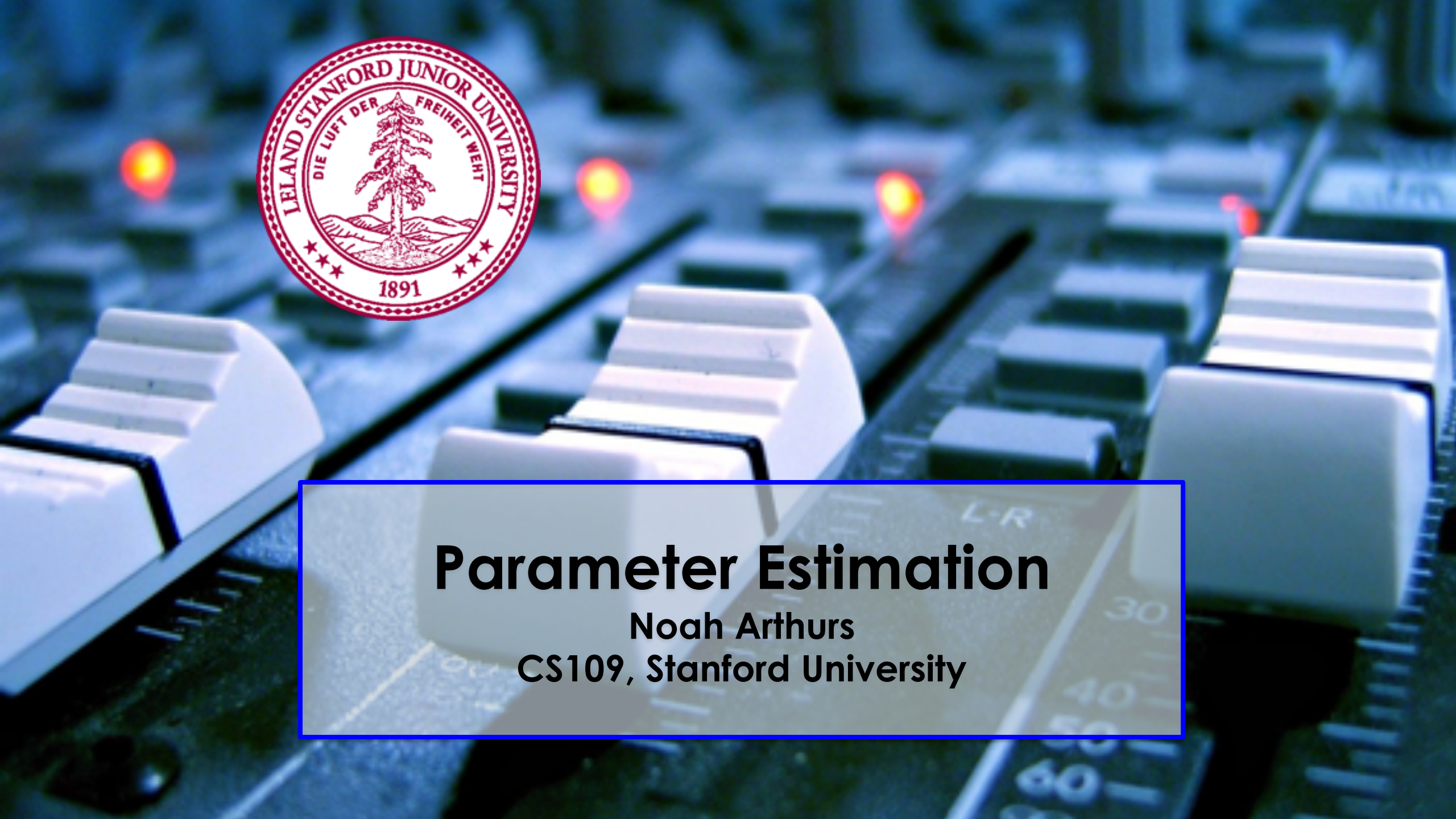


Parameter Estimation

Noah Arthurs
CS109, Stanford University

Review

Bootstrapping for p values

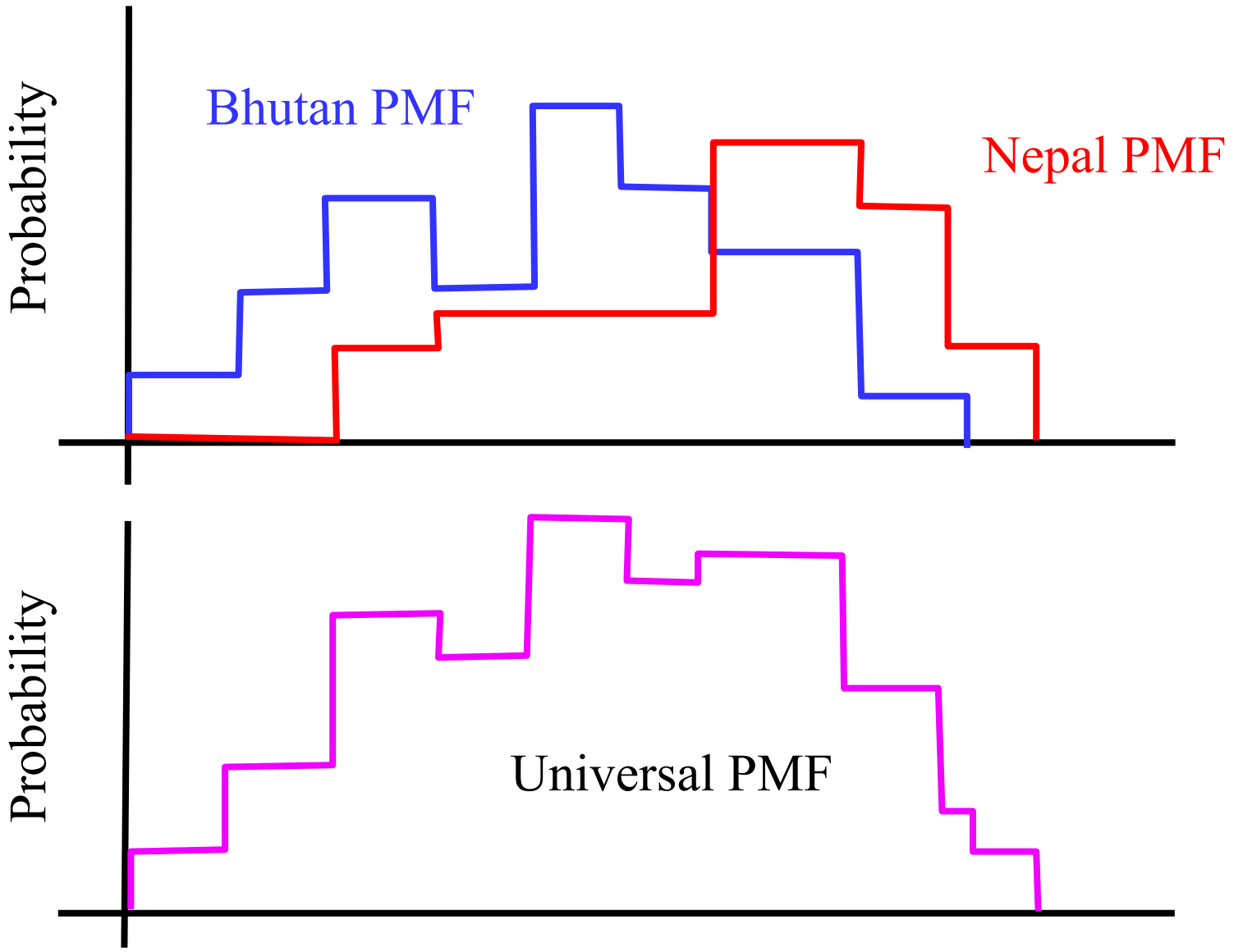
Null Hypothesis Test

Nepal Happiness	Bhutan Happiness
4.44	2.15
3.36	3.01
5.87	2.02
2.31	1.43
...	...
3.70	1.83
$\mu_1 = 3.1$	$\mu_2 = 2.4$

Claim: The difference in happiness between Nepal and Bhutan is 0.7 happiness points.

Null hypothesis: even if **there is no pattern** (ie the two samples are identically distributed) your claim might have arisen by **chance**.

Universal Sample



Bootstrap for P Values

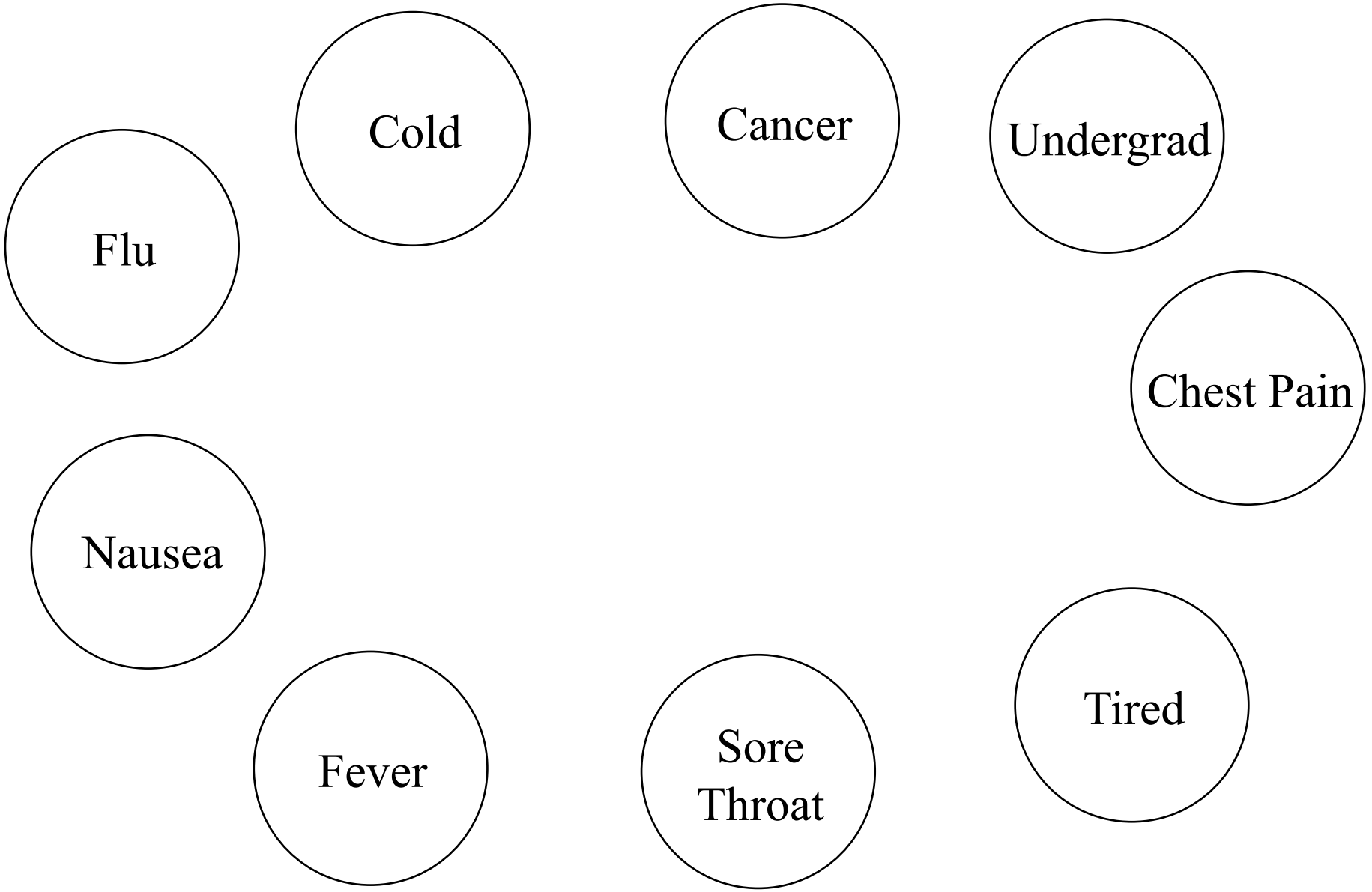
```
def pvalueBootstrap(bhutanSample,nepalSample):
   N = size of the bhutanSample
   M = size of the nepalSample

   uniSamples = combine bhutanSamples and nepalSamples
   count = 0

   repeat 10,000 times:
      bhutanResample = draw N resamples from the uniSamples
      nepalResample = draw M resamples from the uniSamples
      muBhutan = sample mean of the bhutanResample
      muNepal = sample mean of the nepalResample
      meanDiff = |muNepal - muBhutan|
      if meanDiff > observedDifference:
         count += 1

   pValue = count / 10,000
```

Null Hypothesis Test

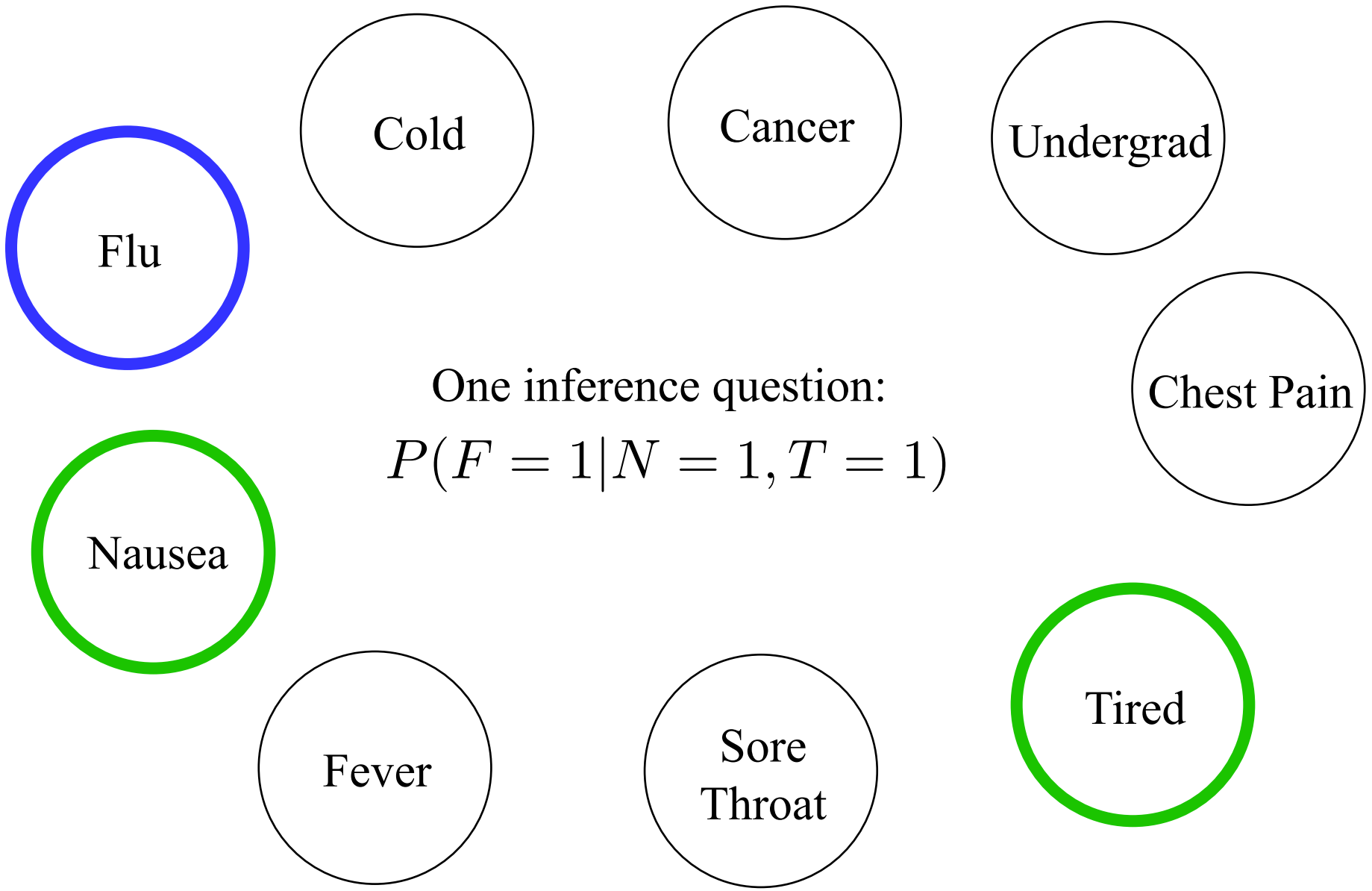
Nepal Happiness	Bhutan Happiness
4.44	2.15
3.36	3.01
5.87	2.02
2.31	1.43
...	...
3.70	1.83
$\mu_1 = 3.1$	$\mu_2 = 2.4$

Claim: The difference in happiness between Nepal and Bhutan is 0.7 happiness points ($p < 0.008$).

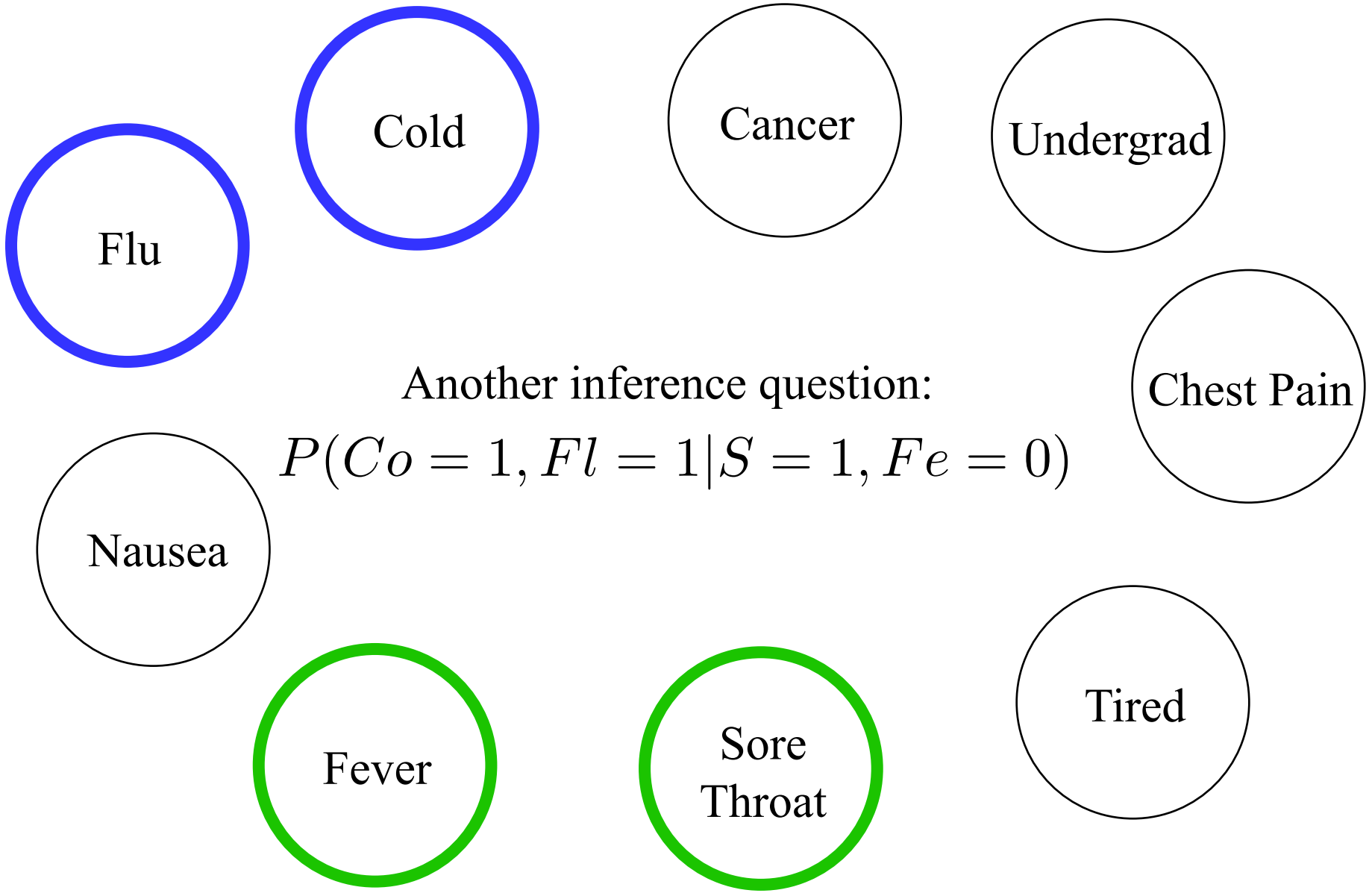
General "Inference"



General "Inference"



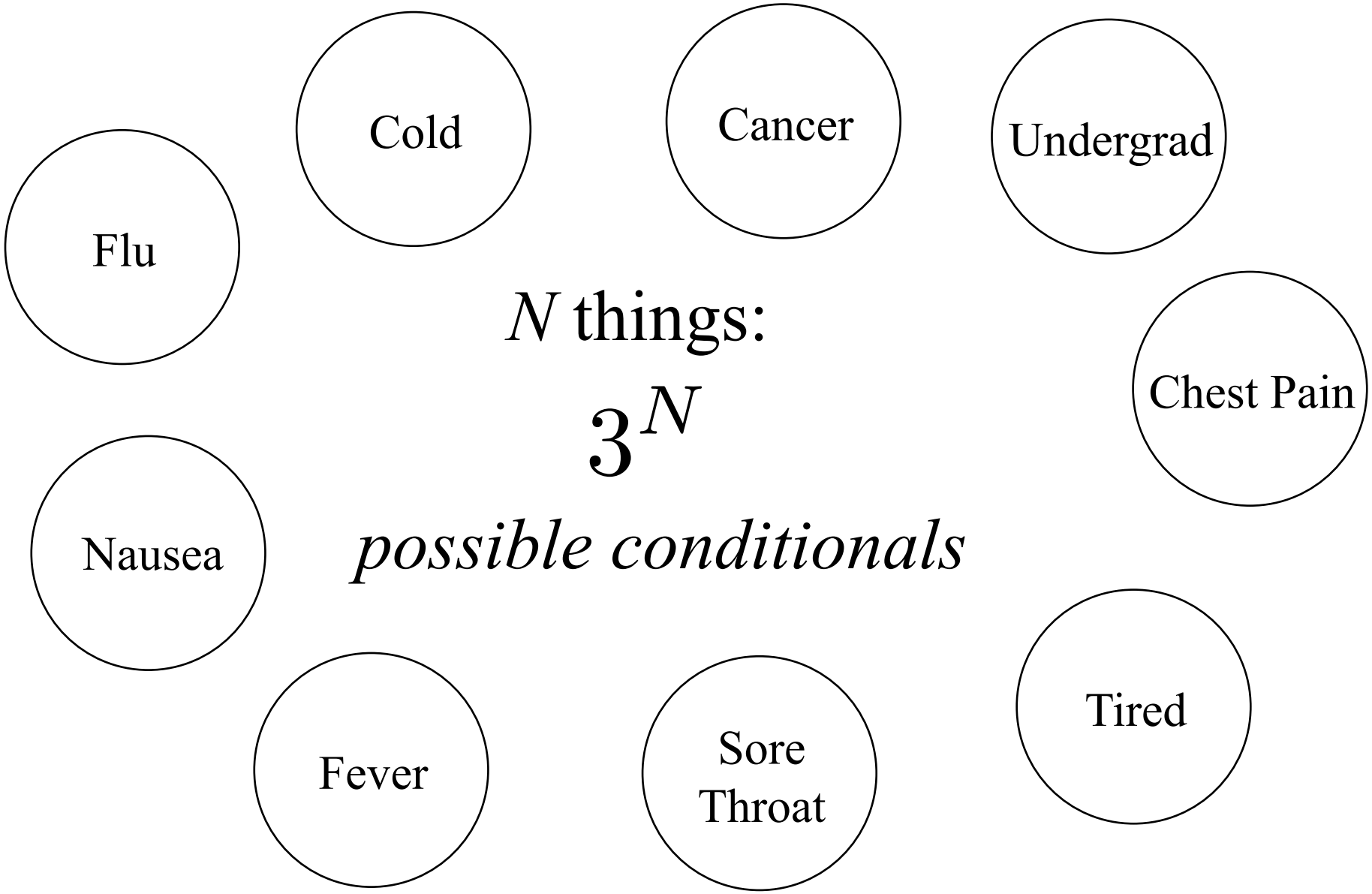
General "Inference"



Another inference question:

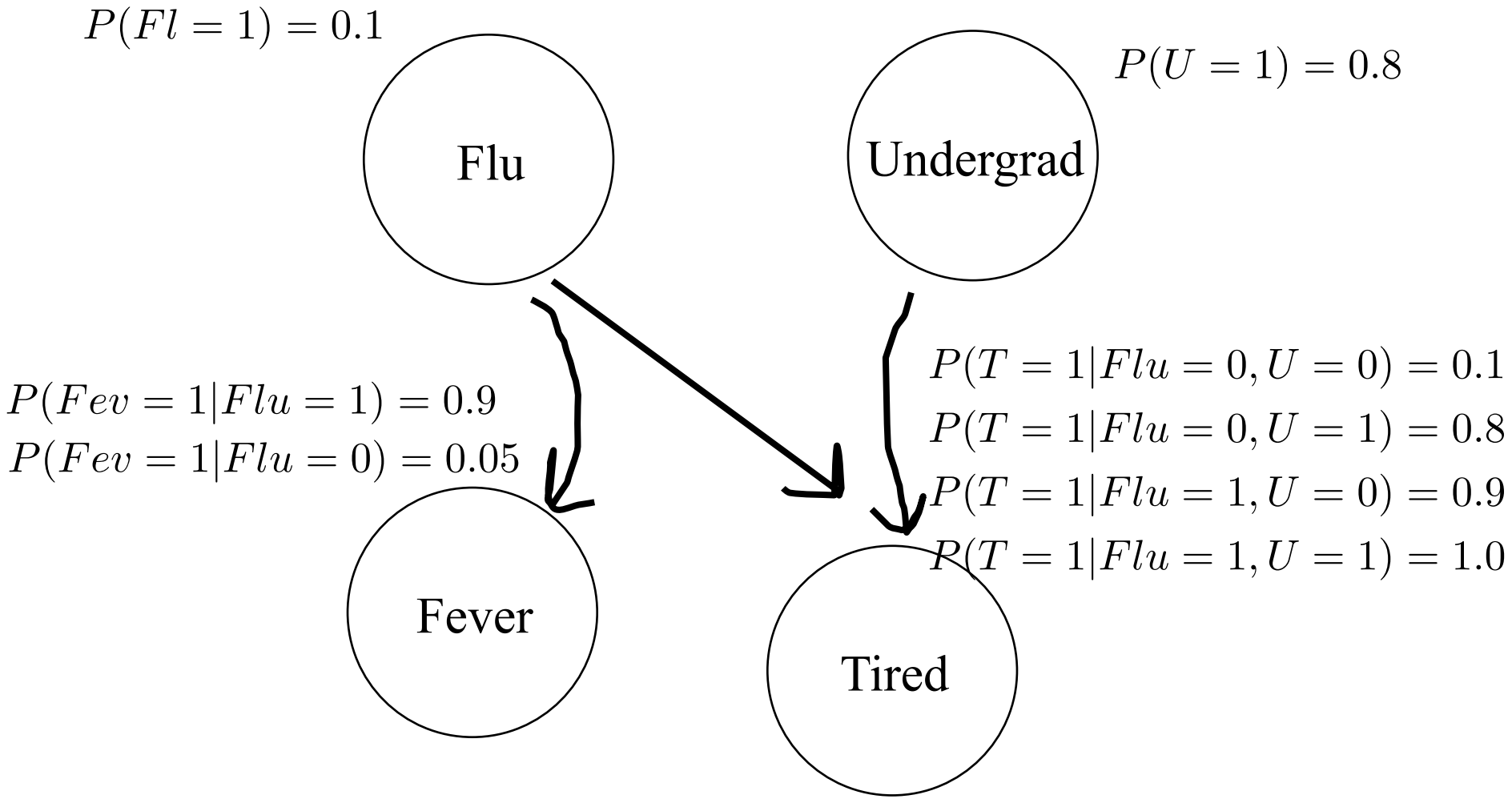
$$P(Co = 1, Fl = 1 | S = 1, Fe = 0)$$

How Many Things Can You Condition On?



Bayesian Network

Describe the joint using causality!



$$P(Fl = a, Fe = b, U = c, T = d)?$$

Bayesian Network:

If you know causality,

Make a network of assumed direct causality for your random vars.

You simply need to give:

$P(\text{values} \mid \text{parents})$

for each random variable.

Prob can be a conditional probability table or an equation!

Alg #1: Joint Sampling

```
3  N_SAMPLES = 100000
4
5  # Program: Joint Sample
6  # ----------------------
7  # we can answer any pro
8  # with multivariate sam
9  # where conditioned var
10 def main():
11     obs = getObservatio
12     print 'Observation
13
14     samples = sampleATo
15     prob = probFluGiven
16     print 'Pr(Flu) = ',
```

```
webMd — -bash — 30×20
[0, 1, 0, 1]
[1, 1, 1, 1]
[0, 1, 0, 1]
[0, 1, 0, 0]
[0, 1, 0, 0]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 0, 0, 0]
[0, 0, 0, 0]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 0]
[0, 0, 0, 0]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[1, 1, 0, 1]
```

webMd — -bash — 39×20

```
[0, 1, 1, 0]
[1, 0, 1, 1]
[0, 1, 0, 1]
[0, 1, 0, 0]
[0, 1, 0, 0]
[0, 1, 1, 0]
[1, 1, 1, 1]
[0, 1, 0, 0]
[0, 0, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 1]
[0, 1, 0, 0]
[0, 1, 0, 1]
[0, 1, 0, 0]
[0, 0, 0, 0]
[0, 0, 0, 1]
Observation =  [None, None, None, None]
Pr(Flu | Obs) =  0.10164
>
```

If you can sample enough from the joint distribution, you can answer any probability question

Each one of these is one joint sample:
[Flu, Undergrad, Fever, Tired]

BAE's Theorem?

$$P(A \mid B\ E) = \frac{P(B \mid A\ E)\ P(A \mid E)}{P(B \mid E)}$$

$P(F = 1 \mid \text{all other rvs})$

Know: P(symptom | flu, undergrad) P(flu) P(undergrad)

Flu is independent of undergrad

Tired and fever are conditionally independent given flu, undergrad

---

$$P(F = 1|\text{symptoms}, U = u)$$

$$= \frac{P(\text{symptoms}|F = 1, U = u)P(F = 1|U = u)}{P(\text{symptoms}|U = u)}$$

$$\propto P(\text{symptoms}|F = 1, U = u)P(F = 1|U = u)$$

$$\propto P(F = 1)P(\text{symptoms}|F = 1, U = u)$$

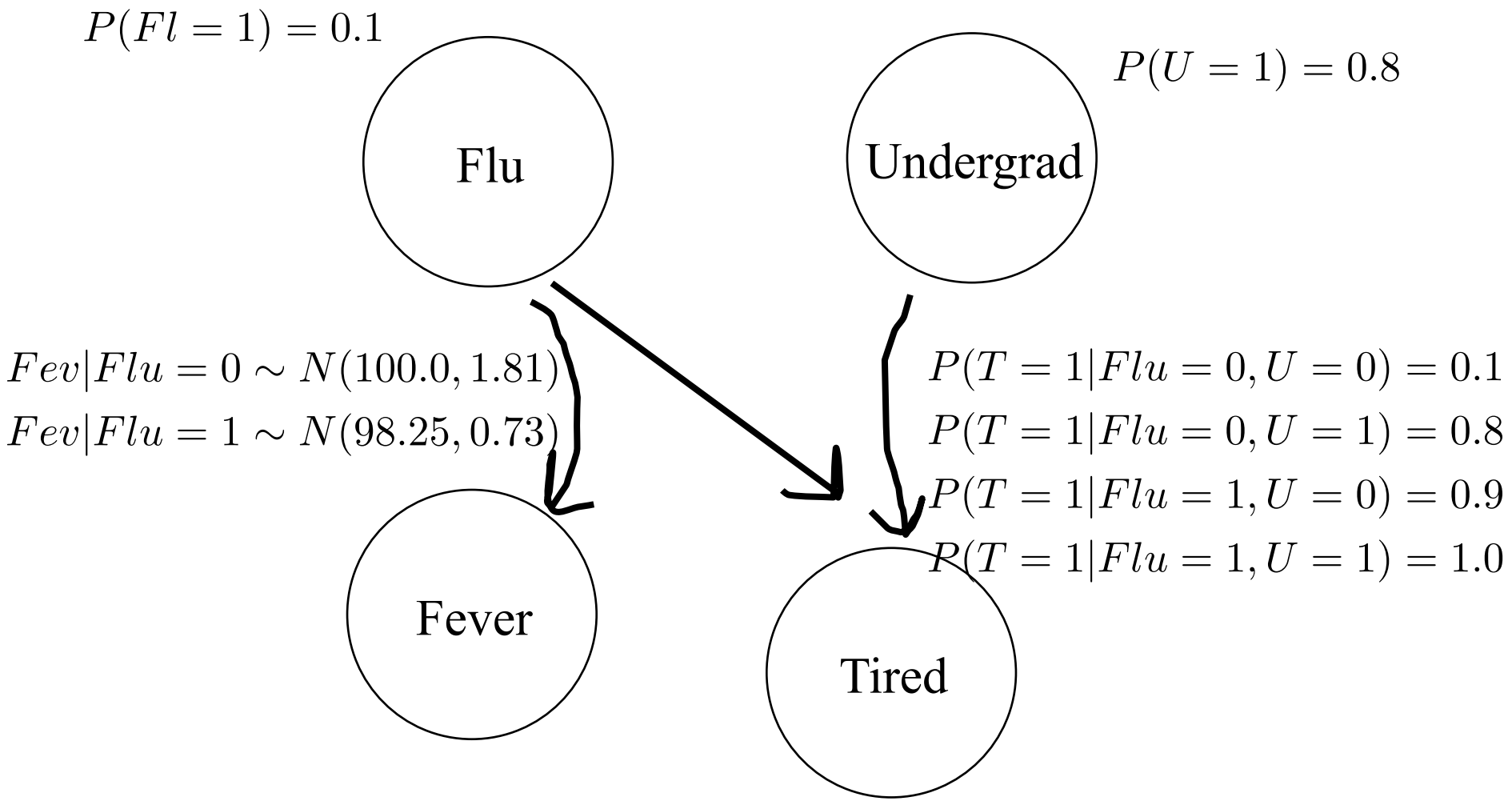
$$\propto P(F = 1)\prod_i P(\text{symptom}_i|F = 1, U = u)$$

General Inference Summary

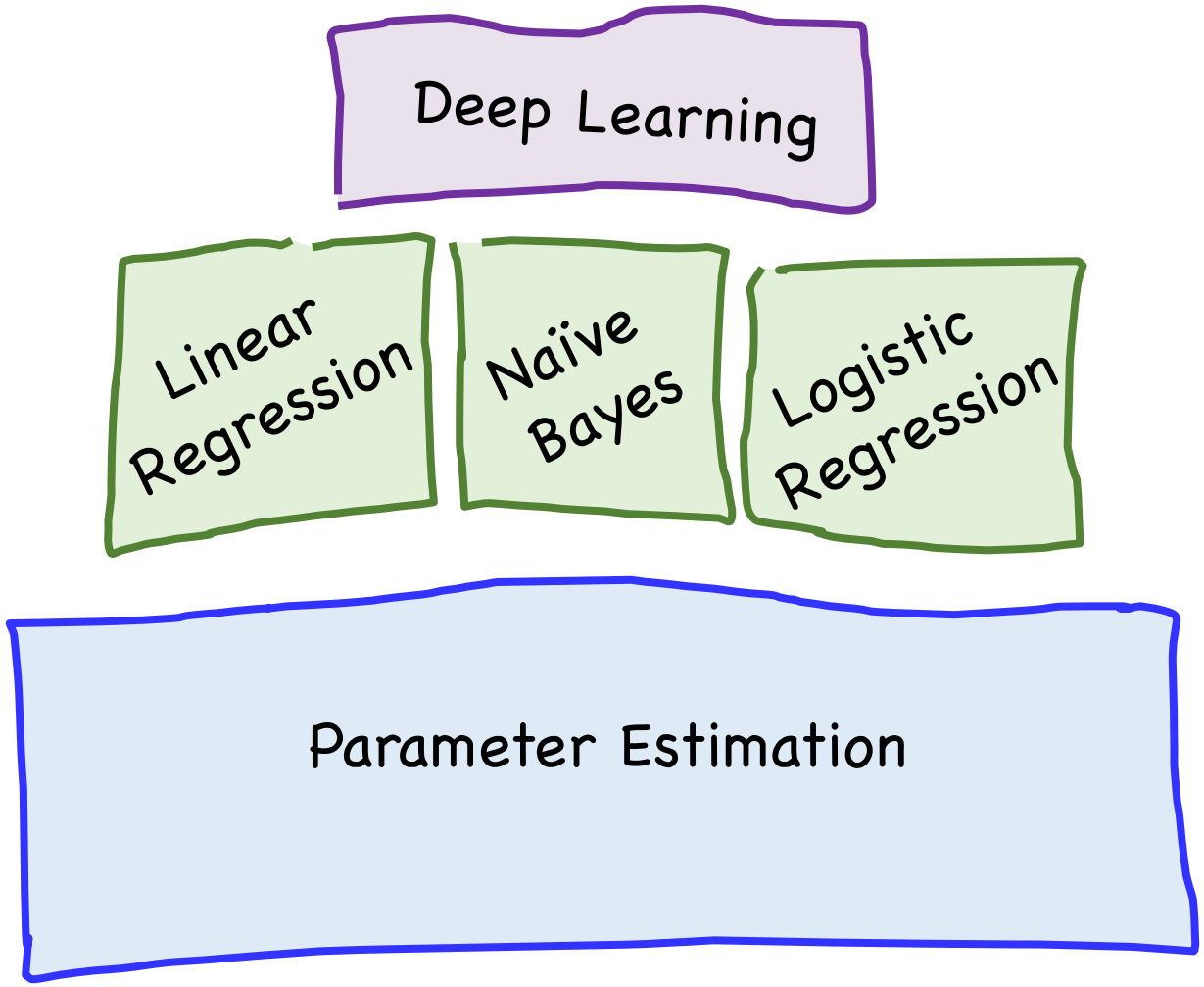
- **Straight Math** is fast, but can be prohibitively hard for complex models (see hw).
- **Joint Sampling** is really easy to program but fails for continuous variables (and when what you are conditioning on is rare)
- **Take CS228** to learn about many, many more algorithms!

MACHINE LEARNING

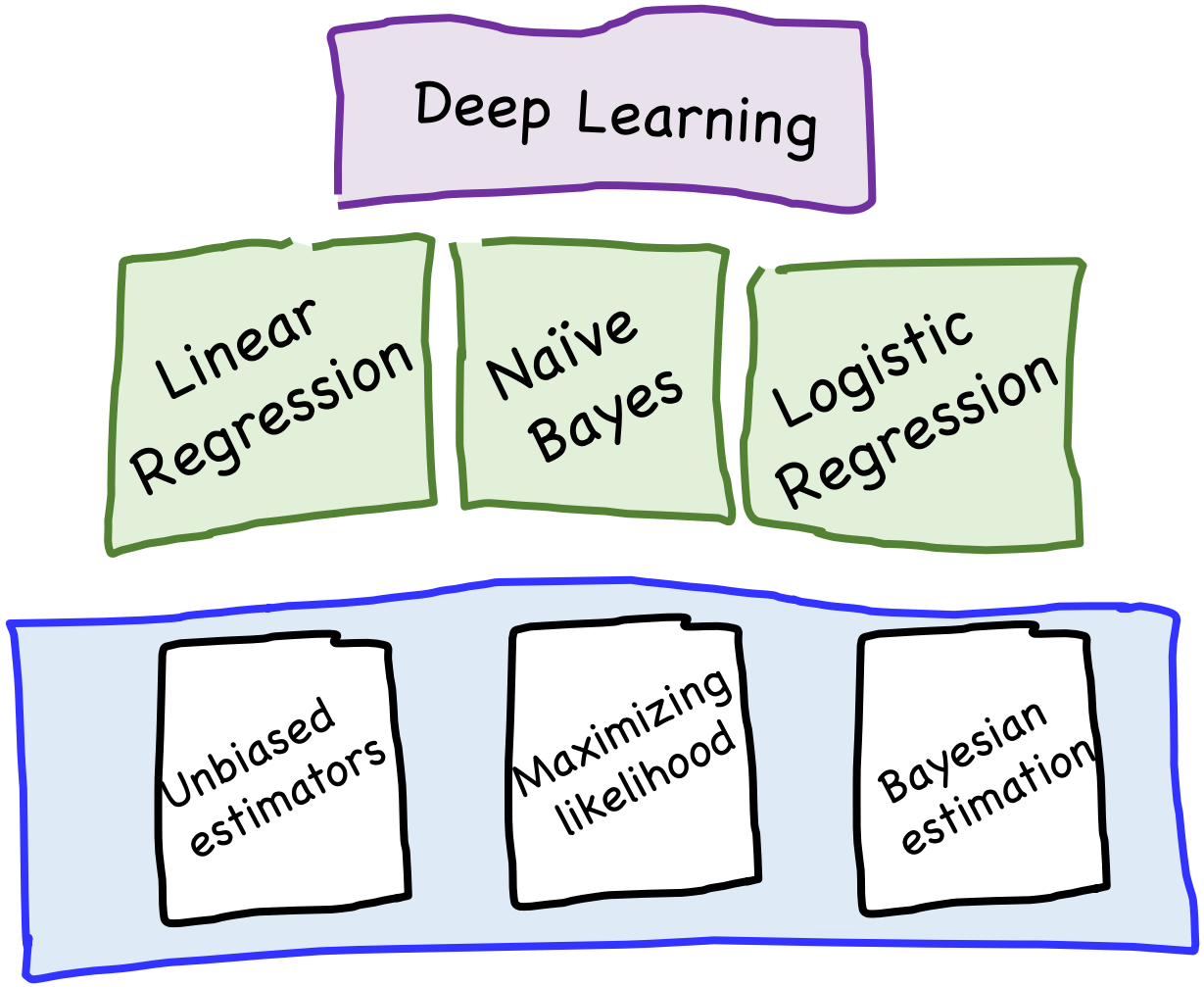
Where Do The Numbers Come From?



Our Path


Deep Learning
Linear Regression
Naïve Bayes
Logistic Regression
Parameter Estimation

Our Path


Deep Learning
Linear Regression
Naïve Bayes
Logistic Regression
Unbiased estimators
Maximizing likelihood
Bayesian estimation

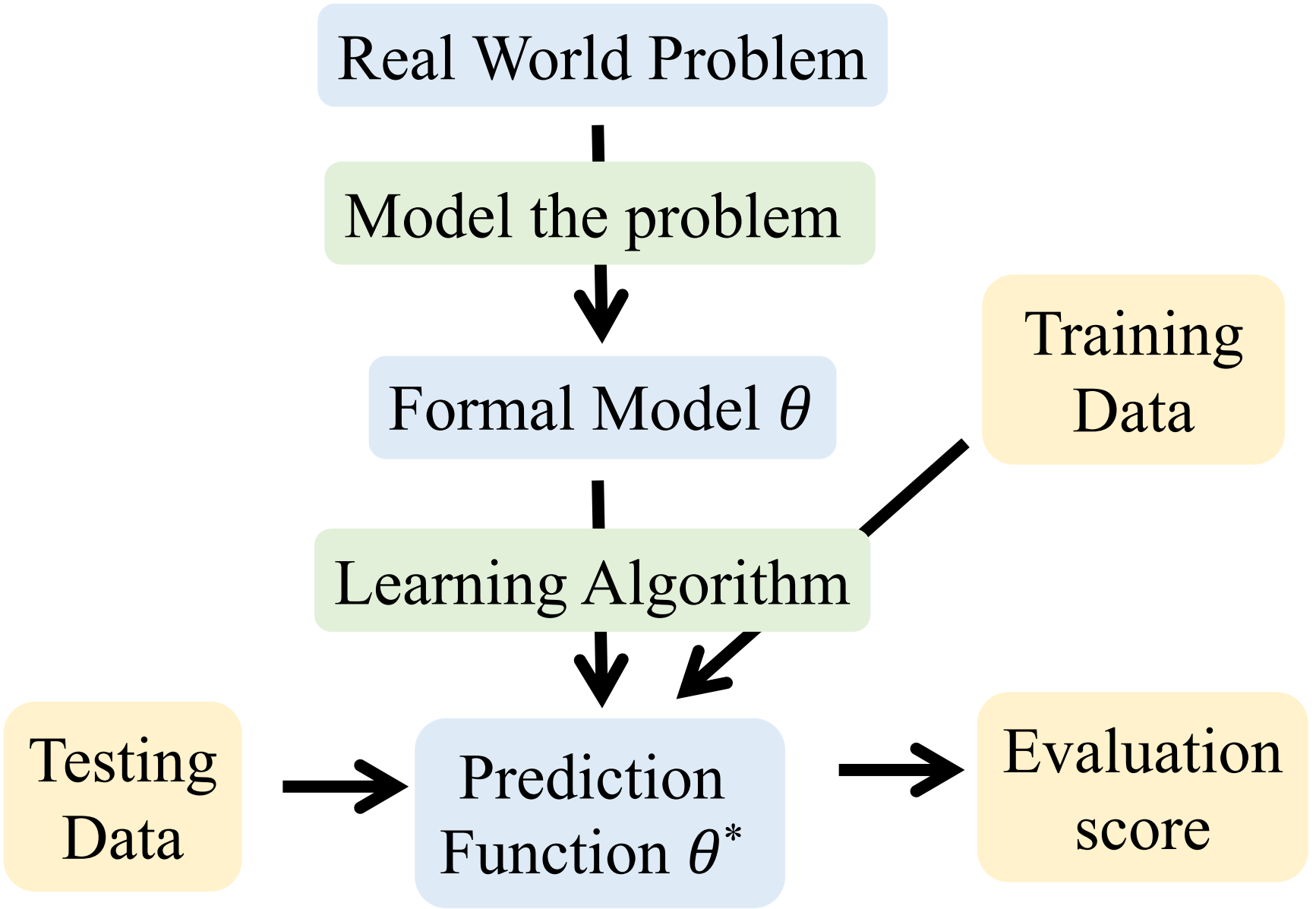
What are Parameters?

- Consider some probability distributions:
  - Ber(p) $\quad \theta = p$
  - Poi(λ) $\quad \theta = \lambda$
  - Uni(α, β) $\quad \theta = (\alpha, \beta)$
  - Normal(μ, σ^2) $\quad \theta = (\mu, \sigma^2)$
  - $Y = mX + b$ $\quad \theta = (m, b)$
  - etc…
- Call these “parametric models”
- Given model, **parameters** yield actual distribution
  - Usually refer to parameters of distribution as θ
  - Note that θ that can be a vector of parameters

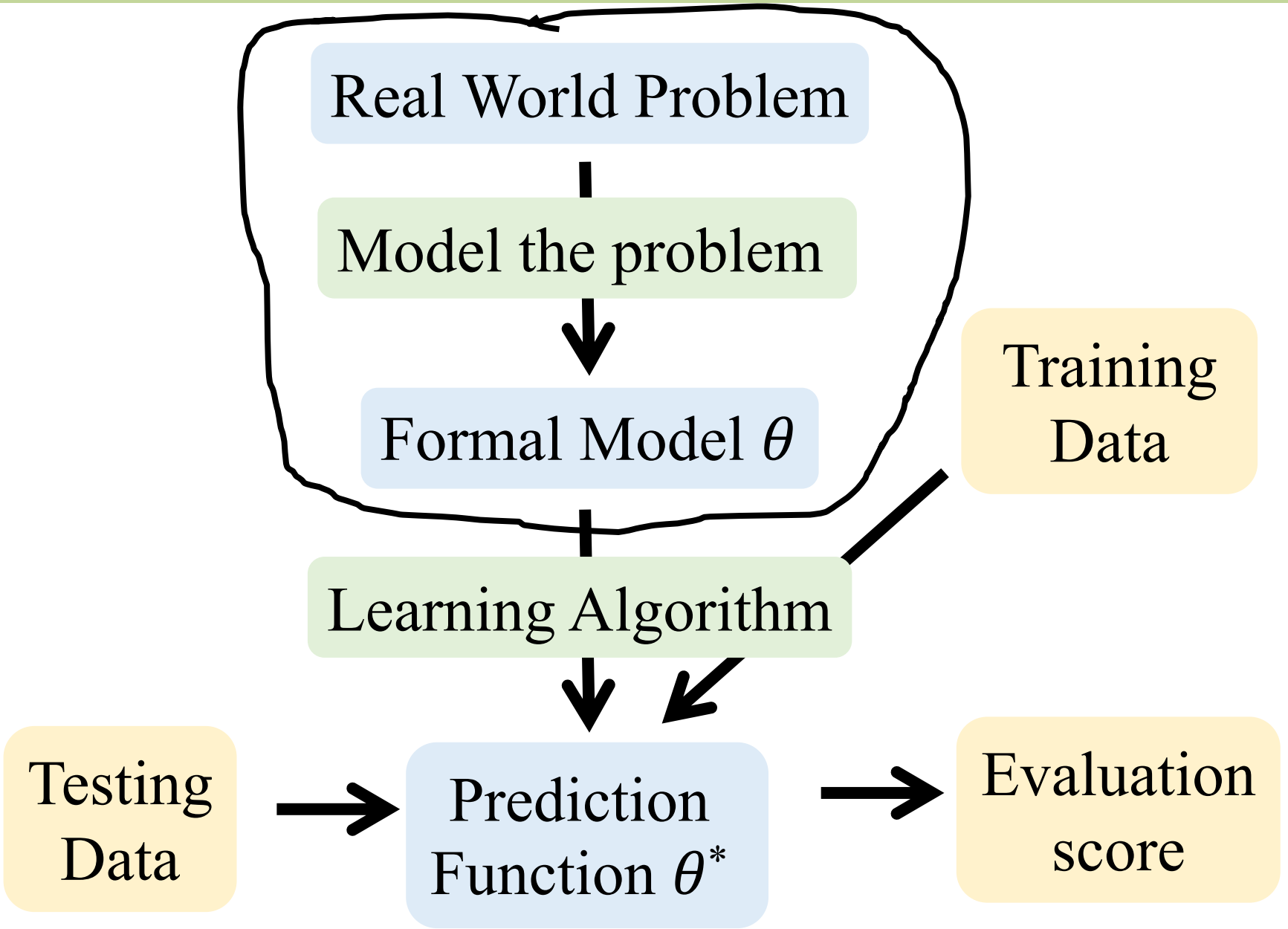
Why Do We Care?

- In real world, don't know "true" parameters
  - But, **we do get to observe data**
    - E.g., number of times coin comes up heads, lifetimes of disk drives produced, number of visitors to web site per day, etc.
  - Need to estimate model parameters from data
  - "Estimator" is random variable estimating parameter
- Estimate of parameters allows:
  - Better understanding of process producing data
  - Future **predictions** based on model
  - Simulation of processes

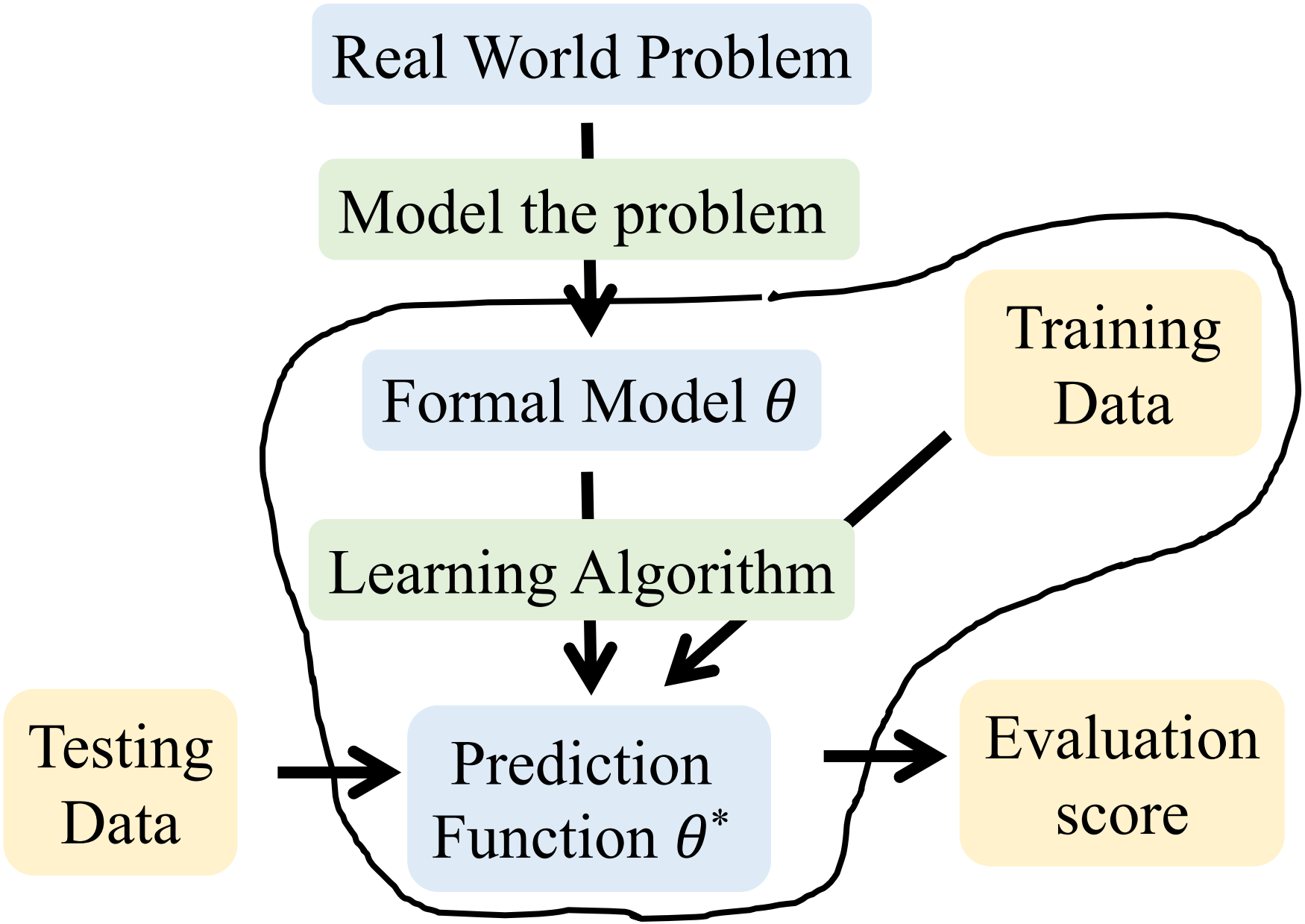
Supervised Learning



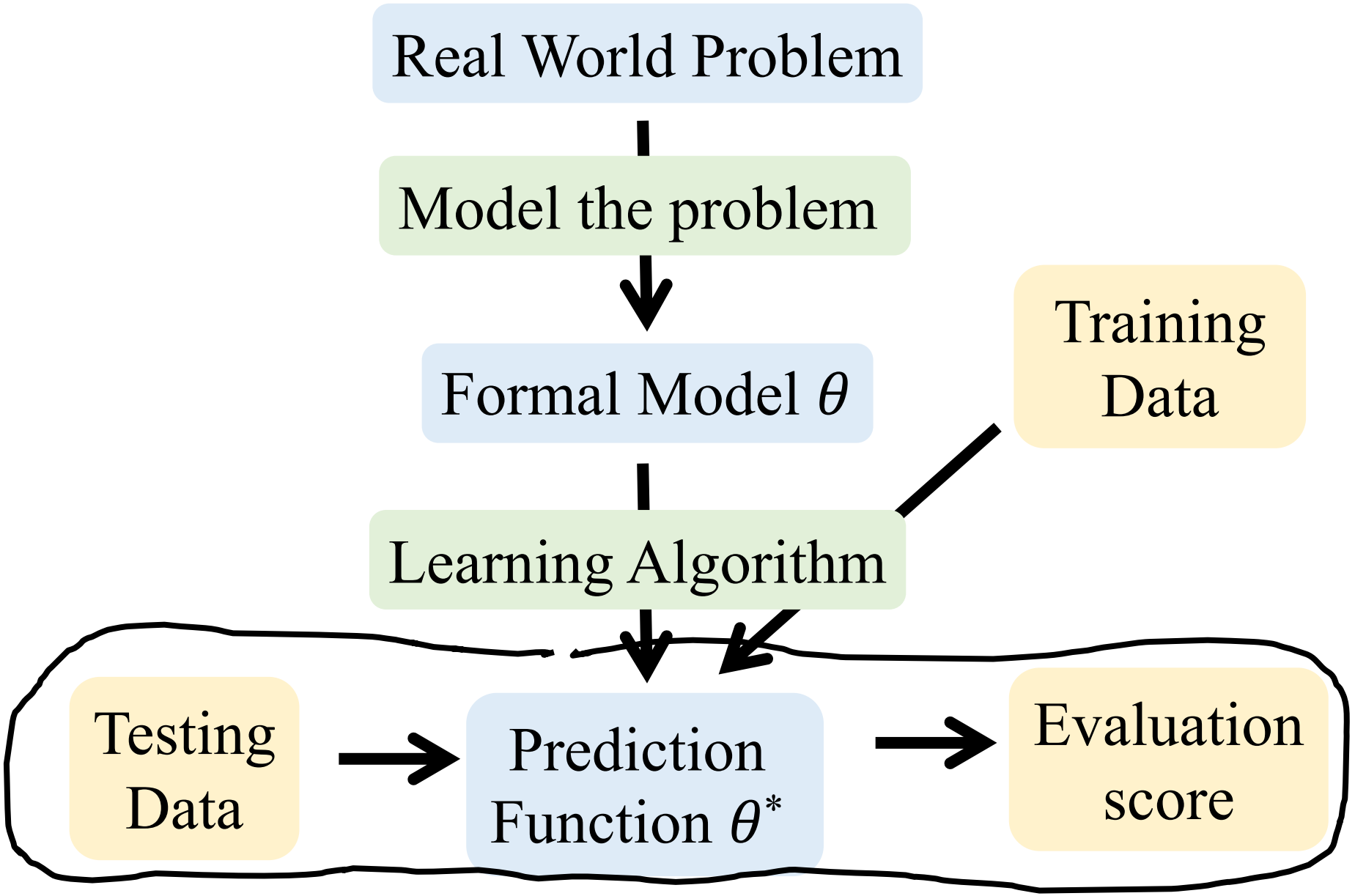
Modelling


Real World Problem
Model the problem
Formal Model θ
Training Data
Learning Algorithm
Testing Data
Prediction Function θ^*
Evaluation score

Training


Real World Problem
Model the problem
Formal Model θ
Learning Algorithm
Training Data
Testing Data
Prediction Function θ^*
Evaluation score

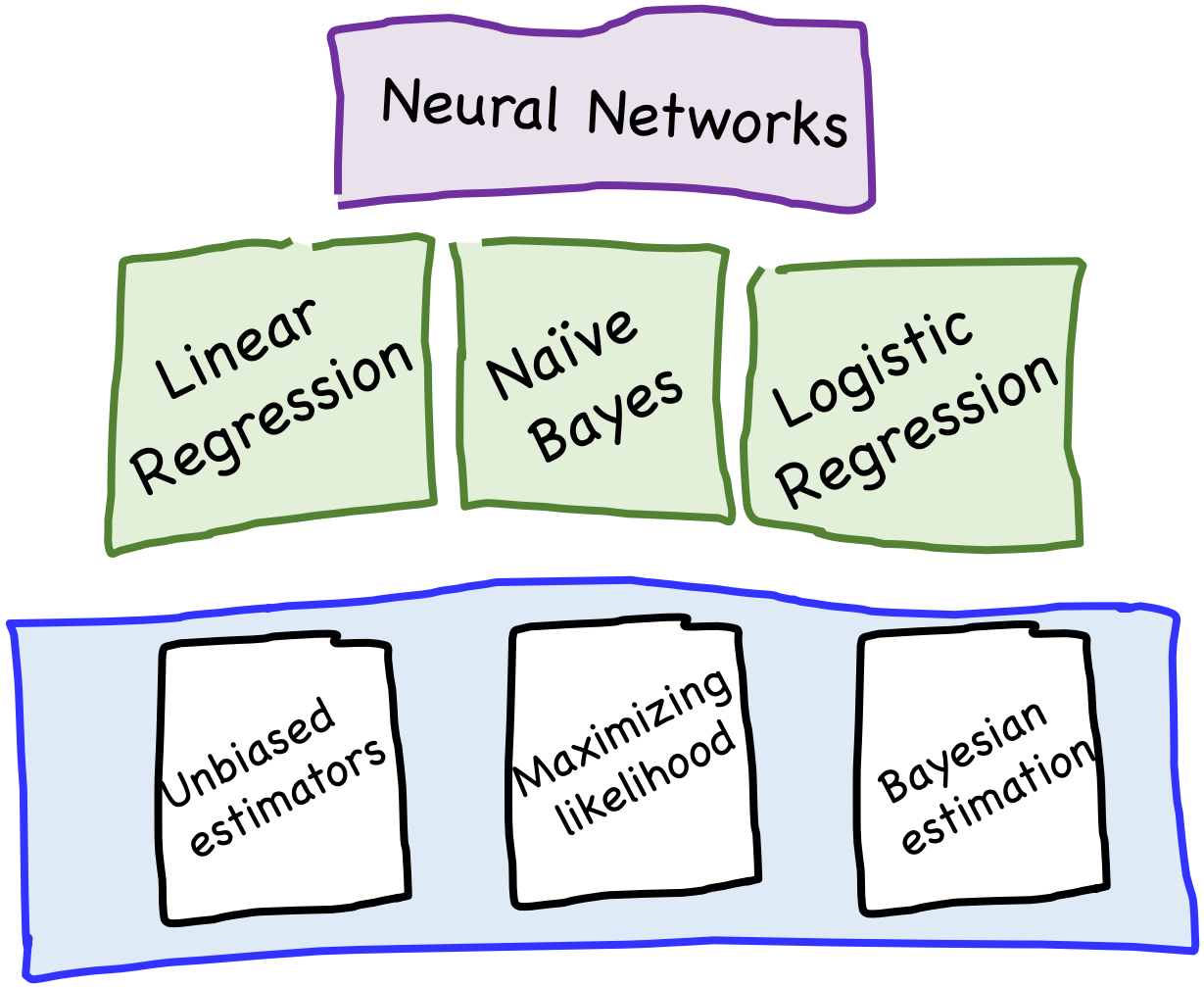
Testing


Real World Problem
Model the problem
Formal Model θ
Training Data
Learning Algorithm
Testing Data
Prediction Function θ^*
Evaluation score

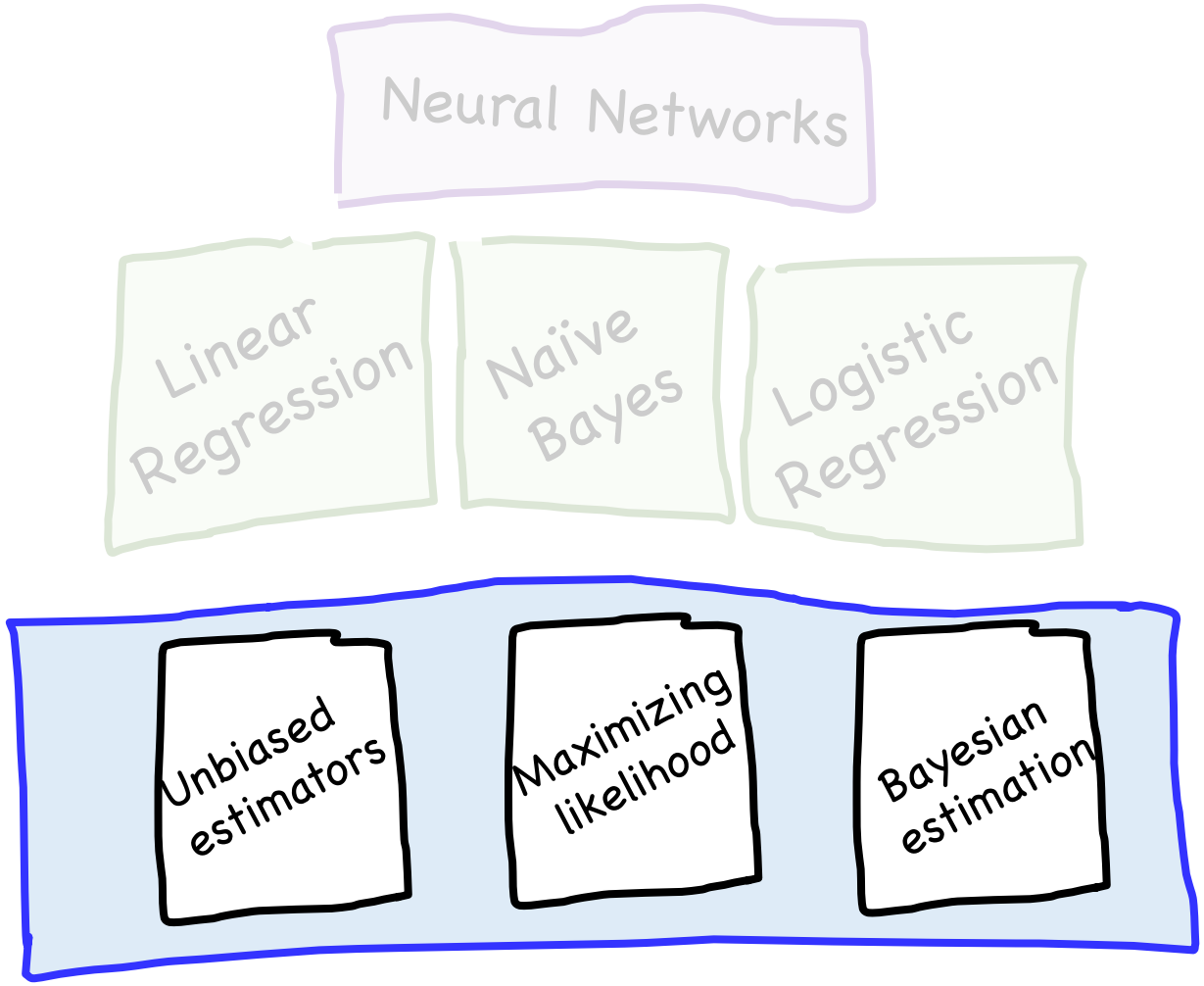
End Review

Basis for learning from data

Our Path


Neural Networks
Linear Regression
Naïve Bayes
Logistic Regression
Unbiased estimators
Maximizing likelihood
Bayesian estimation

Parameter Estimation



Recall Sample Mean + Variance?

- Consider *n* I.I.D. random variables X_1, X_2, ... X_n
  - X_i have distribution *F* with $E[X_i] = \mu$ and $Var(X_i) = \sigma^2$
  - We call sequence of X_i a **sample** from distribution *F*
  - Recall sample mean: $\overline{X} = \sum_{i=1}^{n} \frac{X_i}{n}$ where $E[\overline{X}] = \mu$

$$\overline{X} \sim N(\mu, \frac{\sigma^2}{n}) \quad \text{as} \quad n \to \infty$$

  - Recall sample variance:

$$S^2 = \sum_{i=1}^{n} \frac{(X_i - \overline{X})^2}{n-1} = \text{undefined}$$

Estimate parameters for Bernoulli and Normal

Limited tool: how could we use that for fitting a “Mixture of Gaussians”?

Great idea in Machine Learning

Likelihood of Data

- Consider *n* I.I.D. random variables X_1, X_2, ..., X_n
  - X_i is a sample from density function $f(X_i \mid \theta)$
    - Note: now explicitly specify parameter θ of distribution
  - We want to determine how "likely" the observed data $(x_1, x_2, ..., x_n)$ is based on density $f(X_i \mid \theta)$
  - Define the **Likelihood function**, $L(\theta)$:

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta)$$

    - This is just a product since X_i are I.I.D.
  - Intuitively: what is probability of observed data using density function $f(X_i \mid \theta)$, for some choice of θ

Maximum Likelihood Estimator

- The **Maximum Likelihood Estimator** (MLE) of θ, is the value of θ that maximizes $L(\theta)$
  - More formally: $\theta_{MLE} = \arg\max_{\theta} L(\theta)$

Likelihood (of data given parameters):

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta)$$

Likelihood (of data given parameters):

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta)$$

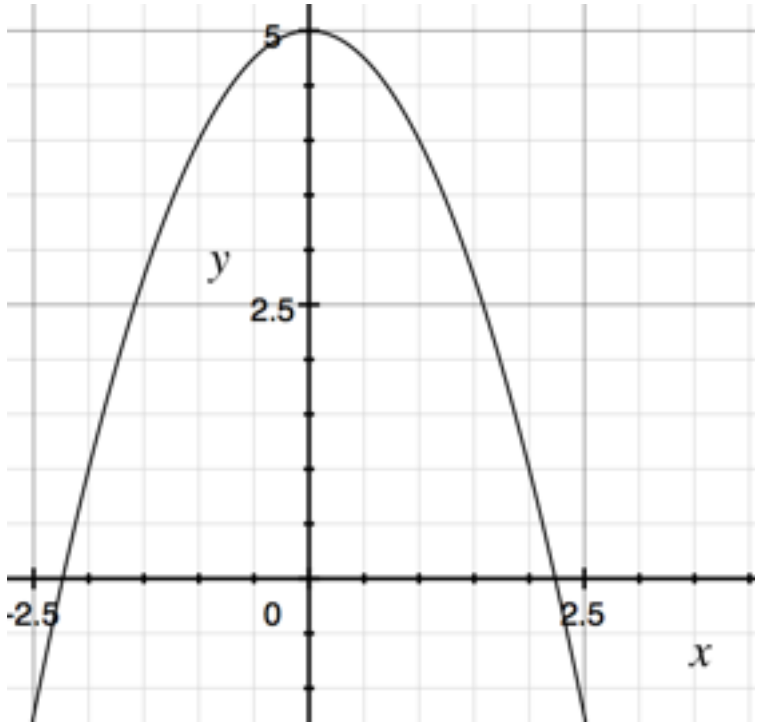
Either the
PDF (continuous) or
PMF (discrete)

Argmax

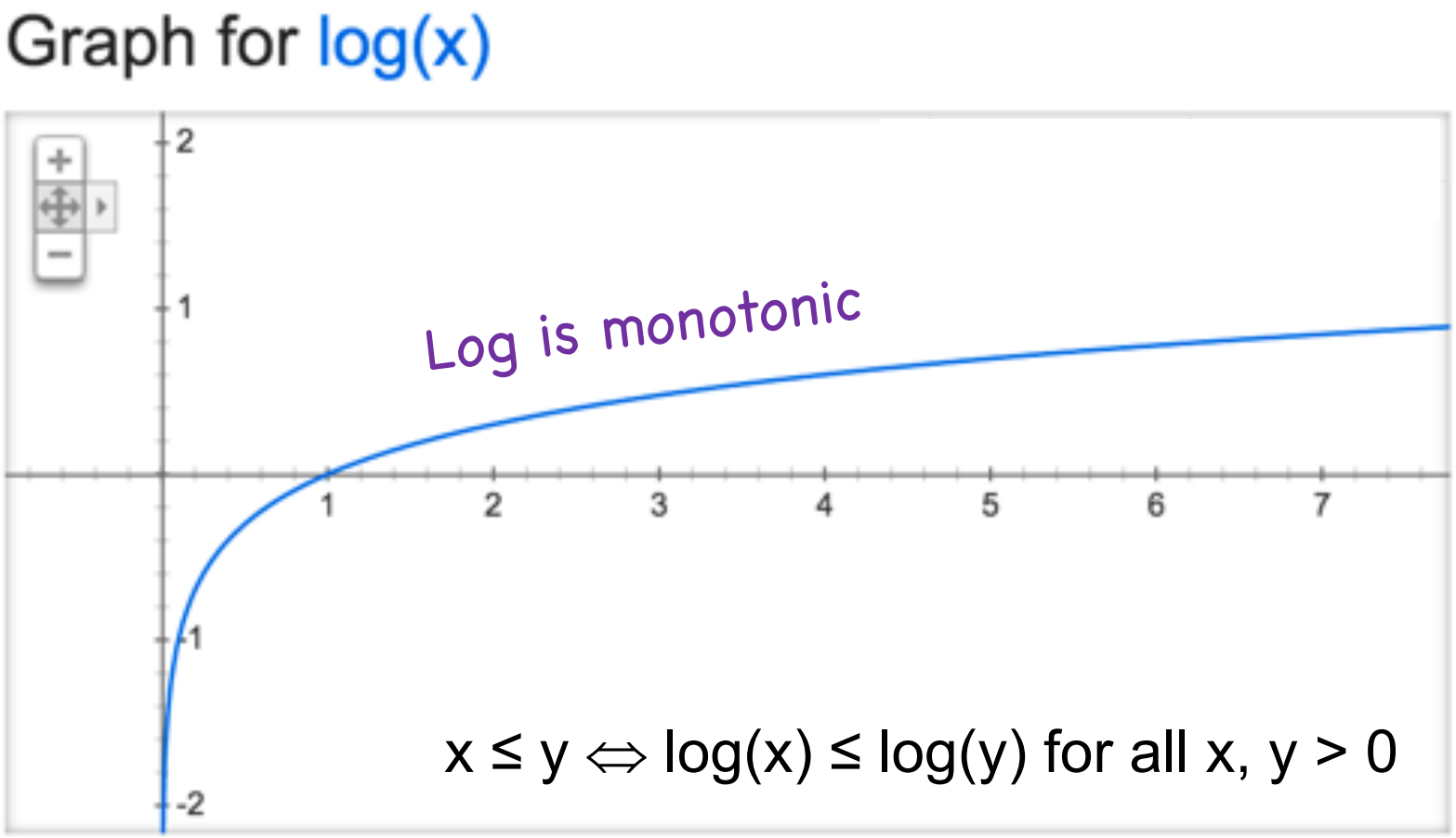
$$f(x) = -x^2 + 5$$

$$\max_x \; -x^2 + 5 = 5$$

$$\operatorname*{argmax}_x \; -x^2 + 5 = 0$$



Argmax of Log



Claim: $\underset{x}{\operatorname{argmax}}\ f(x) = \underset{x}{\operatorname{argmax}}\ \log f(x)$

Argmax of Log

$$\underset{x}{\operatorname{argmax}}\ f(x) = \underset{x}{\operatorname{argmax}}\ \log f(x)$$

Log I Love You

$$\log(ab) = \log(a) + \log(b)$$

Natural Log

Maximum Likelihood Estimator

- The **Maximum Likelihood Estimator** (MLE) of θ, is the value of θ that maximizes $L(\theta)$
  - More formally: $\theta_{MLE} = \arg\max_{\theta} L(\theta)$
  - More convenient to use **log-likelihood function**, $LL(\theta)$:

$$LL(\theta) = \log L(\theta) = \log \prod_{i=1}^{n} f(X_i \mid \theta) = \sum_{i=1}^{n} \log f(X_i \mid \theta)$$

  - θ that maximizes $LL(\theta)$ also maximizes $L(\theta)$
    - Formally: $\arg\max_{\theta} LL(\theta) = \arg\max_{\theta} L(\theta)$
    - Similarly, for any positive constant c (not dependent on θ):

$$\arg\max_{\theta}(c \cdot LL(\theta)) = \arg\max_{\theta} LL(\theta) = \arg\max_{\theta} L(\theta)$$

Story so far: We can chose parameters by finding the argmax of the log likelihood of our data

Maximum Likelihood

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta)$$

$$LL(\theta) = \sum_{i=1}^{n} \log f(X_i|\theta)$$

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\ LL(\theta)$$

But how do we compute argmax?

Option #1: Straight optimization

Computing the MLE

- General approach for finding MLE of θ
  - Determine formula for $LL(\theta)$
  - Differentiate $LL(\theta)$ w.r.t. (each) θ: $\frac{\partial LL(\theta)}{\partial \theta}$
  - To maximize, set $\frac{\partial LL(\theta)}{\partial \theta} = 0$
  - Solve resulting (simultaneous) equations to get θ_{MLE}
    - Make sure that derived $\hat{\theta}_{MLE}$ is actually a maximum (and not a minimum or saddle point). E.g., check $LL(\theta_{MLE} \pm \varepsilon) < LL(\theta_{MLE})$
      - This step often ignored in expository derivations
      - So, we'll ignore it here too (and won't require it in this class)

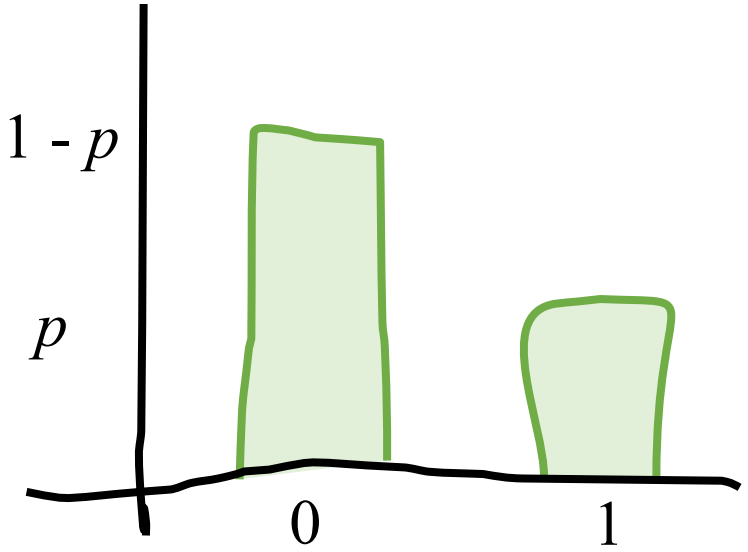
Maximizing Likelihood with Bernoulli

- Consider I.I.D. random variables $X_1, X_2, ..., X_n$
  - $X_i \sim \text{Ber}(p)$
  - Probability mass function, $f(X_i \mid p)$:

Maximizing Likelihood with Bernoulli

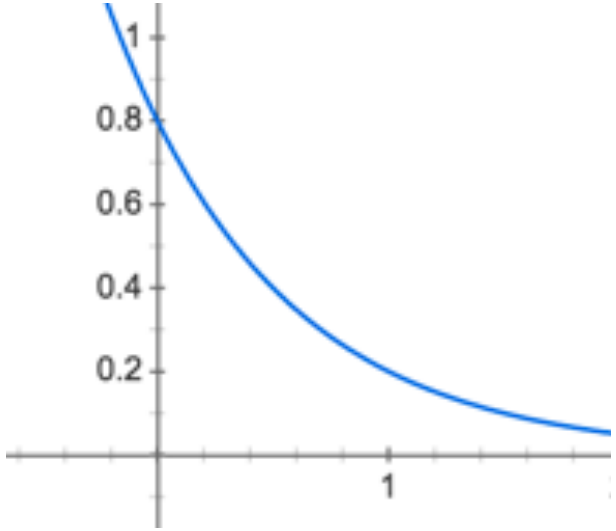
- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - X_i ~ Ber(p)
  - Probability mass function, $f(X_i \mid p)$:

PMF of Bernoulli



$$f(X_i \mid p) = p^{x_i}(1-p)^{1-x_i}$$

PMF of Bernoulli (p = 0.2)



$$f(x) = 0.2^x(1-0.2)^{1-x}$$

Bernoulli PMF

$$X \sim \text{Ber}(p)$$

$$f(X = x|p) = p^x(1-p)^{1-x}$$

Maximizing Likelihood with Bernoulli

- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - $X_i \sim \text{Ber}(p)$
  - Probability mass function, $f(X_i \mid p)$, can be written as:

$$f(X_i \mid p) = p^{x_i}(1-p)^{1-x_i} \quad \text{where } x_i = 0 \text{ or } 1$$

  - Likelihood: $L(\theta) = \prod_{i=1}^{n} p^{X_i}(1-p)^{1-X_i}$
  - Log-likelihood:

$$LL(\theta) = \sum_{i=1}^{n} \log\left(p^{X_i}(1-p)^{1-X_i}\right) = \sum_{i=1}^{n} \left[X_i(\log p) + (1-X_i)\log(1-p)\right]$$

$$= Y(\log p) + (n-Y)\log(1-p) \quad \text{where } Y = \sum_{i=1}^{n} X_i$$

  - Differentiate w.r.t. p, and set to 0:

$$\frac{\partial LL(p)}{\partial p} = Y\frac{1}{p} + (n-Y)\frac{-1}{1-p} = 0 \quad \Rightarrow \quad p_{MLE} = \frac{Y}{n} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Isn’t that the same as unbiased estimator?

Yes. For Bernoulli.

Maximum Likelihood Algorithm

1. Decide on a model for the distribution of your samples. Define the PMF / PDF for your sample.

2. Write out the log likelihood function.

3. State that the optimal parameters are the argmax of the log likelihood function.

4. Use an optimization algorithm to calculate argmax

Maximizing Likelihood with Poisson

- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - $X_i \sim \text{Poi}(\lambda)$
  - PMF: $f(X_i \mid \lambda) = \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}$ Likelihood: $L(\theta) = \prod_{i=1}^{n} \frac{e^{-\lambda}\lambda^{X_i}}{X_i!}$
  - Log-likelihood:

$$LL(\theta) = \sum_{i=1}^{n} \log\left(\frac{e^{-\lambda}\lambda^{X_i}}{X_i!}\right) = \sum_{i=1}^{n} \left[-\lambda \log(e) + X_i \log(\lambda) - \log(X_i!)\right]$$

$$= -n\lambda + \log(\lambda)\sum_{i=1}^{n} X_i - \sum_{i=1}^{n} \log(X_i!)$$

  - Differentiate w.r.t. λ, and set to 0:

$$\frac{\partial LL(\lambda)}{\partial \lambda} = -n + \frac{1}{\lambda}\sum_{i=1}^{n} X_i = 0 \quad \Rightarrow \quad \lambda_{MLE} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

It’s so general!

Maximizing Likelihood with Normal

- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - $X_i \sim N(\mu, \sigma^2)$
  - PDF: $f(X_i \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(X_i-\mu)^2/(2\sigma^2)}$
  - Log-likelihood:

$$LL(\theta) = \sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi}\sigma} e^{-(X_i-\mu)^2/(2\sigma^2)}\right) = \sum_{i=1}^{n} \left[-\log(\sqrt{2\pi}\sigma) - (X_i-\mu)^2/(2\sigma^2)\right]$$

  - First, differentiate w.r.t. μ, and set to 0:

$$\frac{\partial LL(\mu,\sigma^2)}{\partial \mu} = \sum_{i=1}^{n} 2(X_i-\mu)/(2\sigma^2) = \frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i-\mu) = 0$$

  - Then, differentiate w.r.t. σ, and set to 0:

$$\frac{\partial LL(\mu,\sigma^2)}{\partial \sigma} = \sum_{i=1}^{n} -\frac{1}{\sigma} + 2(X_i-\mu)^2/(2\sigma^3) = -\frac{n}{\sigma} + \sum_{i=1}^{n}(X_i-\mu)^2/(\sigma^3) = 0$$

Being Normal, Simultaneously

- Now have two equations, two unknowns:

$$\frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i-\mu)=0 \qquad -\frac{n}{\sigma}+\sum_{i=1}^{n}(X_i-\mu)^2/(\sigma^3)=0$$

- First, solve for μ_{MLE}:

$$\frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i-\mu)=0 \;\Rightarrow\; \sum_{i=1}^{n}X_i=n\mu \;\Rightarrow\; \mu_{MLE}=\frac{1}{n}\sum_{i=1}^{n}X_i$$

- Then, solve for σ^2_{MLE}:

$$-\frac{n}{\sigma}+\sum_{i=1}^{n}(X_i-\mu)^2/(\sigma^3)=0 \;\Rightarrow\; n\sigma^2=\sum_{i=1}^{n}(X_i-\mu)^2$$

$$\sigma^2_{MLE}=\frac{1}{n}\sum_{i=1}^{n}(X_i-\mu_{MLE})^2$$

- Note: μ_{MLE} unbiased, but σ^2_{MLE} biased

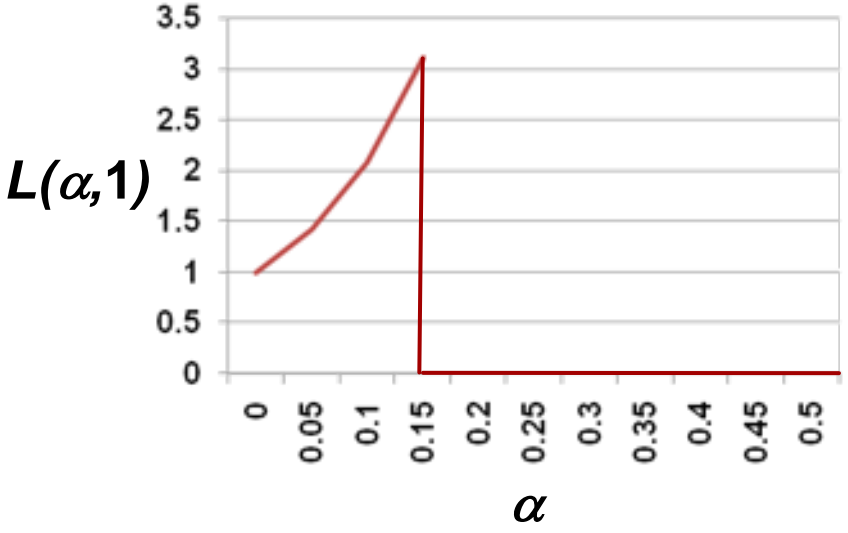
Maximizing Likelihood with Uniform

- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - $X_i \sim \text{Uni}(\alpha, \beta)$
  - PDF: $f(X_i \mid \alpha, \beta) = \begin{cases} \frac{1}{\beta - \alpha} & \alpha \le x_i \le \beta \\ 0 & \text{otherwise} \end{cases}$
  - Likelihood: $L(\theta) = \begin{cases} \left(\frac{1}{\beta - \alpha}\right)^n & \alpha \le x_1, x_2, \ldots, x_n \le \beta \\ 0 & \text{otherwise} \end{cases}$
    - Constraint $\alpha \le x_1, x_2, \ldots, x_n \le \beta$ makes differentiation tricky
    - Intuition: want interval size $(\beta - \alpha)$ to be as small as possible to maximize likelihood function for each data point
    - But need to make sure all observed data contained in interval
      - If all observed data not in interval, then $L(\theta) = 0$
  - Solution: $\alpha_{MLE} = \min(x_1, \ldots, x_n)$ $\beta_{MLE} = \max(x_1, \ldots, x_n)$

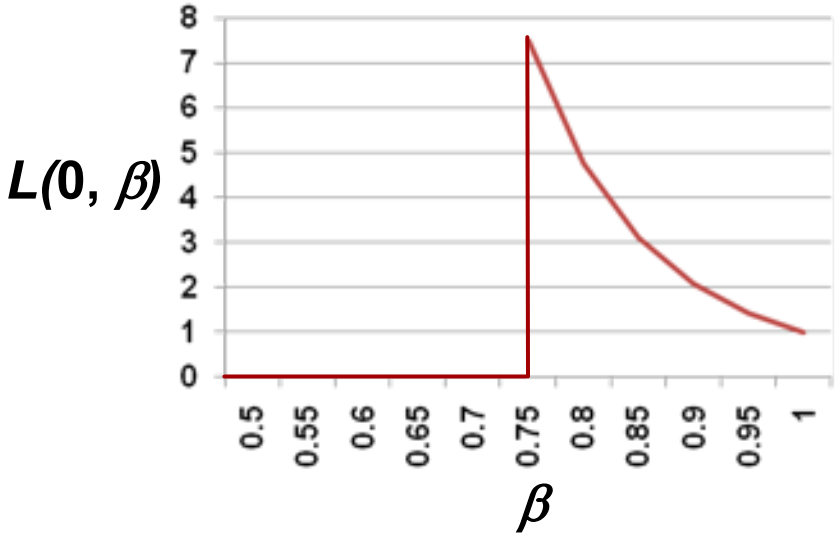
Understanding MLE with Uniform

- Consider I.I.D. random variables X_1, X_2, ..., X_n
  - X_i ~ Uni(0, 1)
  - Observe data:
    - 0.15, 0.20, 0.30, 0.40, 0.65, 0.70, 0.75

Likelihood: L(α,1)



Likelihood: L(0, β)



Small Samples = Problems

- How do small samples affect MLE?
  - In many cases, $\mu_{MLE} = \frac{1}{n}\sum_{i=1}^{n} X_i$ = sample mean
    - Unbiased. Not too shabby…
  - As seen with Normal, $\sigma^2_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \mu_{MLE})^2$
    - Biased. Underestimates for small *n* (e.g., 0 for n = 1)
  - As seen with Uniform, $\alpha_{MLE} \geq \alpha$ and $\beta_{MLE} \leq \beta$
    - Biased. Problematic for small *n* (e.g., $\alpha = \beta$ when n = 1)
  - Small sample phenomena intuitively make sense:
    - Maximum likelihood $\Rightarrow$ best explain data we've seen
    - Does not attempt to generalize to unseen data

Properties of MLE

- Maximum Likelihood Estimators are generally:
  - **Consistent**: $\lim_{n\to\infty} P(|\hat{\theta} - \theta| < \varepsilon) = 1$ for $\varepsilon > 0$
  - Potentially biased (though asymptotically less so)
  - **Asymptotically optimal**
    - Has smallest variance of "good" estimators for large samples
  - **Often used in practice** where sample size is large relative to parameter space
    - But be careful, there are some very large parameter spaces

Machine Learning:
Learn parameters (mostly with MLE) for probabilistic models.

Stretch!

SCPD Code:

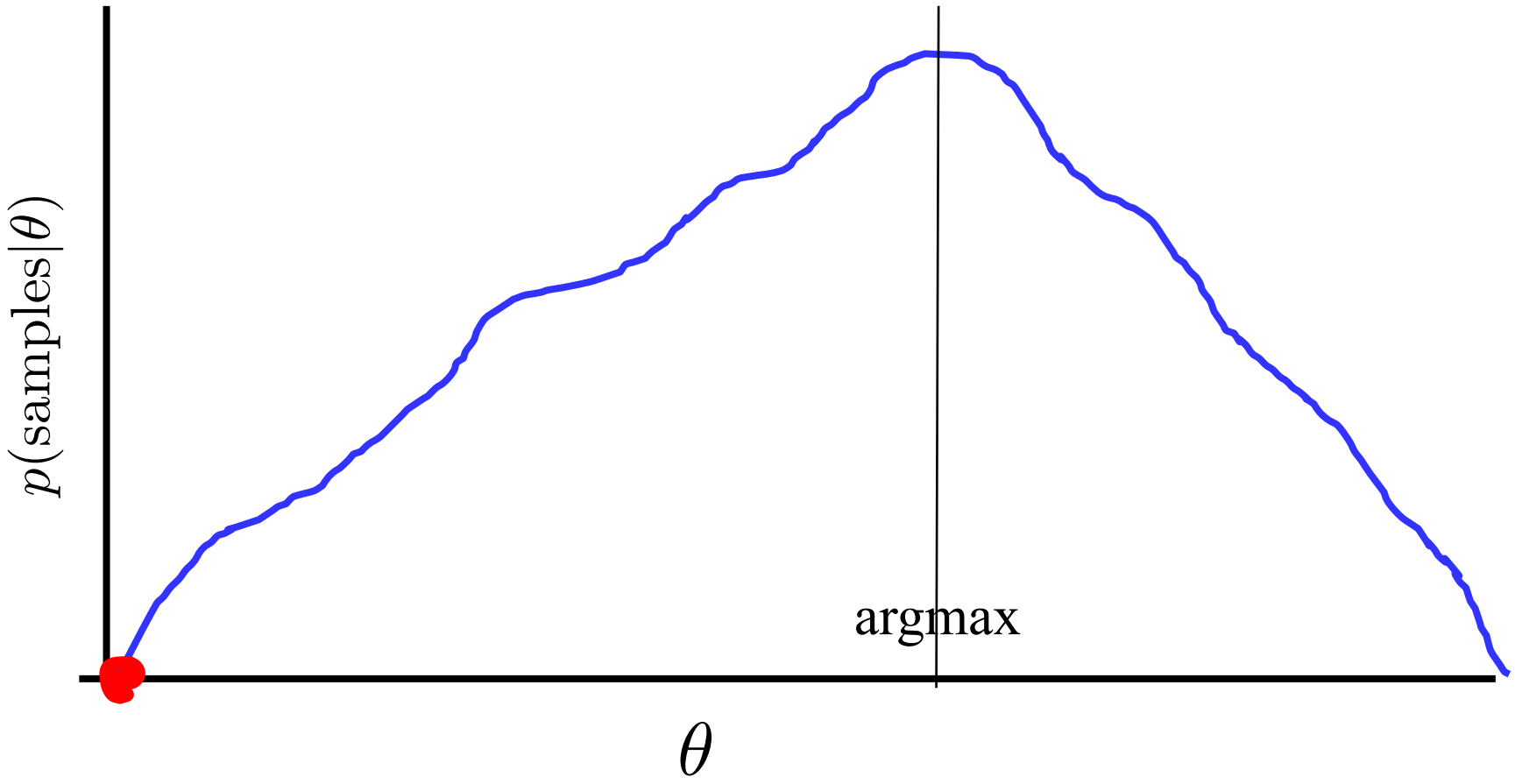
Gradient Ascent

Noah Arthurs
CS109, Stanford University

Argmax
Option #1: Straight optimization

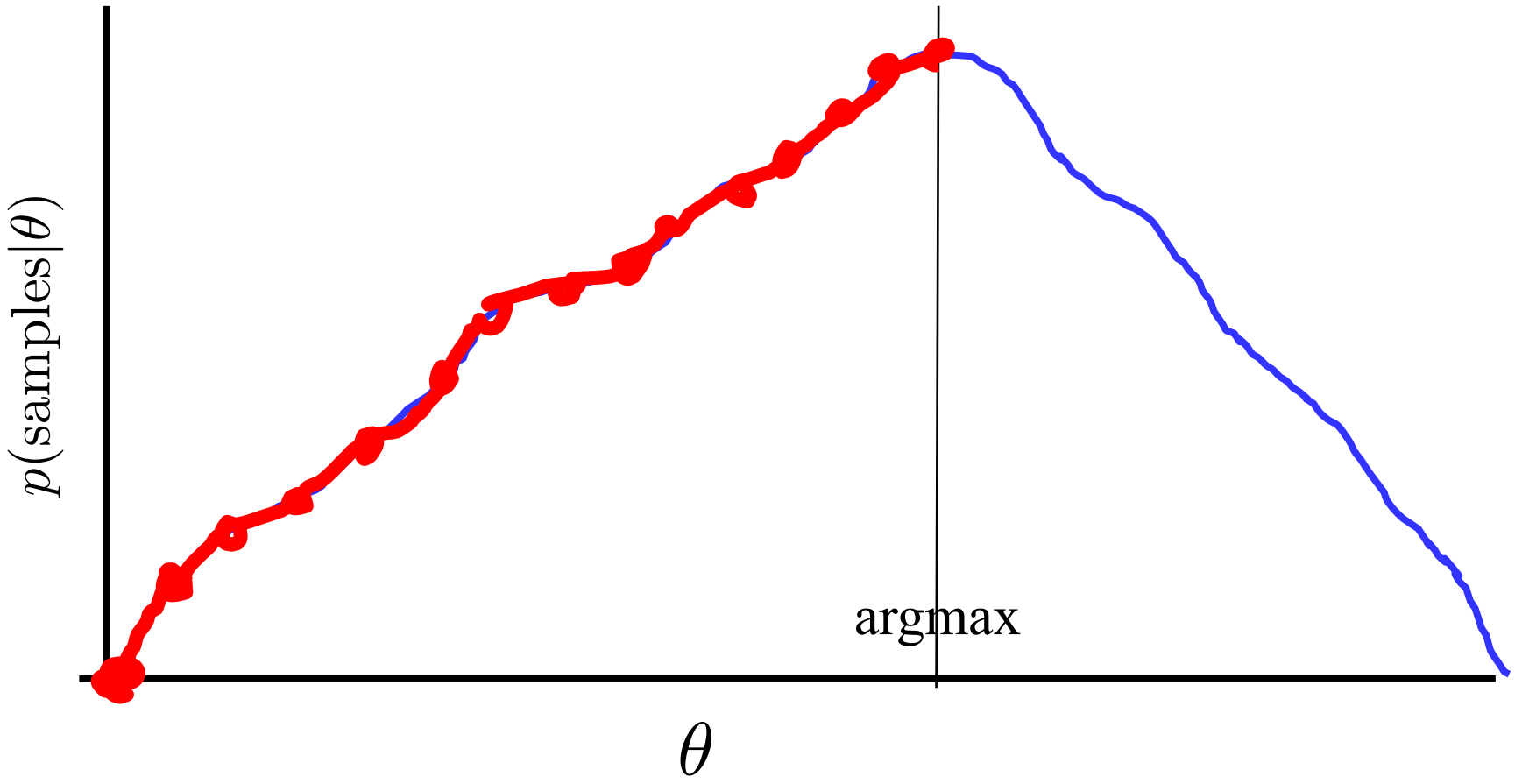
Argmax
Option #2: Gradient Ascent

Gradient Ascent



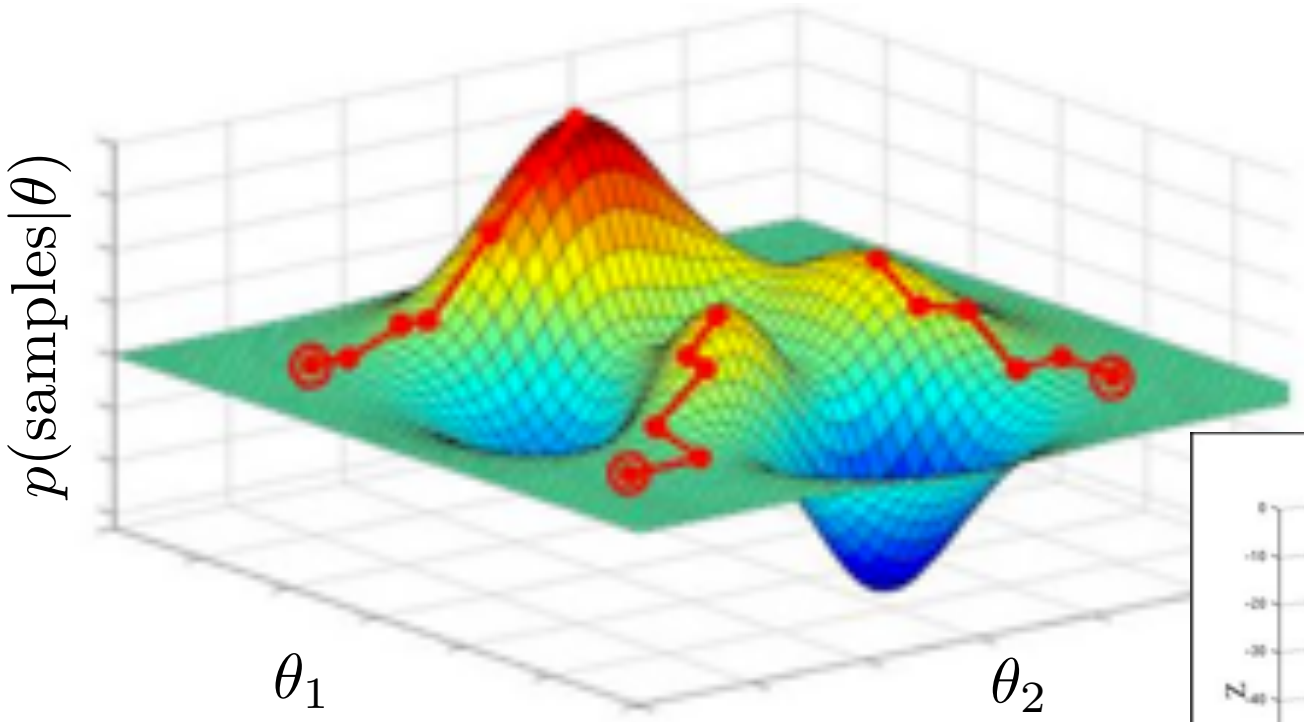
Walk uphill and you will find a local maxima
(if your step size is small enough)

Gradient Ascent

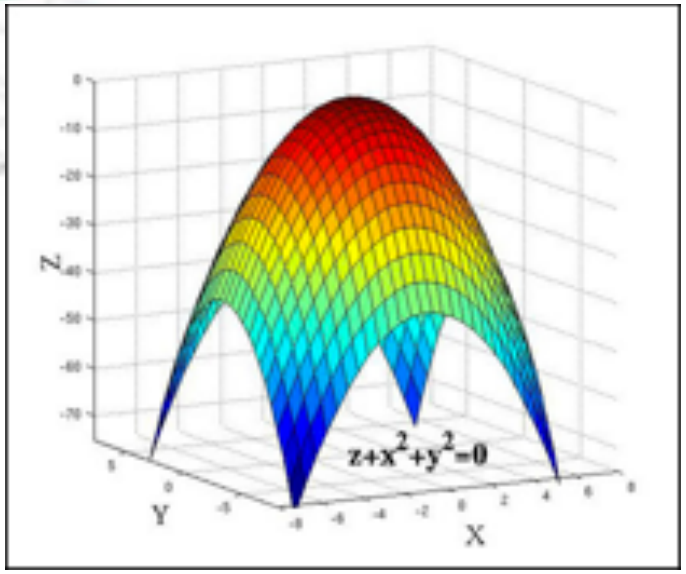


Walk uphill and you will find a local maxima
(if your step size is small enough)

Gradient Ascent



Especially good if function is convex



Walk uphill and you will find a local maxima
(if your step size is small enough)

Gradient Ascent

Repeat many times

$$\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \frac{\partial LL(\theta^{\text{old}})}{\partial \theta_j^{\text{old}}}$$

This is some **profound** life philosophy

Walk uphill and you will find a local maxima
(if your step size is small enough)

Gradient ascent is your bread and butter algorithm for optimization (eg argmax)