}$, drawn from unknown $P(\vec{x})$
  - Dataset $D = \{(\vec{x}_1, y_1), \dots, (\vec{x}_N, y_N)\}$
  - Recall that $y_n = f(\vec{x}_n)$ (will discuss noisy target function f later in the semester)

- "Fix" a hypothesis h
  - For each marble $\vec{x}$, color it as below
    - If $h(\vec{x}) = f(\vec{x})$, color it as green marble
    - If $h(\vec{x}) \neq f(\vec{x})$, color it as red marble
  - With the above coloring
    - $\mu = \Pr_{\vec{x} \sim P(\vec{x})}[h(\vec{x}) \neq f(\vec{x})]$
      $\stackrel{\text{def}}{=} E_{out}(h)$ [Out-of-sample error of h]
    - $\nu = \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}[h(\vec{x}_n) \neq f(\vec{x}_n)]$
      $\stackrel{\text{def}}{=} E_{in}(h)$ [in-sample error of h]



Connection to Learning

- Look at the error again
  - $E_{out}(h)$: What we really care about but unknown to us
  - $E_{in}(h)$: What we can calculate from dataset D

- Fixed a h, What can we say about $E_{out}(h)$ from $E_{in}(h)$?

Hoeffding's Inequality

$$\Pr[|E_{out}(h) - E_{in}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad \text{for any } \epsilon > 0$$

Are we done?

Not really…..

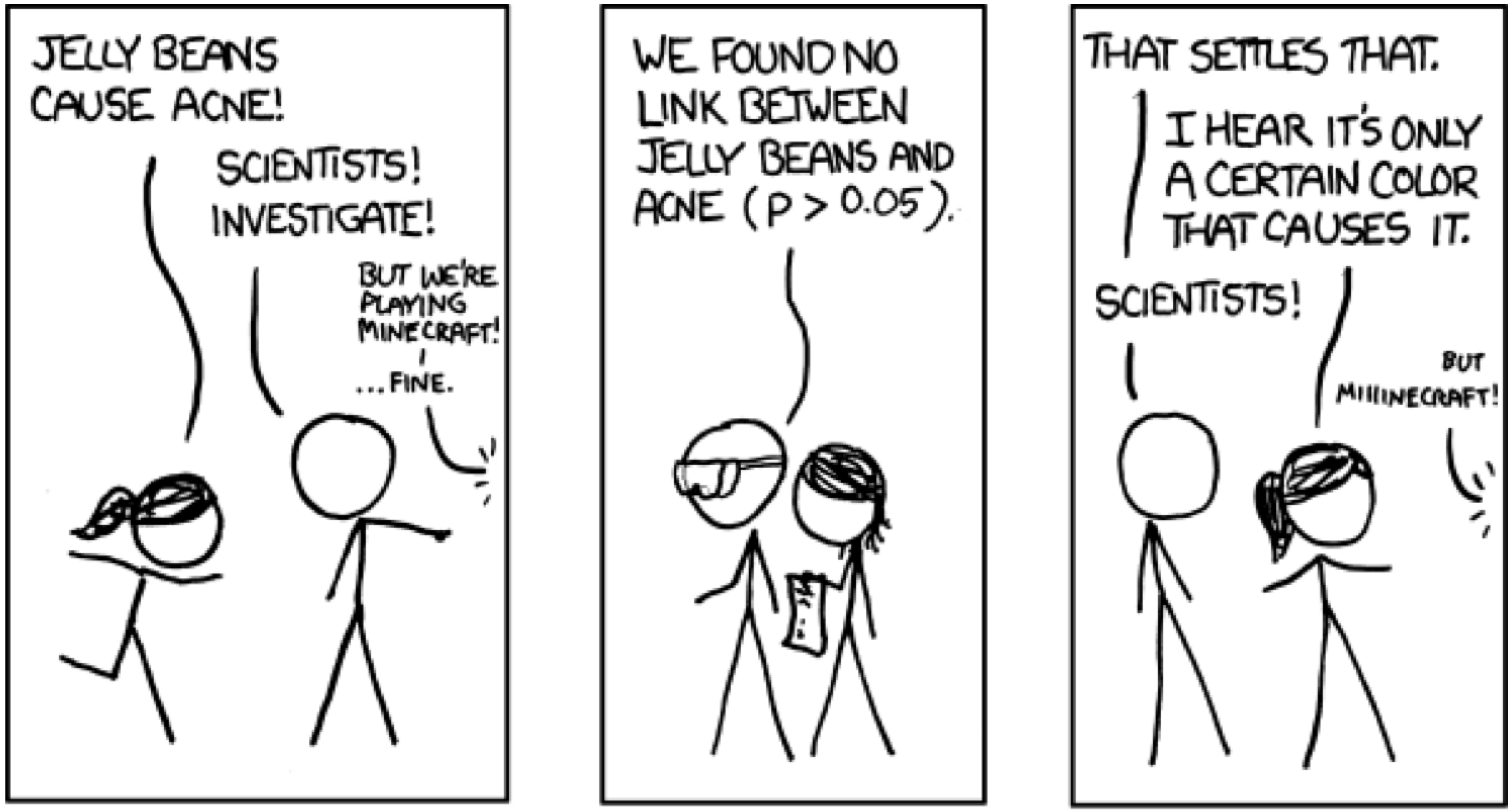
Connection to "Real" Learning

- Given a finite hypothesis set $H = \{h_1, \ldots, h_M\}$
- Apply some learning algorithm on D, output a $g \in H$
  - For example, choosing the hypothesis that minimizes in-sample error
    - $g = argmin_{h \in H} E_{in}(h)$

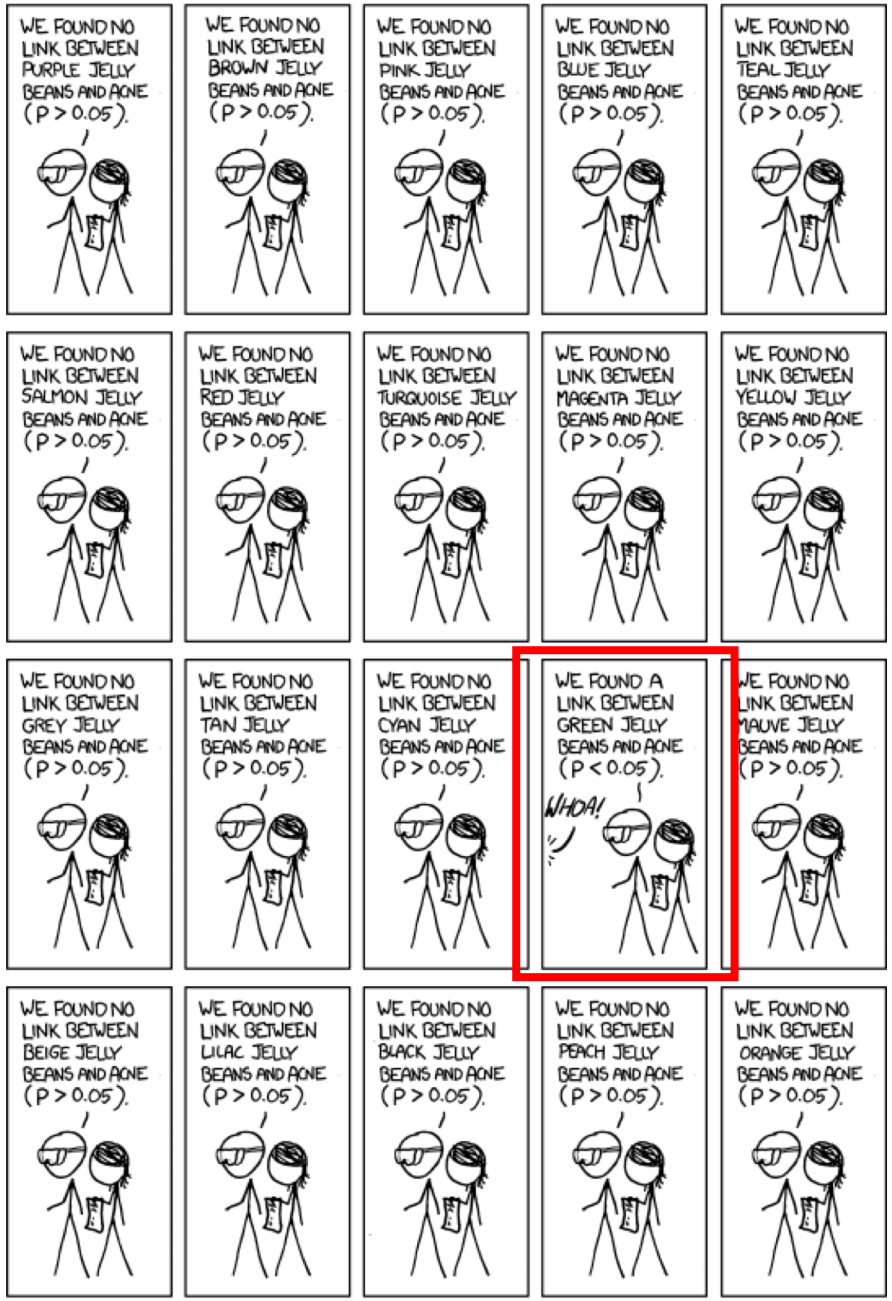
- Can we apply Hoeffding's inequality and claim

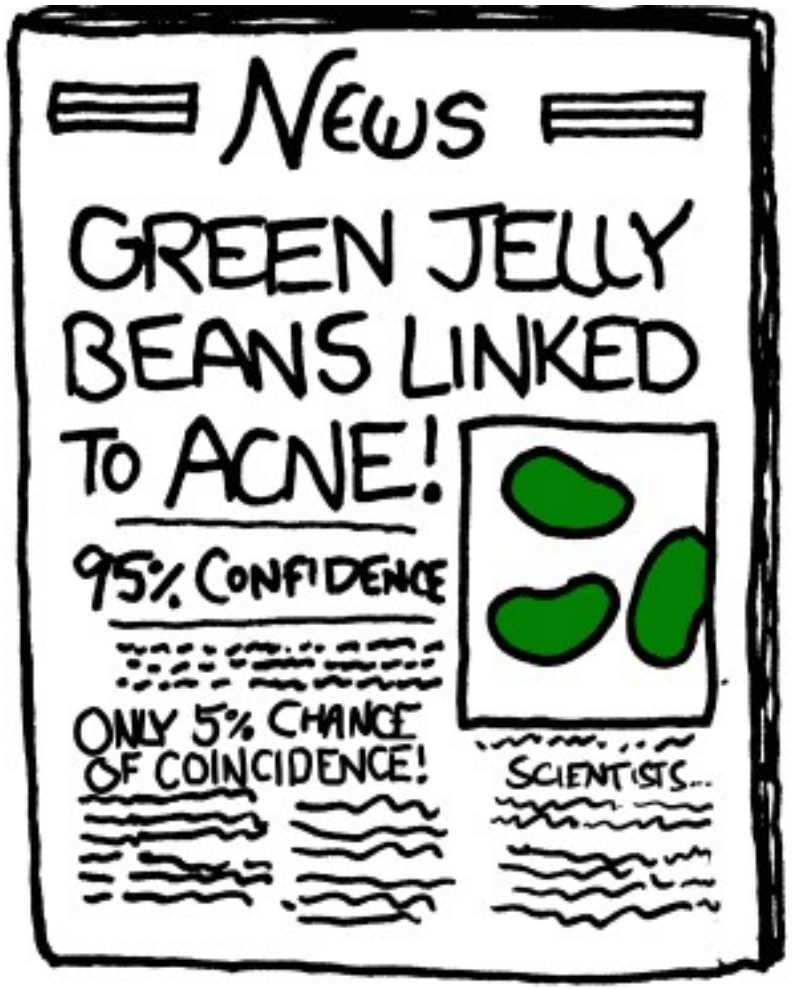
$$\Pr[|E_{out}(g) - E_{in}(g)| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad \text{for any } \epsilon > 0$$

- **No!**



From xkcd, by Randall Munroe: http://xkcd.com/882





From xkcd, by Randall Munroe: http://xkcd.com/882

Another Analogy

- If you toss a fair coin 10 times, the probability that it comes up heads 10 times is $2^{-10} \approx$ 1e-3
- If you toss 1000 fair coins 10 times each, the probability that at least one coin comes up heads 10 times is

$$1 - \left(\frac{1023}{1025}\right)^{1000} \approx 62.36\%$$

- If each hypothesis is doing random guessing (i.e., tossing a fair coin), if we have 1000 hypothesis with 10 data points, more than 60% chance there will be at least one hypothesis with **zero in-sample error**
  - But that hypothesis is still random guessing and has 50% out-of-sample error