



General Inference

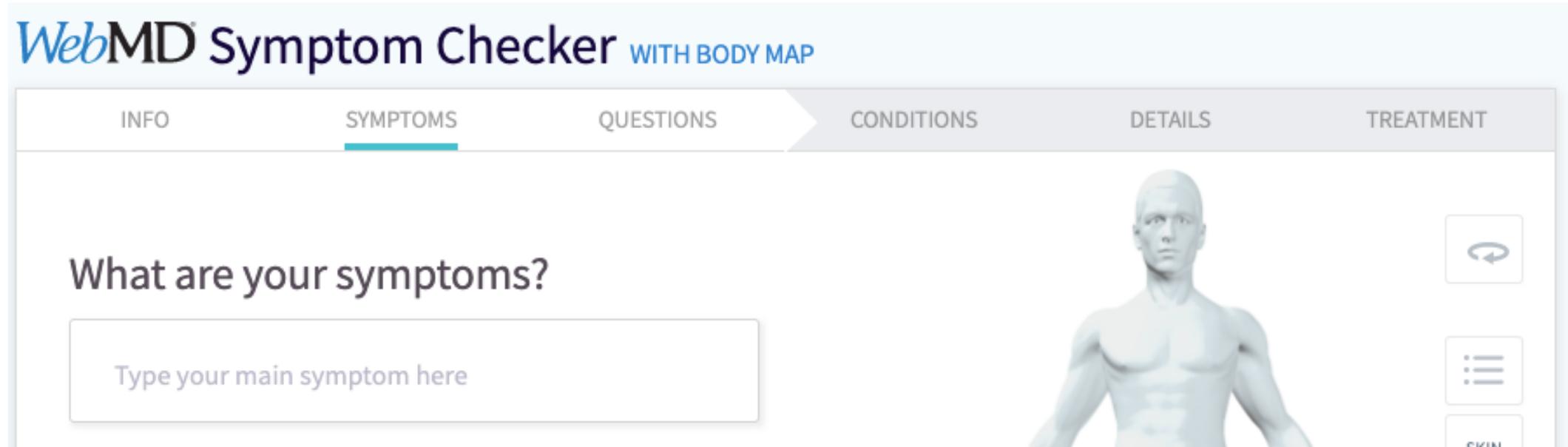
Chris Piech

CS109, Stanford University

Why You Need a Model

*Web*MD[®]

Why You Need a Model



Multiple Random Variables. Start of Digital Revolution

Conditions that match your symptoms

UNDERSTANDING YOUR RESULTS [i](#)

Migraine headache (adult)



Moderate match



Acute Sinusitis



Fair match



Stroke



Fair match



Gender **Male**

Age **30**

[Edit](#)

My Symptoms

[Edit](#)

dizziness, one sided headache

Surprisingly Simple (if you can code)

Code



Probability

Three Guiding Questions

1. How do people actually define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

At this point you know inference with
two random variables

Today: Five New Real + Exciting Problems

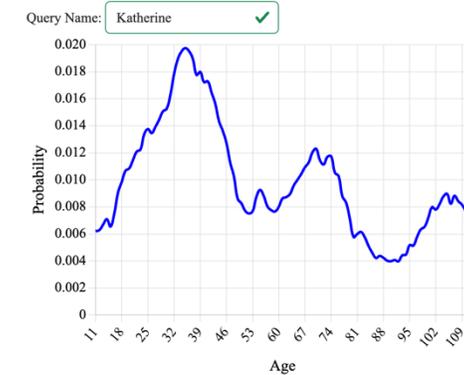
Age from C14



Updated Delivery Prob



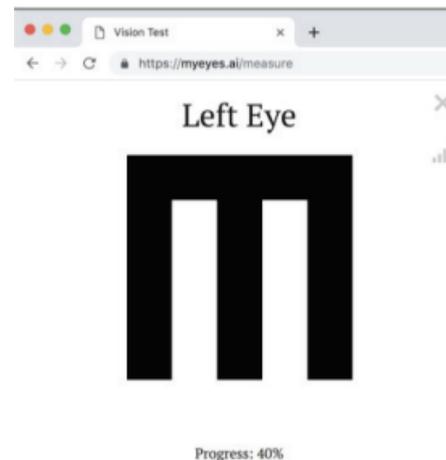
Age from Name



Hidden Chambers

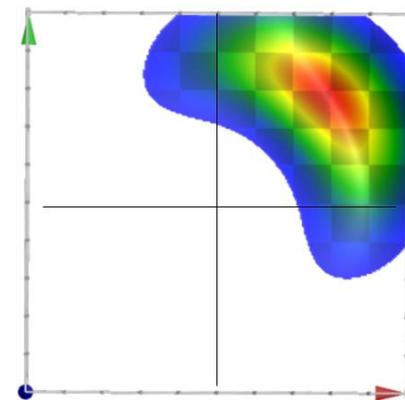


Stanford Eye Test



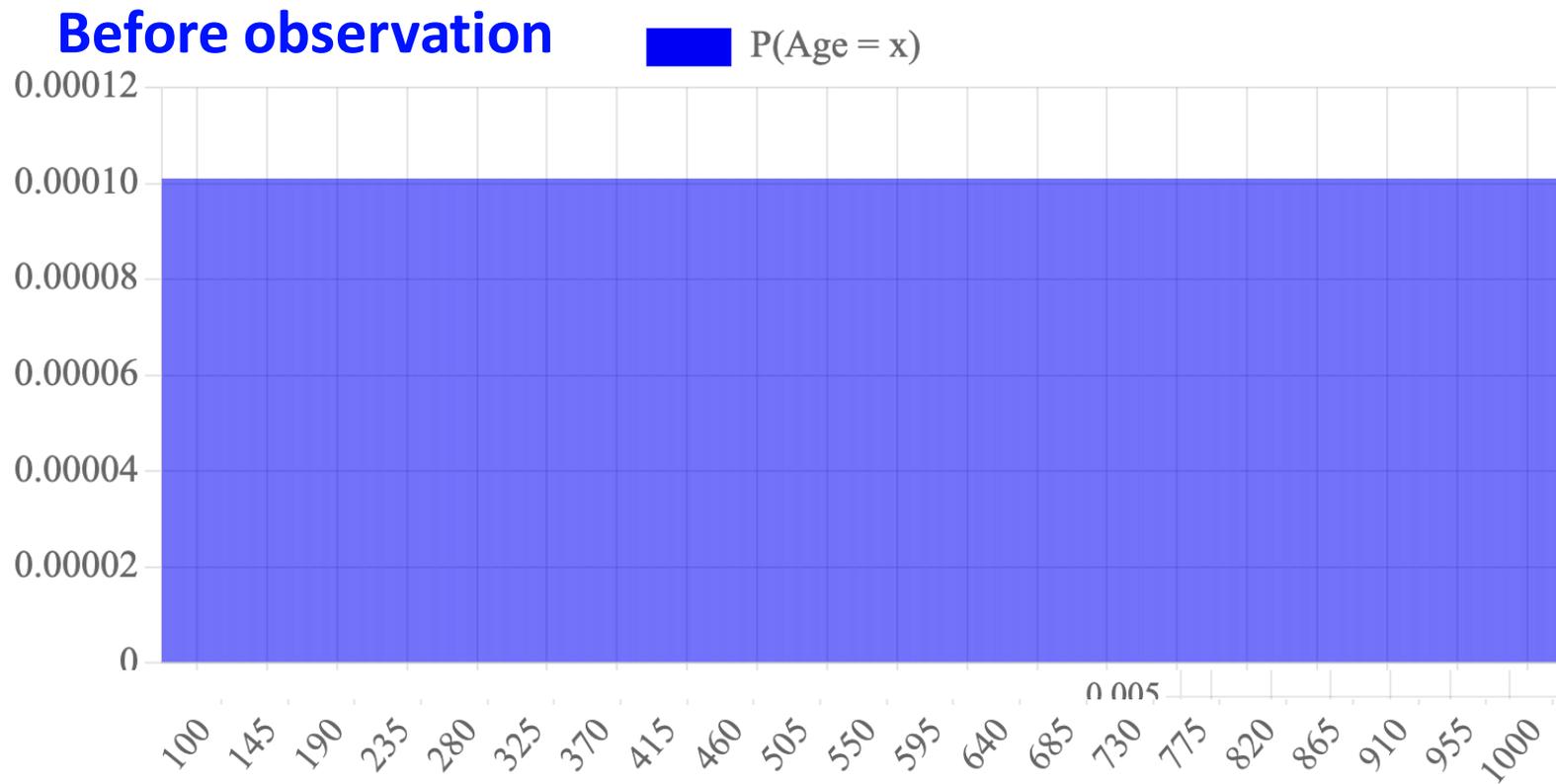
Chris Piech, CS109

Cellphone Tracking Cont.



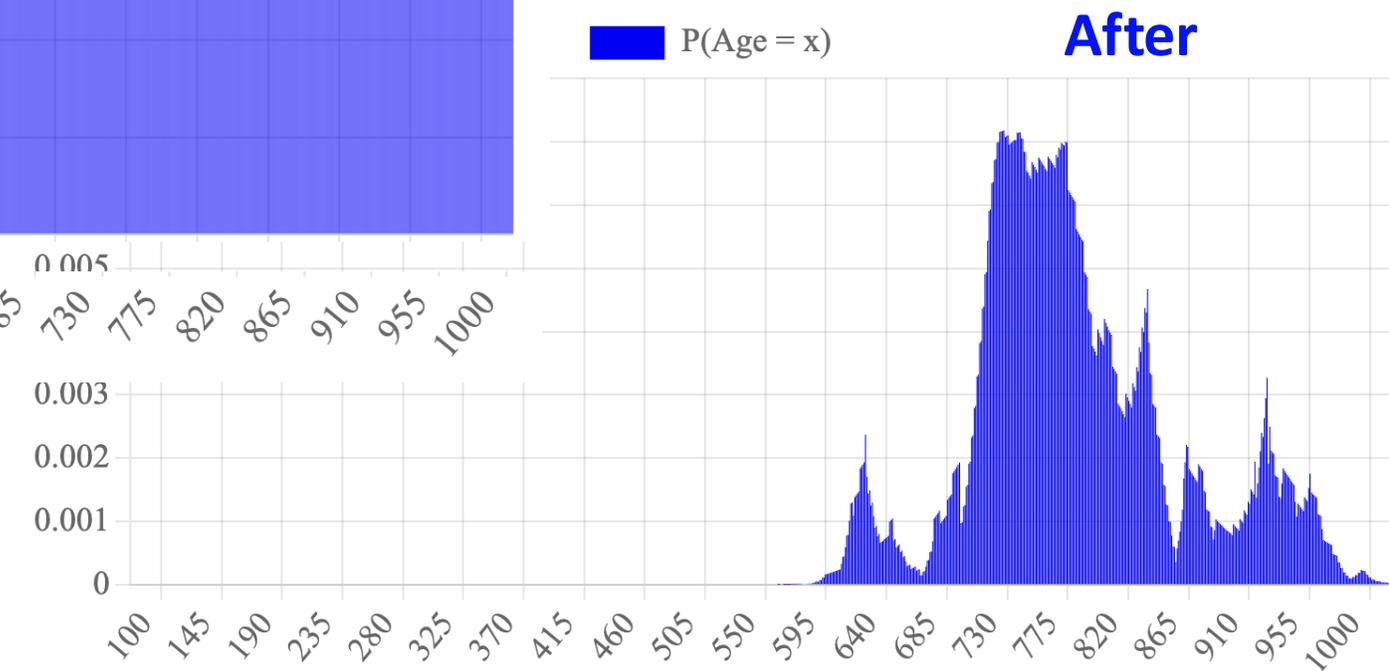
Stanford University

Update Belief PMF

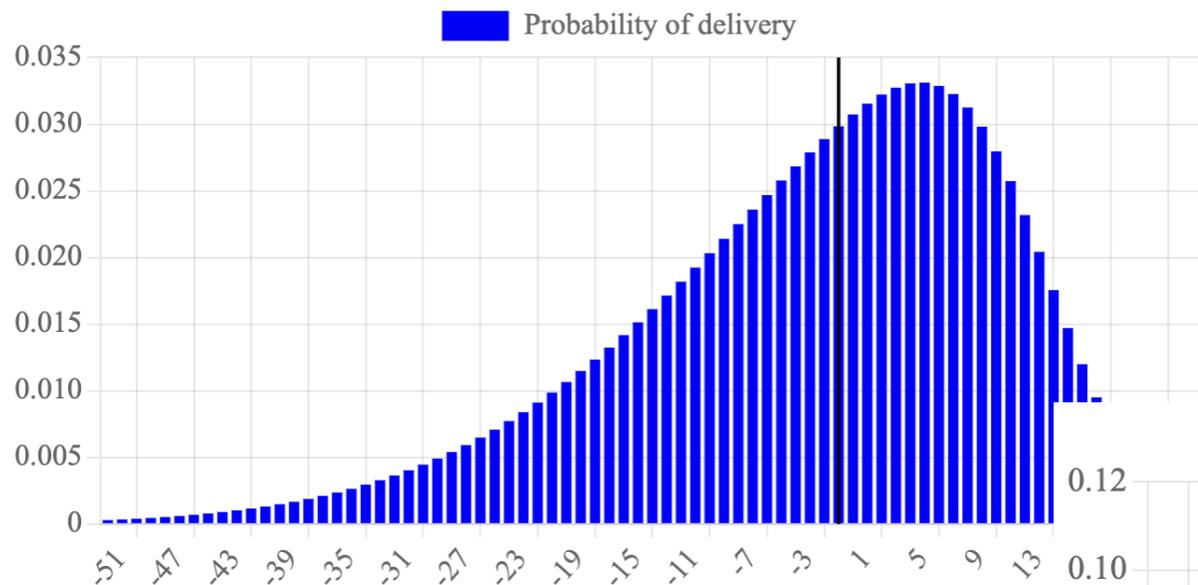


Observation

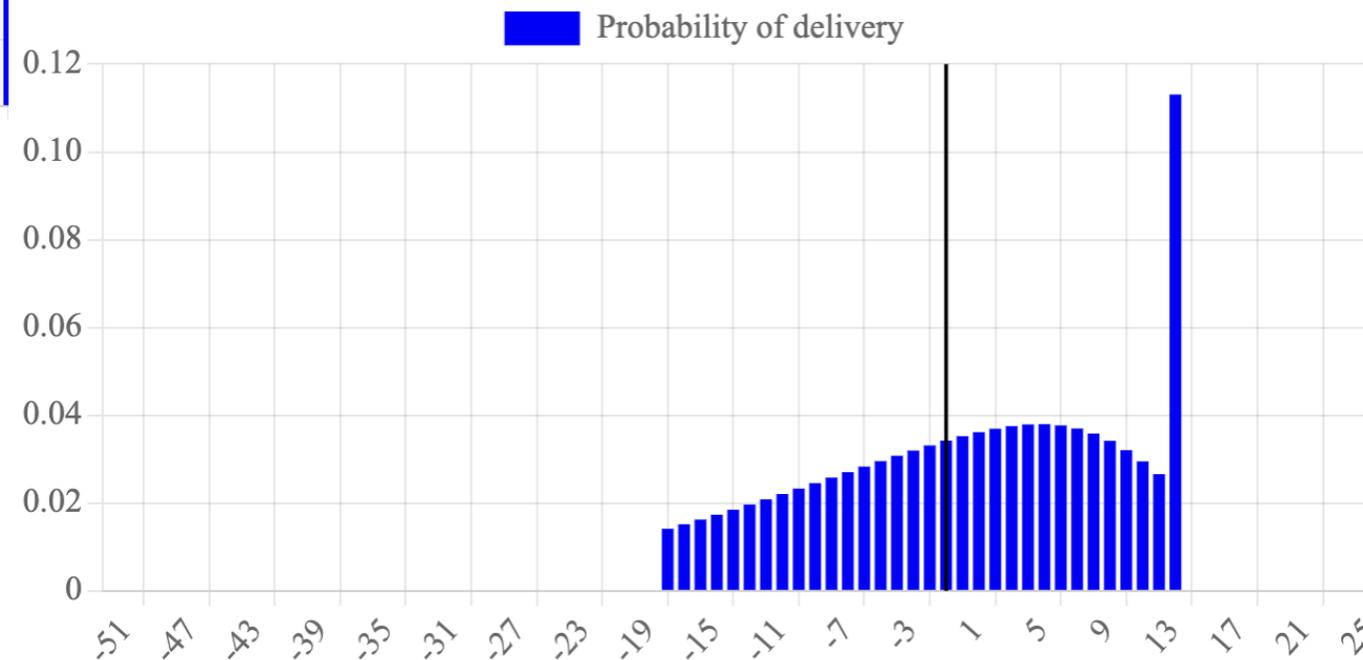
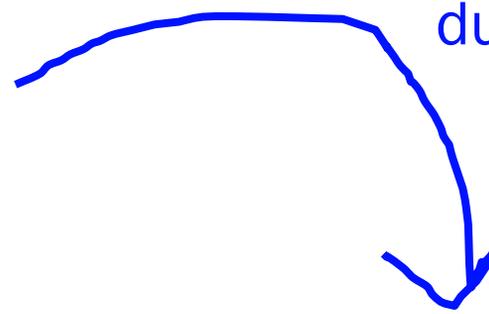
Remaining C14:



Baby delivery



Its 19 days until the due date and no baby



For each value d :

$$P(D = d | \text{no child so far})$$

$$= \frac{P(\text{no child so far} | D = d) P(D = d)}{P(\text{no child so far})}$$

```
def update_belief_carbon_dating(m = 900):  
    # pr_A[i] is P(Age = i | m = 900).  
    pr_A = {}  
    for i in range(100,10000+1):  
        prior = 1 / n_years # P(A = i)  
        likelihood = calc_likelihood(m, i) #P(M=m | A=i)  
        pr_A[i] = likelihood * prior  
    # implicitly computes the normalization constant  
    normalize(pr_A)  
    return pr_A
```

```
def update_belief_baby(prior, today = 10):  
    # pr_D[i] is P(D = i | No Baby Yet).  
    pr_D = {}  
    for i in range(-50,25):  
        # P(NoBaby | D = i)  
        likelihood = 0 if i < today else 1  
        pr_D[i] = likelihood * prior[i]  
    # implicitly computes the LOTP  
    normalize(pr_D)  
    return pr_D
```

What do you notice
is the same. What is
different?

Normalize does so much for you!!!

Normalize: scale each value so that they would sum to 1

```
normalize({  
  'cat'    : 5,  
  'dog'    : 10,  
  'axolotyl' : 5  
})
```



```
{  
  'cat'    : 0.2,  
  'dog'    : 0.4,  
  'axolotyl' : 0.2  
}
```

Hidden Pyramid Chambers with Poisson + Bayes



Hidden Pyramid Chambers with Poisson + Bayes

Number of Muons

$$f(X = x|M = 12) = \frac{P(M = 12|X = x)f(X = x)}{P(M = 12)}$$

Amount of limestone

Denominator option #1: Law of total probability

$$P(M = 12) = \int_0^{100} P(M = 12|X = x)f(X = x)dx$$

Hidden Pyramid Chambers with Poisson + Bayes

Number of Muons

$$f(X = x | M = 12) = \frac{P(M = 12 | X = x) f(X = x)}{P(M = 12)}$$

Amount of limestone

Denominator option #2: Solve for K

$$\begin{aligned} f(X = x | M = 12) &= \frac{P(M = 12 | X = x) f(X = x)}{P(M = 12)} \\ &= \frac{(100 \cdot e^{-x/40})^{12} e^{-(100 \cdot e^{-x/40})}}{12!} \cdot \frac{1}{100} \propto e^{-\frac{12x}{40}} - 100 \cdot e^{-x/40} \end{aligned}$$

Teachers hate him!

See how you (approximate) the
integral of any function with
one simple trick.

ProTip: Integration via Sampling

6 Goodbye integral, my old friend [14 points]

Fall 2017

In this problem we are going to compute probabilities for a random variable X that can take on values in the range $0 \leq x \leq 1$, and has probability density function:

$$f_X(x) = \frac{1}{K} \cdot g(x)$$

Where $g(x)$ is some terribly nasty and non-integratable function. For your sanity I won't even write out g . It is that bad. In such a situation, we can turn to the power of computers to help us (you may assume that while g is impossible to integrate, you were able to code g as a function in a programming language). The key idea that we are going to use is called Monte Carlo Integration:

Generate N values (X_1, X_2, \dots, X_N) uniformly sampled over a range (a, b) . We can approximate the integral of a function h over (a, b) as:

$$\int_a^b h(x) dx \approx \frac{(b-a)}{N} \sum_{i=1}^N h(X_i)$$

Pretty amazing! Why did we bother with integrals at all? This question requires you to write pseudo code. Such code does not have to compile, but it should be specific enough that a knowledgeable programmer could implement what you have described. You may use a function `random(a, b)` which returns a sample from $X \sim \text{Uni}(a, b)$. You may also use a function `g(x)` which is the hard-to-integrate term in $f_X(x)$.

Today: Five New Real + Exciting Problems

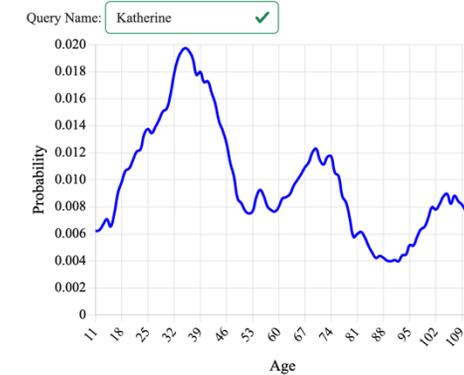
Age from C14



Updated Delivery Prob



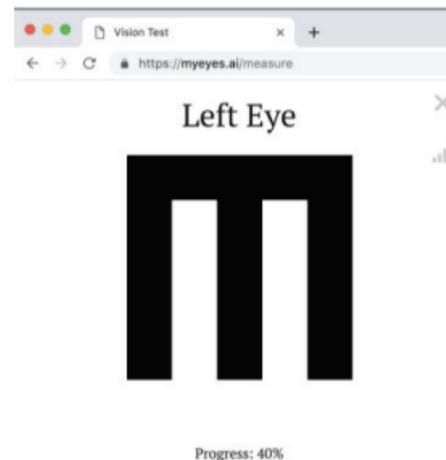
Age from Name



Hidden Chambers

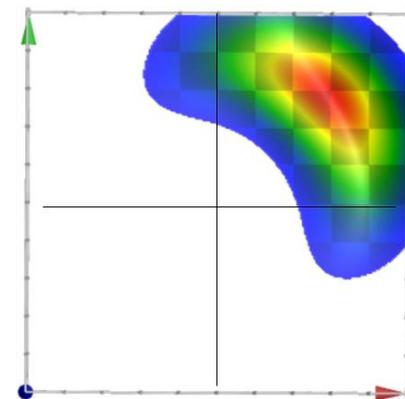


Stanford Eye Test



Chris Piech, CS109

Cellphone Tracking Cont.

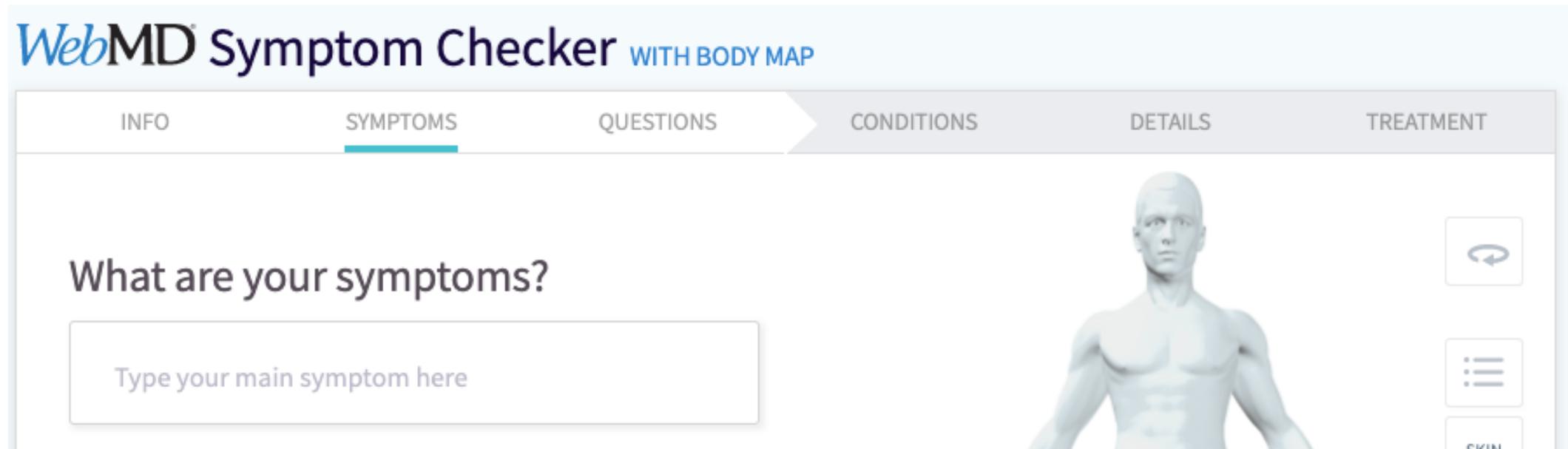


Many real world problems have way more than two random variables...

Why You Need a Model

*Web*MD[®]

Why You Need a Model



Multiple Random Variables. Start of Digital Revolution

Conditions that match your symptoms

UNDERSTANDING YOUR RESULTS [i](#)

Migraine headache (adult)



Moderate match



Acute Sinusitis



Fair match



Stroke



Fair match



Gender **Male**

Age **30**

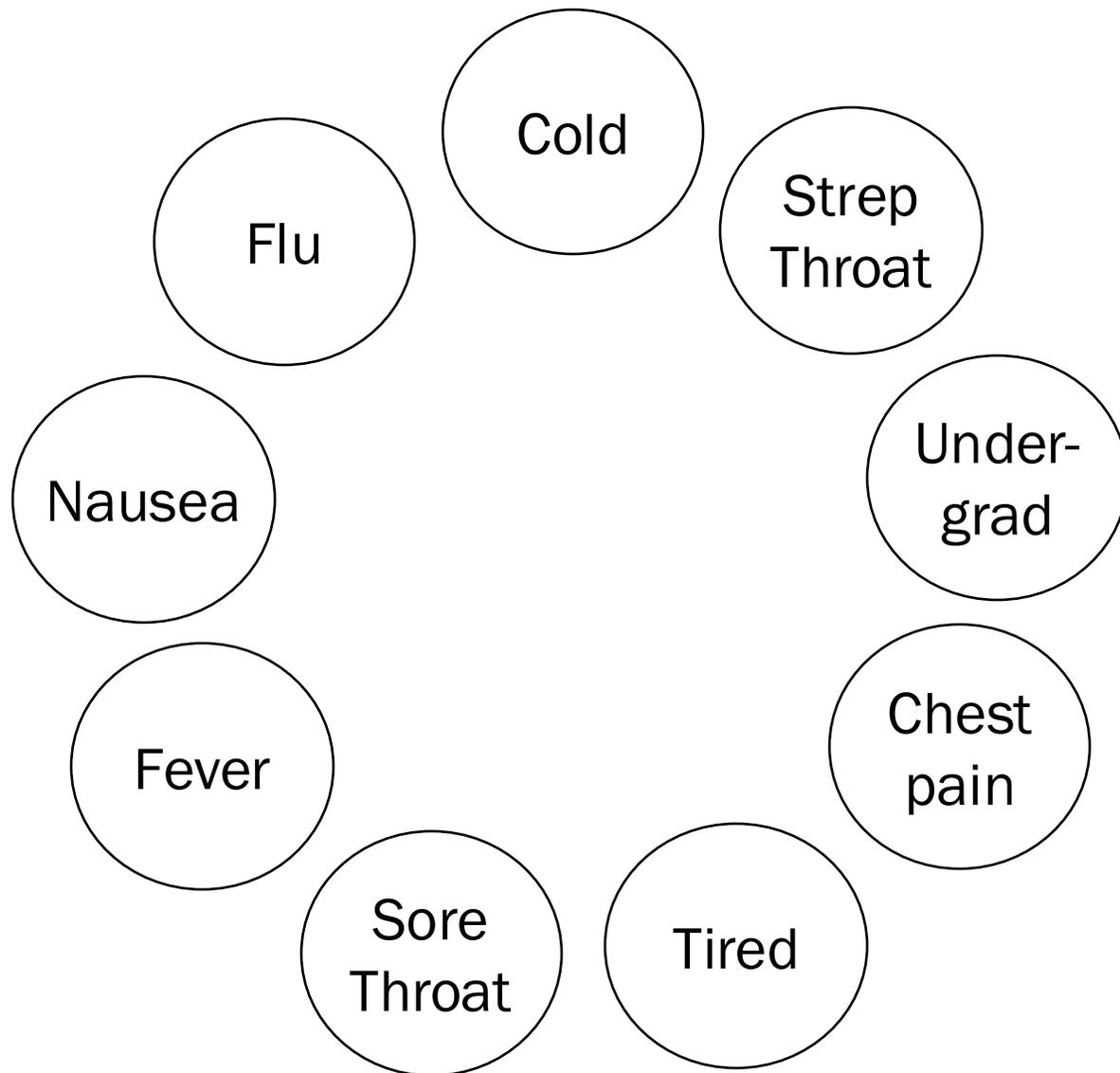
[Edit](#)

My Symptoms

[Edit](#)

dizziness, one sided headache

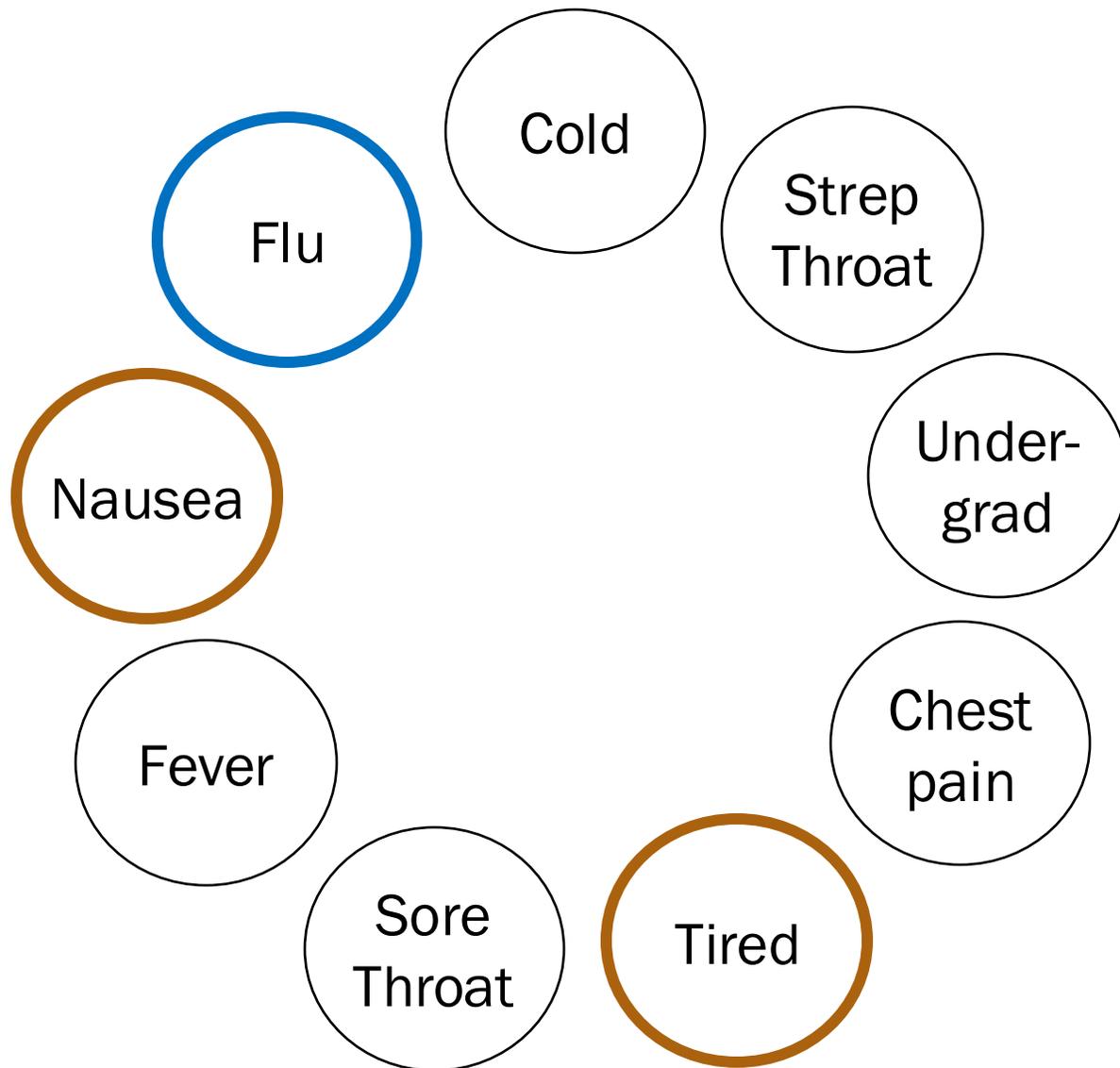
Challenge #1: Many Inference Questions



Inference question:

Given the values of some random variables, what are the conditional distributions of some other random variables?

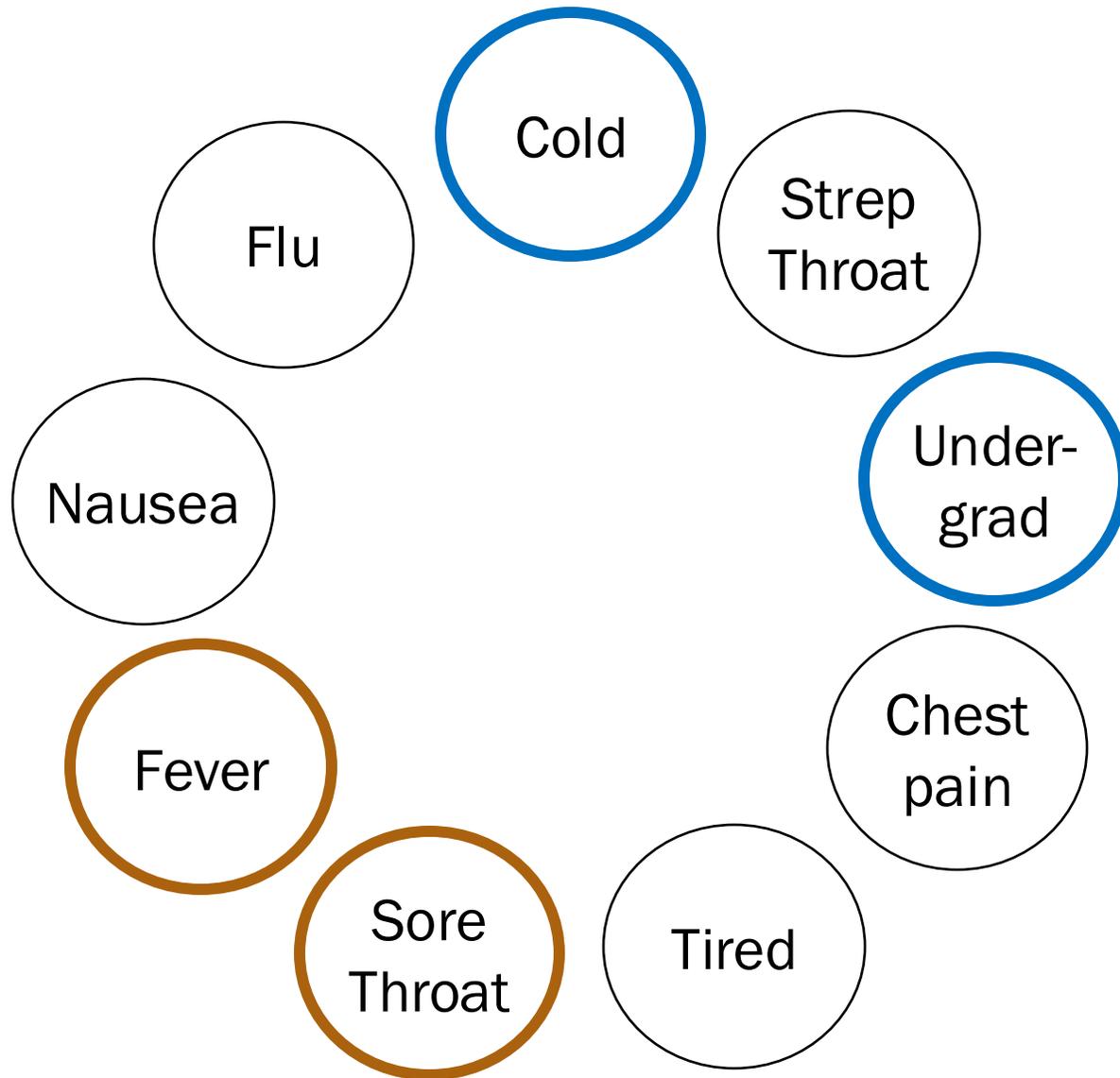
Challenge #1: Many Inference Questions



One inference question:

$$P(F = 1 | N = 1, T = 1) \\ = \frac{P(F = 1, N = 1, T = 1)}{P(N = 1, T = 1)}$$

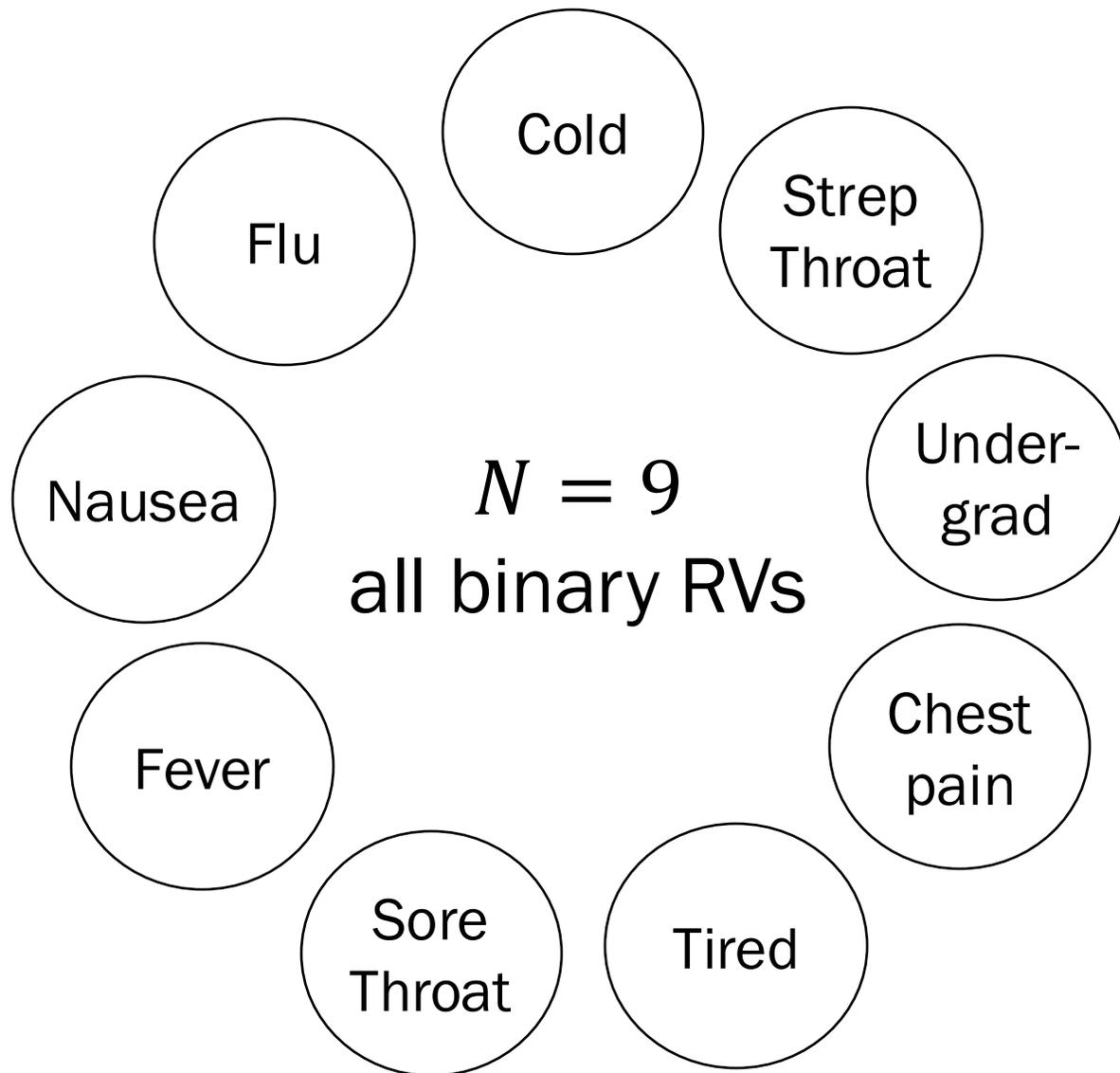
Challenge #1: Many Inference Questions



Another inference question:

$$P(C_o = 1, U = 1 | S = 0, F_e = 0) \\ = \frac{P(C_o = 1, U = 1, S = 0, F_e = 0)}{P(S = 0, F_e = 0)}$$

Challenge #2: Joint is Large



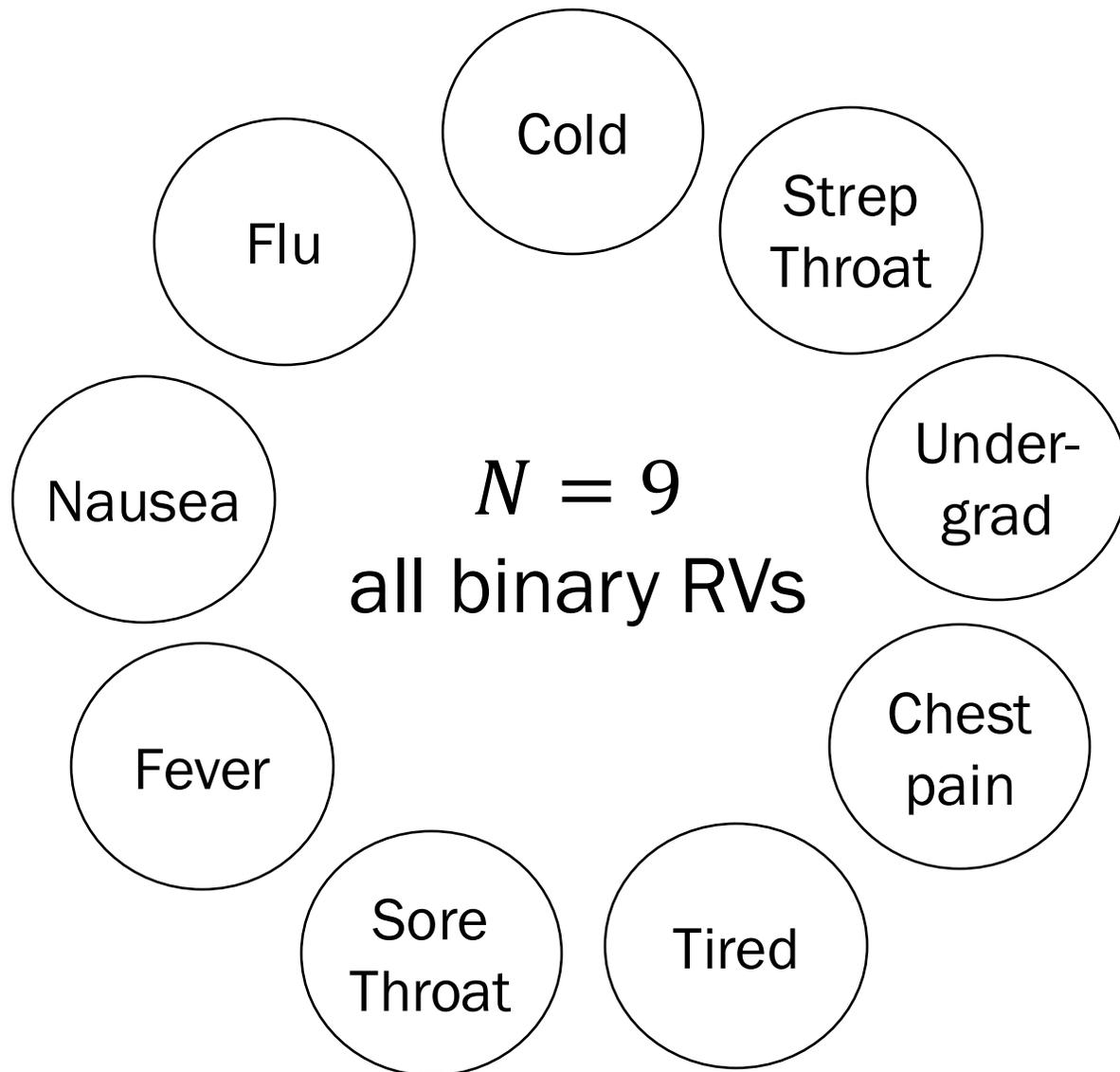
If we knew the **joint distribution**, we can answer all probabilistic inference questions.

What is the size of the joint probability table?

- A. 2^{N-1} entries
- B. N^2 entries
- C. 2^N entries
- D. None/other/don't know



Challenge #2: Joint is Large



If we knew the **joint distribution**, we can answer all probabilistic inference questions.

What is the size of the joint probability table?

- A. 2^{N-1} entries
- B. N^2 entries
- C. 2^N entries**
- D. None/other/don't know

Naively specifying a joint distribution is, in general, intractable.

Three Guiding Questions

1. How do people actually define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

Three Guiding Questions

1. How do people actually define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

Why You Need a Model

*Web*MD[®]

A simpler WebMD

Flu

Under-
grad

Fever

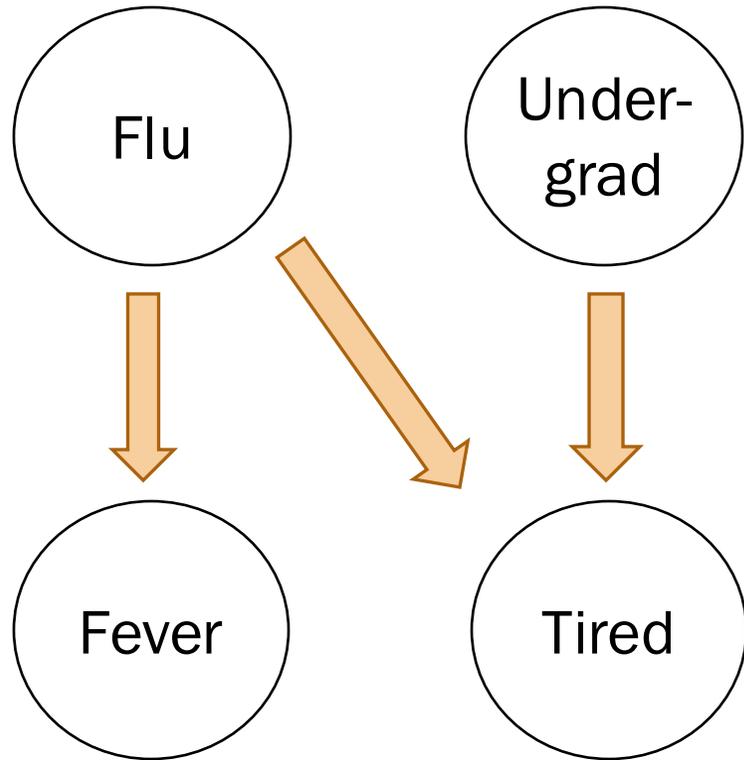
Tired

Great! Just specify $2^4 = 16$ joint probabilities...?

$$P(F_{lu} = a, F_{ev} = b, U = c, T = d)$$

We can compress the joint if we know the generative story...

Constructing a Bayesian Network

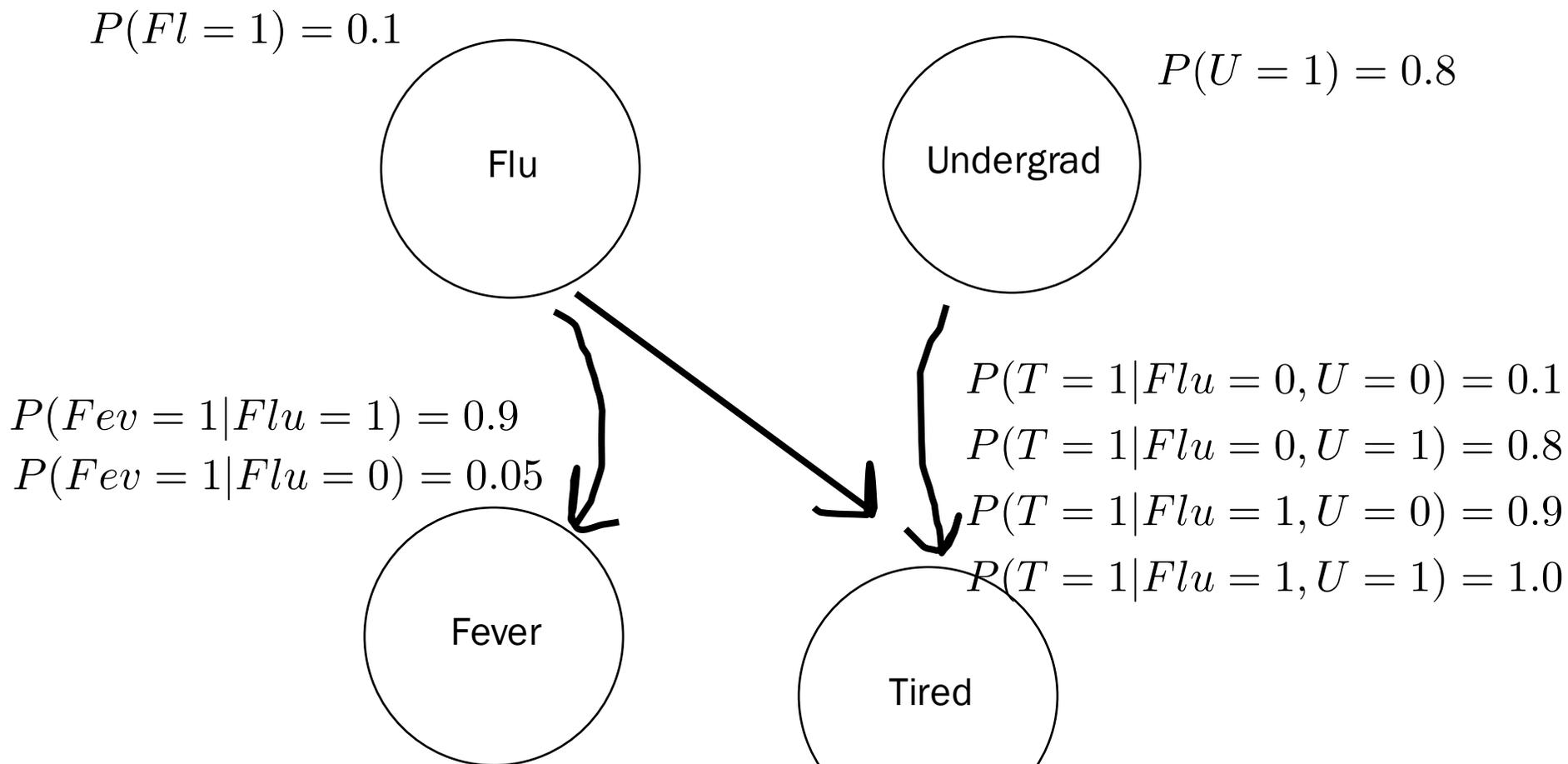


What would a Stanford flu expert do?

- ✓ 1. Describe the causality.
2. Provide $P(\text{values}|\text{causal parents})$ for each random variable
3. Implicitly assumes independences.

Recall: Probabilistic Model

- ✓ 2. Provide $P(\text{values}|\text{causal parents})$ for each random variable



Could we write a python program which makes a fake person from this joint?

To the Code



Midjourney 2023. Prompt: “a lot of excited (non human) pixar characters running off to computers”

```
3 def make_sample():
4     """
5     Make Sample
6     -----
7     chose a single sample from the joint distribution
8     """
9     # prior on causal factors
10    flu = bern(0.1)
11    undergrad = bern(0.8)
12
13    # choose fever based on flue
14    if flu == 1: fever = bern(0.9)
15    else:      fever = bern(0.05)
16
17    # choose tired based on (undergrade and flu)
18    if undergrad == 1 and flu == 1: tired = bern(1.0)
19    elif undergrad == 1 and flu == 0: tired = bern(0.8)
20    elif undergrad == 0 and flu == 1: tired = bern(0.9)
21    else:      tired = bern(0.1)
22
23    # a sample from the joint has an
24    # assignment to *all* random variables
25    return {
26        'flu':flu,
27        'undergrad':undergrad,
28        'fever':fever,
29        'tired':tired
30    }
```

```
3 def make_sample():
4     """
5     Make Sample
6     -----
7     chose a single sample from the joint distribution
8     """
9     # prior on causal factors
10    flu = bern(0.1)
11    undergrad = bern(0.8)
12
13    # choose fever based on flue
14    if flu == 1: fever = bern(0.9)
15    else:      fever = bern(0.05)
16
17    # choose tired based on (undergrade and flu)
18    if undergrad == 1 and flu == 1: tired = bern(1.0)
19    elif undergrad == 1 and flu == 0: tired = bern(0.8)
20    elif undergrad == 0 and flu == 1: tired = bern(0.9)
21    else:      tired = bern(0.1)
22
23    # a sample from the joint has an
24    # assignment to *all* random variables
25    return {
26        'flu':flu,
27        'undergrad':undergrad,
28        'fever':fever,
29        'tired':tired
30    }
```

```
3 def make_sample():
4     """
5     Make Sample
6     -----
7     chose a single sample from the joint distribution
8     """
9     # prior on causal factors
10    flu = bern(0.1)
11    undergrad = bern(0.8)
12
13    # choose fever based on flue
14    if flu == 1: fever = bern(0.9)
15    else:      fever = bern(0.05)
16
17    # choose tired based on (undergrade and flu)
18    if undergrad == 1 and flu == 1: tired = bern(1.0)
19    elif undergrad == 1 and flu == 0: tired = bern(0.8)
20    elif undergrad == 0 and flu == 1: tired = bern(0.9)
21    else:      tired = bern(0.1)
22
23    # a sample from the joint has an
24    # assignment to *all* random variables
25    return {
26        'flu':flu,
27        'undergrad':undergrad,
28        'fever':fever,
29        'tired':tired
30    }
```

```
3 def make_sample():
4     """
5     Make Sample
6     -----
7     chose a single sample from the joint distribution
8     """
9     # prior on causal factors
10    flu = bern(0.1)
11    undergrad = bern(0.8)
12
13    # choose fever based on flue
14    if flu == 1: fever = bern(0.9)
15    else:      fever = bern(0.05)
16
17    # choose tired based on (undergrade and flu)
18    if undergrad == 1 and flu == 1: tired = bern(1.0)
19    elif undergrad == 1 and flu == 0: tired = bern(0.8)
20    elif undergrad == 0 and flu == 1: tired = bern(0.9)
21    else:      tired = bern(0.1)
22
23    # a sample from the joint has an
24    # assignment to *all* random variables
25    return {
26        'flu':flu,
27        'undergrad':undergrad,
28        'fever':fever,
29        'tired':tired
30    }
```

```
3 def make_sample():
4     """
5     Make Sample
6     -----
7     chose a single sample from the joint distribution
8     """
9     # prior on causal factors
10    flu = bern(0.1)
11    undergrad = bern(0.8)
12
13    # choose fever based on flue
14    if flu == 1: fever = bern(0.9)
15    else:      fever = bern(0.05)
16
17    # choose tired based on (undergrade and flu)
18    if undergrad == 1 and flu == 1: tired = bern(1.0)
19    elif undergrad == 1 and flu == 0: tired = bern(0.8)
20    elif undergrad == 0 and flu == 1: tired = bern(0.9)
21    else:      tired = bern(0.1)
22
23    # a sample from the joint has an
24    # assignment to *all* random variables
25    return {
26        'flu':flu,
27        'undergrad':undergrad,
28        'fever':fever,
29        'tired':tired
30    }
```

Can You Sample from the Joint?



Writing a python program that can **sample** from the joint, is the same as defining the joint.

Make a *Generative* Model



A good probabilistic model is **generative**. It explains the process through which the joint is **created**.

Generative Model of Binomial Questions

```
16     })
17
18 class DeclareExpTask(Decision):
19
20     def renderCode(self):
21         explicit = self.getChoice('explicitRv')
22         if explicit:
23             return self.expand('DeclareExplicitExpTask')
24         else:
25             return self.expand('DeclareSubtleExpTask')
26
27
28 TEMPLATES = {
29     'standard': {
30         'template': 'what is the expected number of {successes}',
31         'weight': 5
32     },
33     'v2': {
34         'template': 'what is the expectation of {successes}',
35         'weight': 5
36     },
37     'v3': {
38         'template': 'what is the average number of {successes}',
39         'weight': 2
40     },
41 }
42
43 class DeclareSubtleExpTask(Decision):
44     def registerChoices(self):
45         self.addChoice('expStyle1', gu.makeChoicesFromMap(TEMP
46
47     def renderCode(self):
48         tempVars = {
49             'successes': self.getState('successesStr')
50         }
51         key = self.getChoice('expStyle1')
52         template = TEMPLATES[key]['template']
```

You are flipping a coin 50 times. The probability of a head on each coin-flip is 1/5. What is the probability that the number of heads is 21?

Answer:
Let X be the number of heads.
 $X \sim \text{Bin}(n = 50, p = 1/5)$
 $P(X = 21) = \binom{50}{21} p^{21} (1 - p)^{50 - 21}$

You are trying to mine bitcoins. You try 100 times. The probability of a mining a bit coin on each attempt is 3/25. What is the probability that the number of bitcoins mined is 99?

Answer:
Let X be the number of bitcoins mined.
 $X \sim \text{Bin}(n = 100, p = 3/25)$
 $P(X = 99) = \binom{100}{99} p^{99} (1 - p)^{100 - 99}$

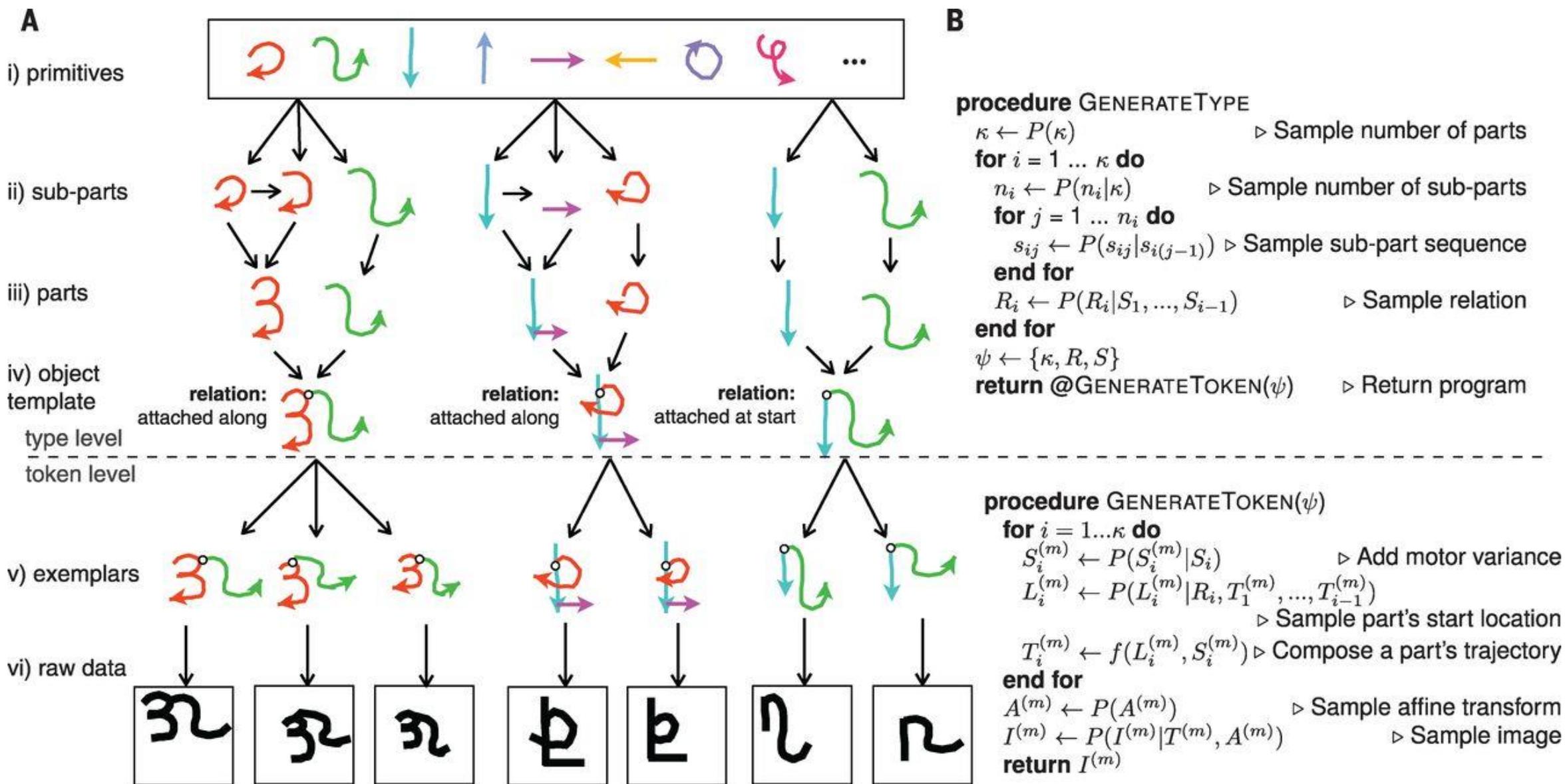
You are running in an election. The number of votes for you can be represented by a random variable X. $X \sim \text{Bin}(n = 100, p = 1/20)$. What is the probability that X is equal to 6?

Answer:
Let X be the number of votes for you.
 $X \sim \text{Bin}(n = 100, p = 1/20)$
 $P(X = 6) = \binom{100}{6} p^6 (1 - p)^{100 - 6}$

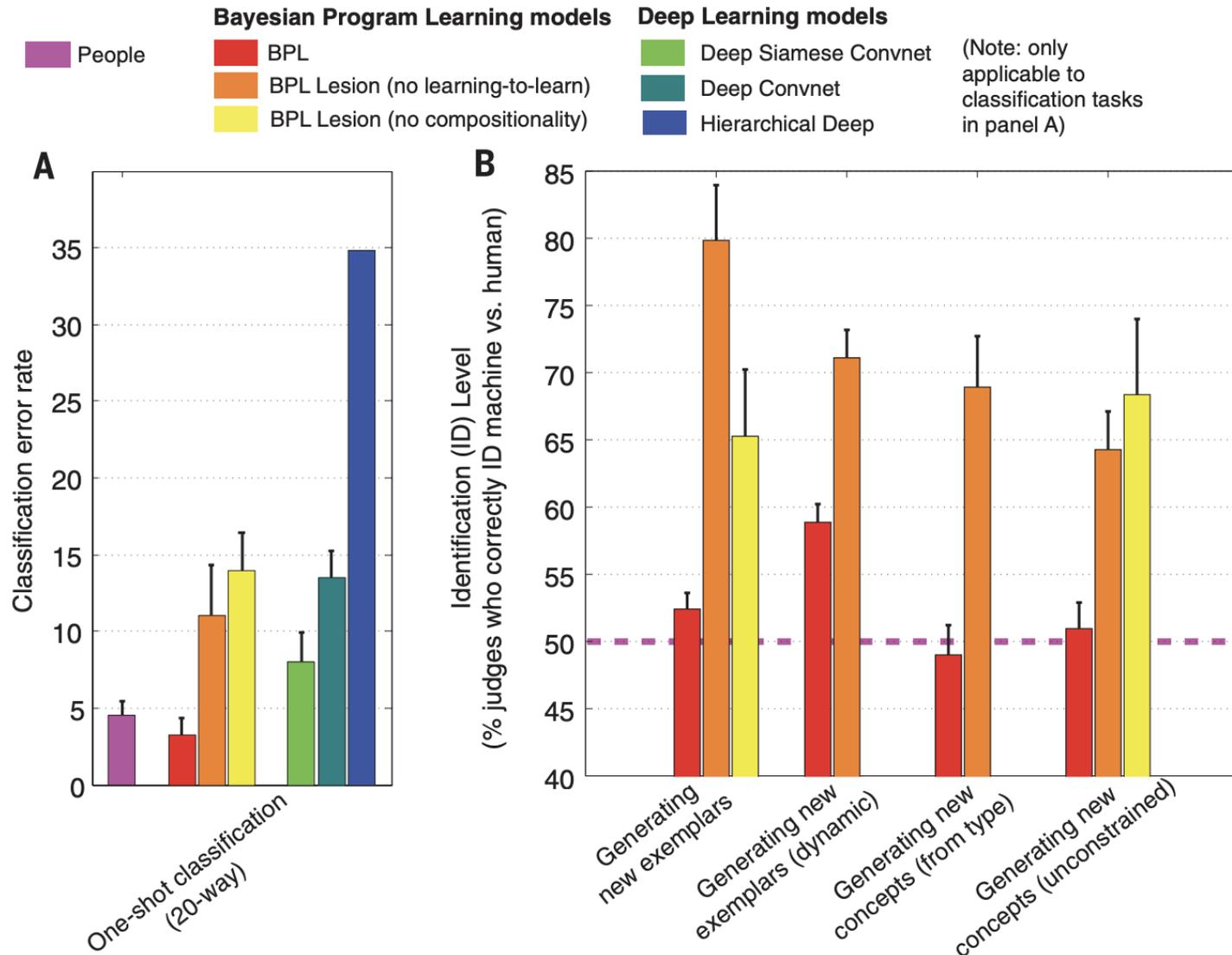
A ball hits a series of 10 pins where it can bounce either right or left. The probability of a right on each pin hit is 0.5. What is the probability that the number of rights is greater than 7?

Answer:
Let X be the number of rights.
 $X \sim \text{Bin}(n = 10, p = 0.5)$
 $P(X > 7) = P(8 \leq X \leq 10)$
 $= \sum_{i=8}^{10} \binom{10}{i} p^i (1 - p)^{10 - i}$

Generative Model of Hand Written Letters

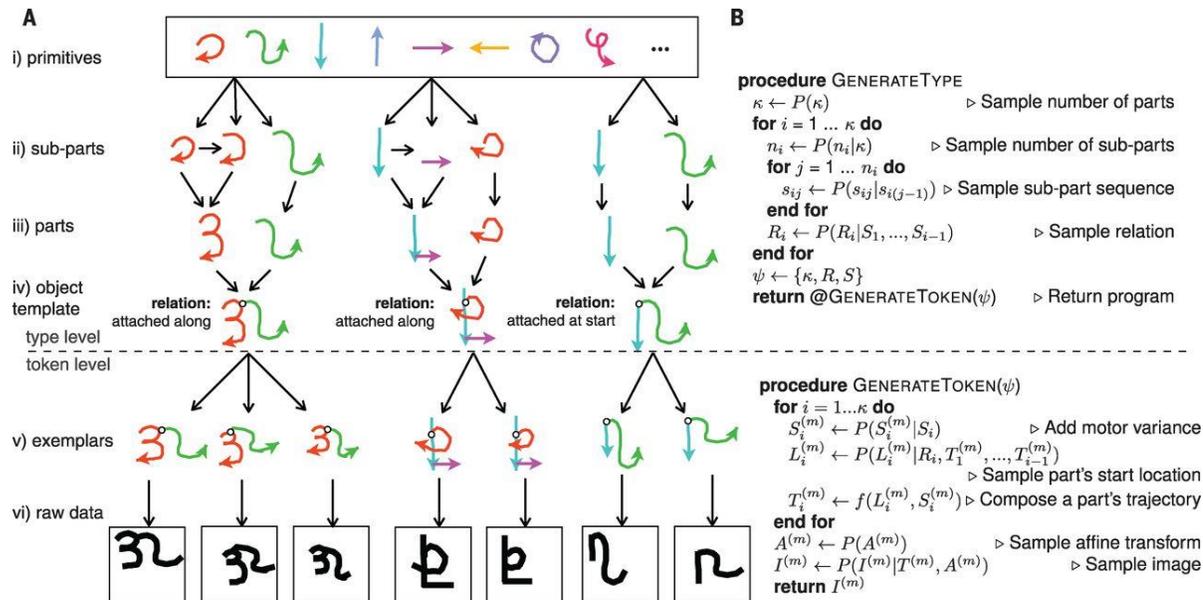


Human Level. And More!

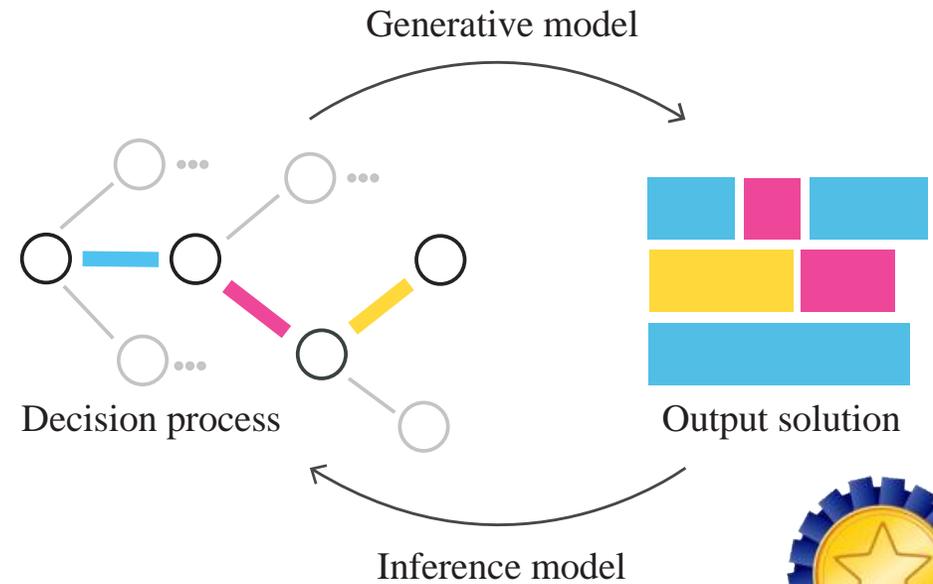


Generative Model of Grading

Lake et al, 2015



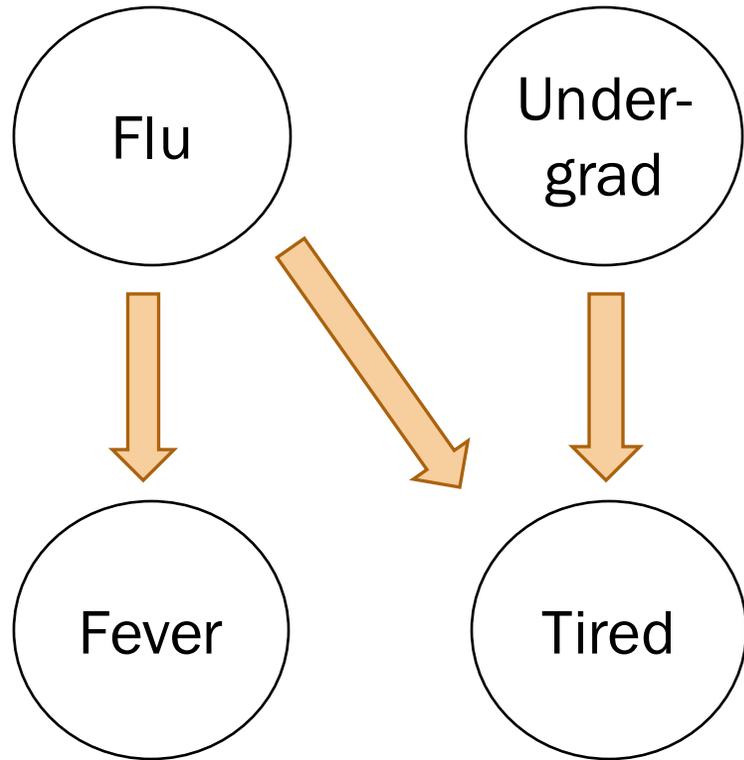
Muke Wu, Ali Malik, Noah Goodman, Chris Piech, 2019



Outstanding
Paper Award, AAAI 2019

Stanford University

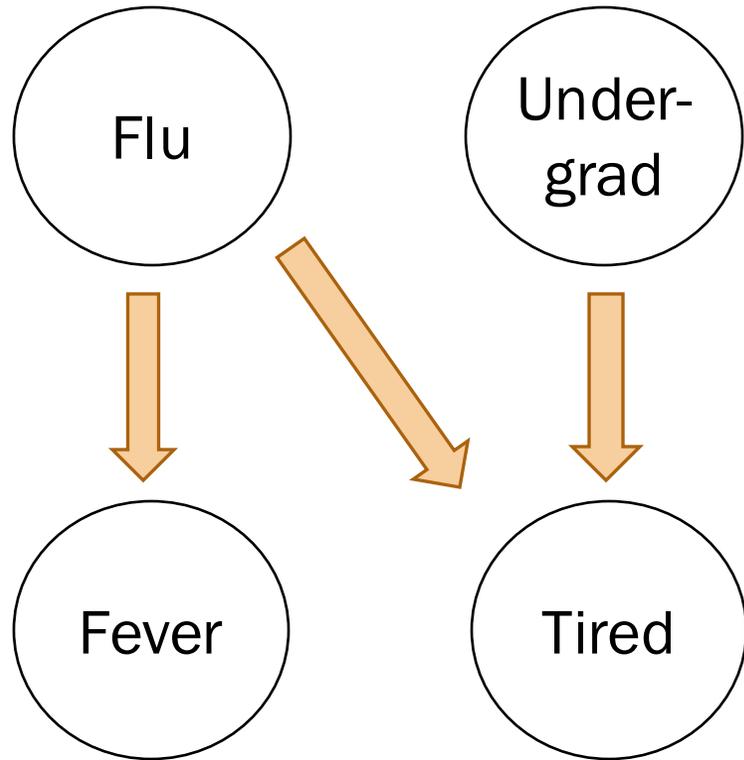
Generative Models make Independence Assumptions



What would a Stanford flu expert do?

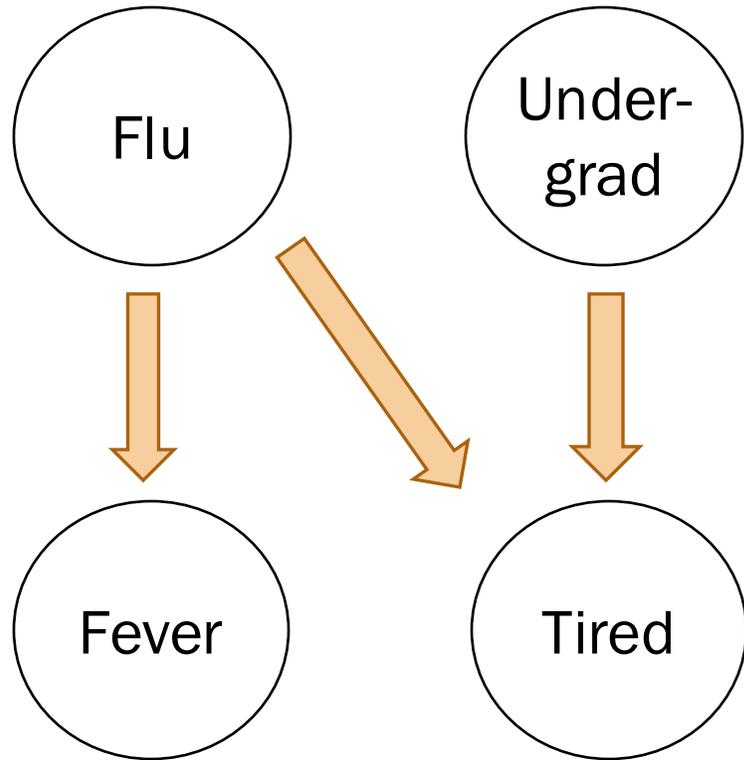
- ✓ 1. Describe the causality.
- ✓ 2. Provide $P(\text{values}|\text{causal parents})$ for each random variable
3. Implicitly assumes independences.

Generative Models make Independence Assumptions



Each random variable is **conditionally independent** of its causal non-descendants, **given its causal parents**.

Generative Models make Independence Assumptions

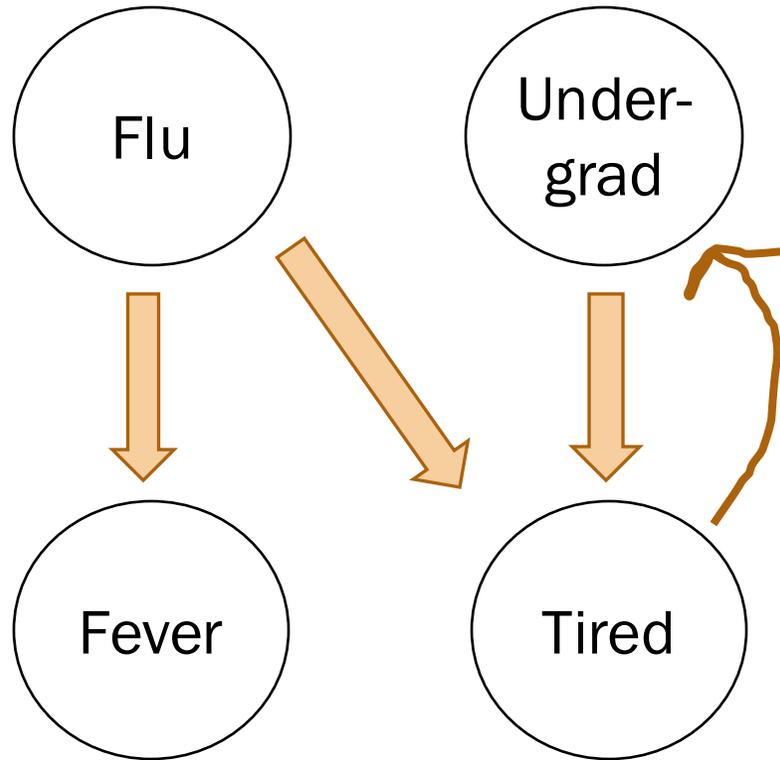


This model assumes that Flu and being an Undergraduate are independent.

Advanced: it also assumes that fever and tired are conditionally independent given Flu.

You need to tell a generative story. The independence assumptions come for free.

Bug: Constructing a Bayesian Network



Must be acyclic!

Three Guiding Questions

1. How do people define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

Three Guiding Questions

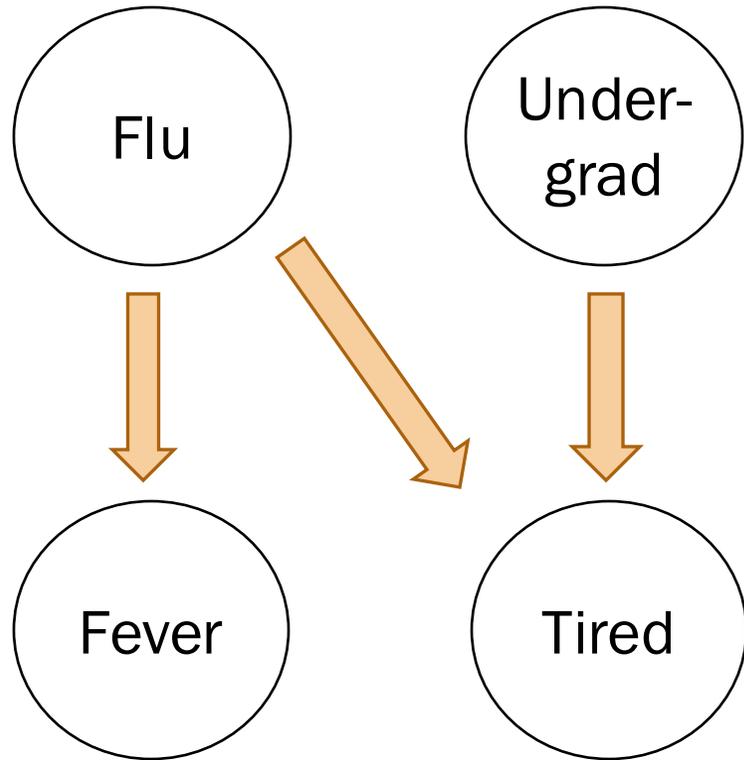
1. How do people define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

First, the hard way

Algorithm #1: Inference via math

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



1. $P(F_{lu} = 0, U = 1, F_{ev} = 0, T = 1)$?

$$= P(F_{lu} = 0)$$

$$\cdot P(U = 1 | F_{lu} = 0)$$

$$\cdot P(F_{ev} = 0 | F_{lu} = 0, U = 1)$$

$$\cdot P(T = 1 | F_{ev} = 0, F_{lu} = 0, U = 1)$$

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

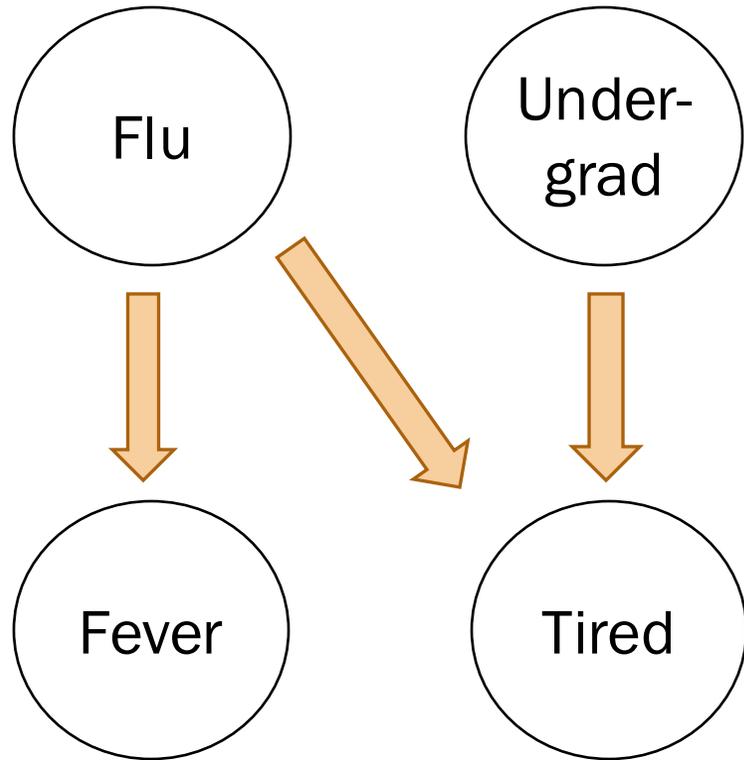
$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

Algorithm #1: Inference via math

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



1. $P(F_{lu} = 0, U = 1, F_{ev} = 0, T = 1)$?

$$= P(F_{lu} = 0)$$

$$\cdot P(U = 1)$$

$$\cdot P(F_{ev} = 0 | F_{lu} = 0, U = 1)$$

$$\cdot P(T = 1 | F_{ev} = 0, F_{lu} = 0, U = 1)$$

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

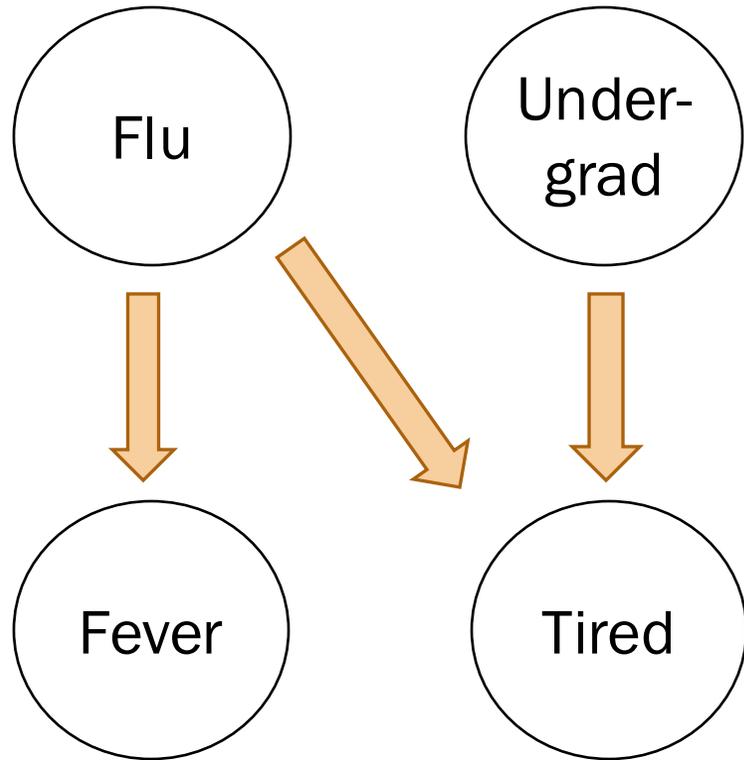
$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

Algorithm #1: Inference via math

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



1. $P(F_{lu} = 0, U = 1, F_{ev} = 0, T = 1)$?

$$= P(F_{lu} = 0)$$

$$\cdot P(U = 1)$$

$$\cdot P(F_{ev} = 0 | F_{lu} = 0)$$

$$\cdot P(T = 1 | F_{ev} = 0, F_{lu} = 0, U = 1)$$

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

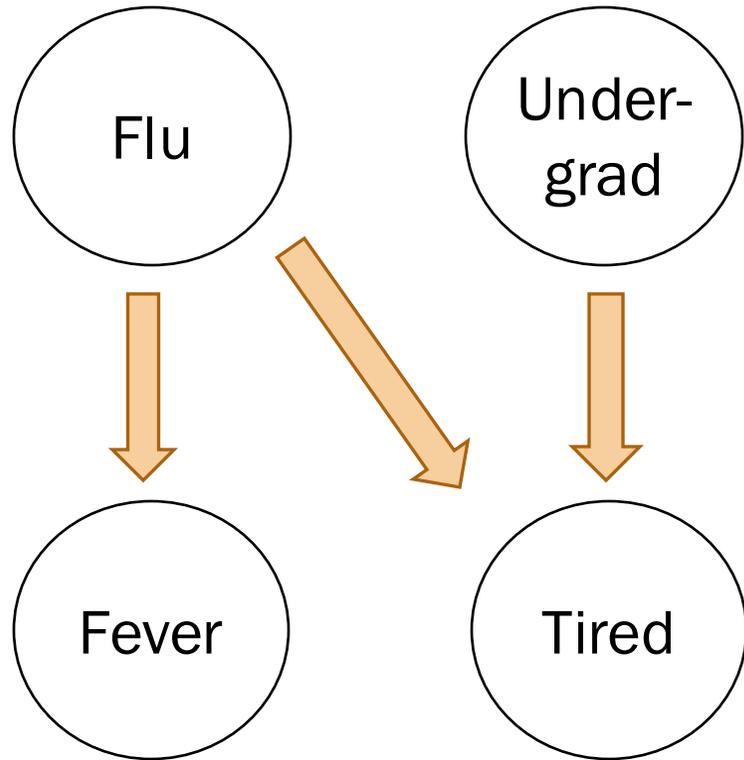
$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

Algorithm #1: Inference via math

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



1. $P(F_{lu} = 0, U = 1, F_{ev} = 0, T = 1)$?

$$= P(F_{lu} = 0)$$

$$\cdot P(U = 1)$$

$$\cdot P(F_{ev} = 0 | F_{lu} = 0)$$

$$\cdot P(T = 1 | F_{lu} = 0, U = 1)$$

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

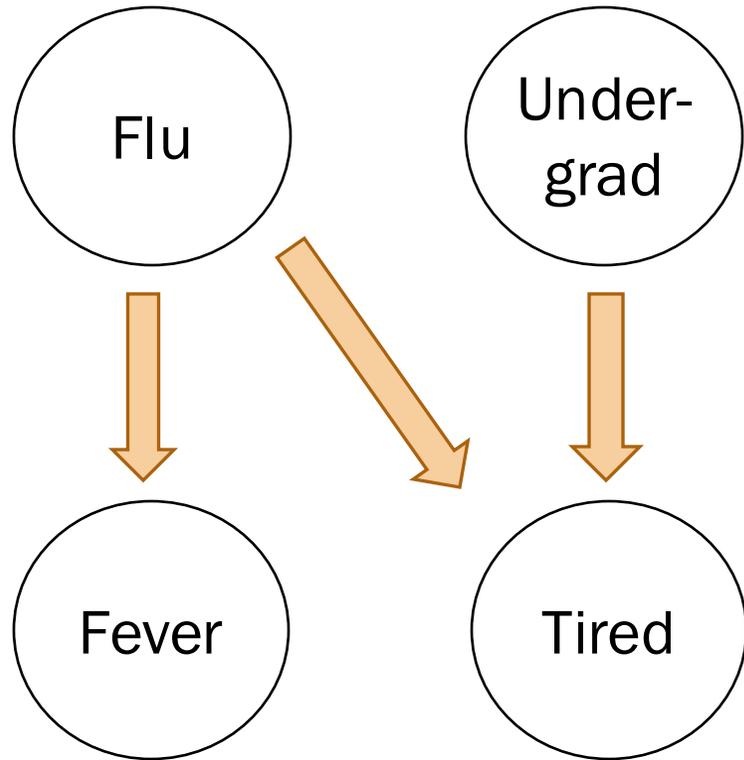
$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

Algorithm #1: Inference via math

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



1. $P(F_{lu} = 0, U = 1, F_{ev} = 0, T = 1)$?

$$= P(F_{lu} = 0) \quad 0.9$$

$$\cdot P(U = 1) \quad 0.2$$

$$\cdot P(F_{ev} = 0 | F_{lu} = 0) \quad 0.05$$

$$\cdot P(T = 1 | F_{lu} = 0, U = 1) \quad 0.8$$

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

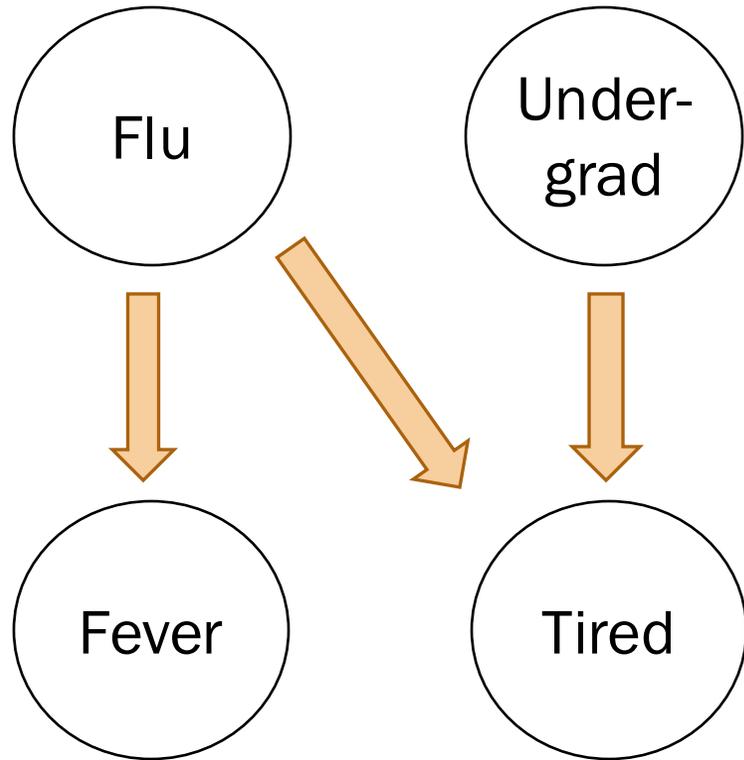
$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

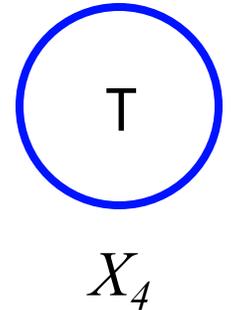
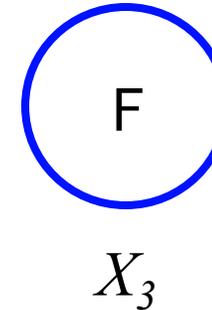
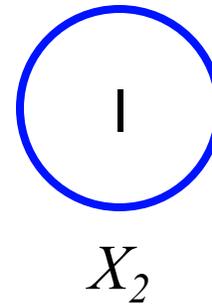
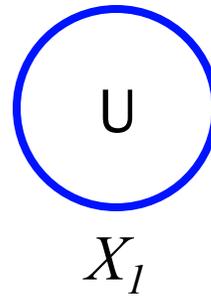
$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

Algorithm #1: Inference via math



Order nodes by ancestry



$$P(\text{Joint}) = \prod_i P(x_i | x_{i-1}, \dots, x_1)$$

←

$$= \prod_i P(x_i | \text{Values of parents of } X_i)$$

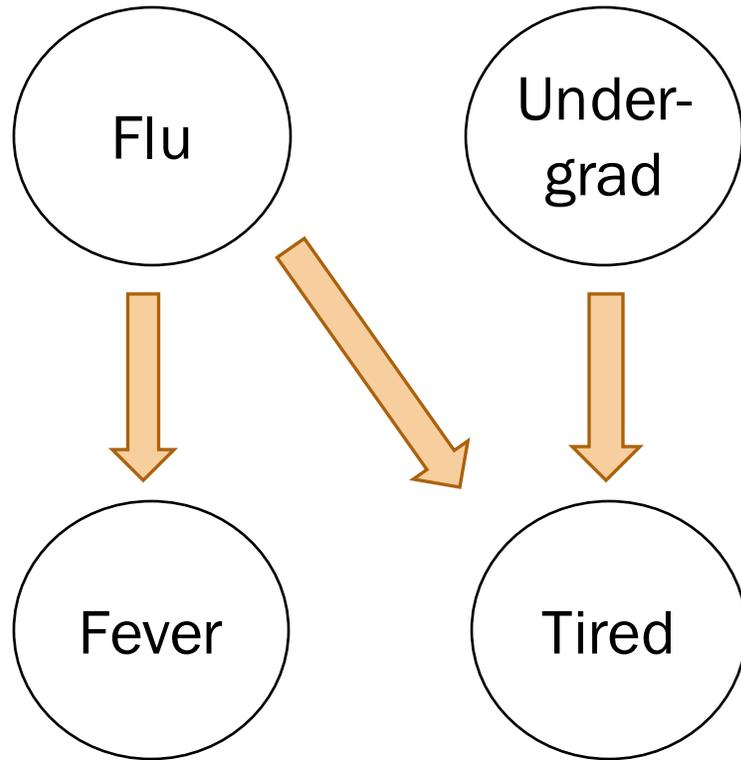
Each random variable is **conditionally independent** of its causal non-descendants, **given its causal parents**.

Inference via math

$$P(\text{Joint}) = \prod_i P(x_i | \text{Values of parents of } X_i)$$

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



2. $P(F_{lu} = 1 | F_{ev} = 0, U = 0, T = 1)$?

1. Compute joint probabilities

$$P(F_{lu} = 1, F_{ev} = 0, U = 0, T = 1)$$

$$P(F_{lu} = 0, F_{ev} = 0, U = 0, T = 1)$$

2. Definition of conditional probability

$$\frac{P(F_{lu} = 1, F_{ev} = 0, U = 0, T = 1)}{\sum_x P(F_{lu} = x, F_{ev} = 0, U = 0, T = 1)}$$

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

$$= 0.095$$

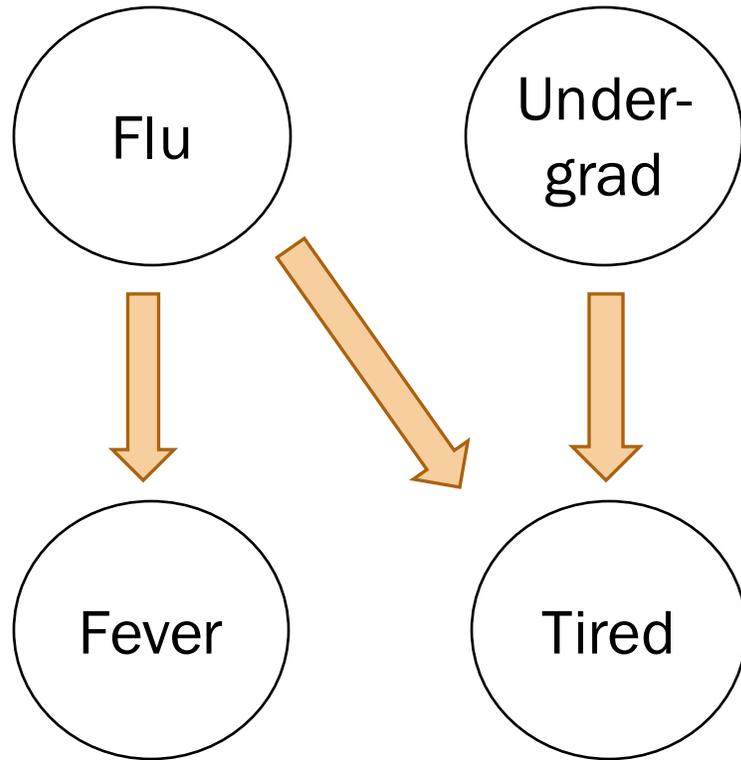
Want more practice?

Inference via math

$$P(\text{Joint}) = \prod_i P(x_i | \text{Values of parents of } X_i)$$

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$

$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$

$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$

$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$

$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$

3. $P(F_{lu} = 1 | U = 1, T = 1)$?

1. Compute joint probabilities

$$P(F_{lu} = 1, U = 1, F_{ev} = 1, T = 1)$$

...

$$P(F_{lu} = 0, U = 1, F_{ev} = 0, T = 1)$$

2. Definition of conditional probability

$$\frac{\sum_y P(F_{lu} = 1, U = 1, F_{ev} = y, T = 1)}{\sum_x \sum_y P(F_{lu} = x, U = 1, F_{ev} = y, T = 1)}$$

$$= 0.122$$

There must be a better way...

[suspense]

Algorithm #2: Rejection Sampling

```
13  def main():
14      obs = get_observation()
15      samples = sample_a_ton()
16      prob = prob_flu_given_obs(samples, obs)
17      print('Observation = ', obs)
18      print('Pr(Flu | Obs) = ', prob)
```

Algorithm #2: Rejection Sampling

```
13 def main():
14     obs = get_observation()
15     samples = sample_a_ton()
16     prob = prob_flu_given_obs(samples)
17     print('Observation = ', obs)
18     print('Pr(Flu | Obs) = ', prob)
```

```
webMd --zsh -- 56x42
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
```

Algorithm #2: Rejection Sampling

```
13  def main():
14      obs = get_observation()
15      samples = sample_a_ton()
16      prob = prob_flu_given_obs(samples, obs)
17      print('Observation = ', obs)
18      print('Pr(Flu | Obs) = ', prob)
```

Algorithm #2: Rejection Sampling

```
35 def probab_flu_given_obs(samples, obs):
36     """
37     Calculate the probability of flu given many
38     samples from the joint distribution and a set
39     of observations to condition on.
40     """
41     # reject all samples which don't align
42     # with condition
43     keep_samples = []
44     for sample in samples:
45         if check_obs_match(sample, obs):
46             keep_samples.append(sample)
47
48     # from remaining, simply count...
49     flu_count = 0
50     for sample in keep_samples:
51         if sample['flu'] == 1:
52             flu_count += 1
53
54     # counting can be so sweet...
55     return float(flu_count) / len(keep_samples)
```

Algorithm #2: Rejection Sampling

```
35 def prob_flu_given_obs(samples, obs):
36     """
37     Calculate the probability of flu given many
38     samples from the joint distribution and a set
39     of observations to condition on.
40     """
41     # reject all samples which don't align
42     # with condition
43     keep_samples = []
44     for sample in samples:
45         if check_obs_match(sample, obs):
46             keep_samples.append(sample)
47
48     # from remaining, simply count...
49     flu_count = 0
50     for sample in keep_samples:
51         if sample['flu'] == 1:
52             flu_count += 1
53
54     # counting can be so sweet...
55     return float(flu_count) / len(keep_samples)
```

Algorithm #2: Rejection Sampling

```
35 def prob_flu_given_obs(samples, obs):
36     """
37     Calculate the probability of flu given many
38     samples from the joint distribution and a set
39     of observations to condition on.
40     """
41     # reject all samples which don't align
42     # with condition
43     keep_samples = []
44     for sample in samples:
45         if check_obs_match(sample, obs):
46             keep_samples.append(sample)
47
48     # from remaining, simply count...
49     flu_count = 0
50     for sample in keep_samples:
51         if sample['flu'] == 1:
52             flu_count += 1
53
54     # counting can be so sweet...
55     return float(flu_count) / len(keep_samples)
```

Algorithm #2: Rejection Sampling

```
35 def prob_flu_given_obs(samples, obs):
36     """
37     Calculate the probability of flu given m
38     samples from the joint distribution and
39     of observations to condition on.
40     """
41     # reject all samples which don't align
42     # with condition
43     keep_samples = []
44     for sample in samples:
45         if check_obs_match(sample, obs):
46             keep_samples.append(sample)
47
48     # from remaining, simply count...
49     flu_count = 0
50     for sample in keep_samples:
51         if sample['flu'] == 1:
52             flu_count += 1
53
54     # counting can be so sweet...
55     return float(flu_count) / len(keep_samples)
```

```
webMd --zsh -- 53x25
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
-----
Observation = {'flu': None, 'undergrad': 1, 'fever':
None, 'tired': 1}
Pr(Flu | Obs) = 0.1228646517739816
piech@Chriss-MBP-5 webMd %
```

Lets try it!

BACK ←
TO **CODE**
THE

Rejection sampling algorithm

Inference question: What is $P(F_{lu} = 1 | U = 1, T = 1)$?

$$\text{probability} \approx \frac{\# \text{ samples with } (F_{lu} = 1, U = 1, T = 1)}{\# \text{ samples with } (U = 1, T = 1)}$$

Why would this definition of approximate probability make sense?



Why would this approximate probability make sense?

Inference question: What is $P(F_{lu} = 1 | U = 1, T = 1)$?

$$\text{probability} \approx \frac{\# \text{ samples with } (F_{lu} = 1, U = 1, T = 1)}{\# \text{ samples with } (U = 1, T = 1)}$$

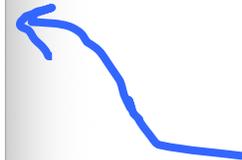
Recall our definition of probability as a frequency: $P(E) = \lim_{n \rightarrow \infty} \frac{n(E)}{n}$ $n = \#$ of total trials
 $n(E) = \#$ trials where E occurs



```
webMd --zsh -- 53x25
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 1, 'undergrad': 1, 'fever': 1, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 0, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 1}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
{'flu': 0, 'undergrad': 1, 'fever': 0, 'tired': 0}
-----
Observation = {'flu': None, 'undergrad': 1, 'fever':
None, 'tired': 1}
Pr(Flu | Obs) = 0.1228646517739816
piech@Chriss-MBP-5 webMd %
```



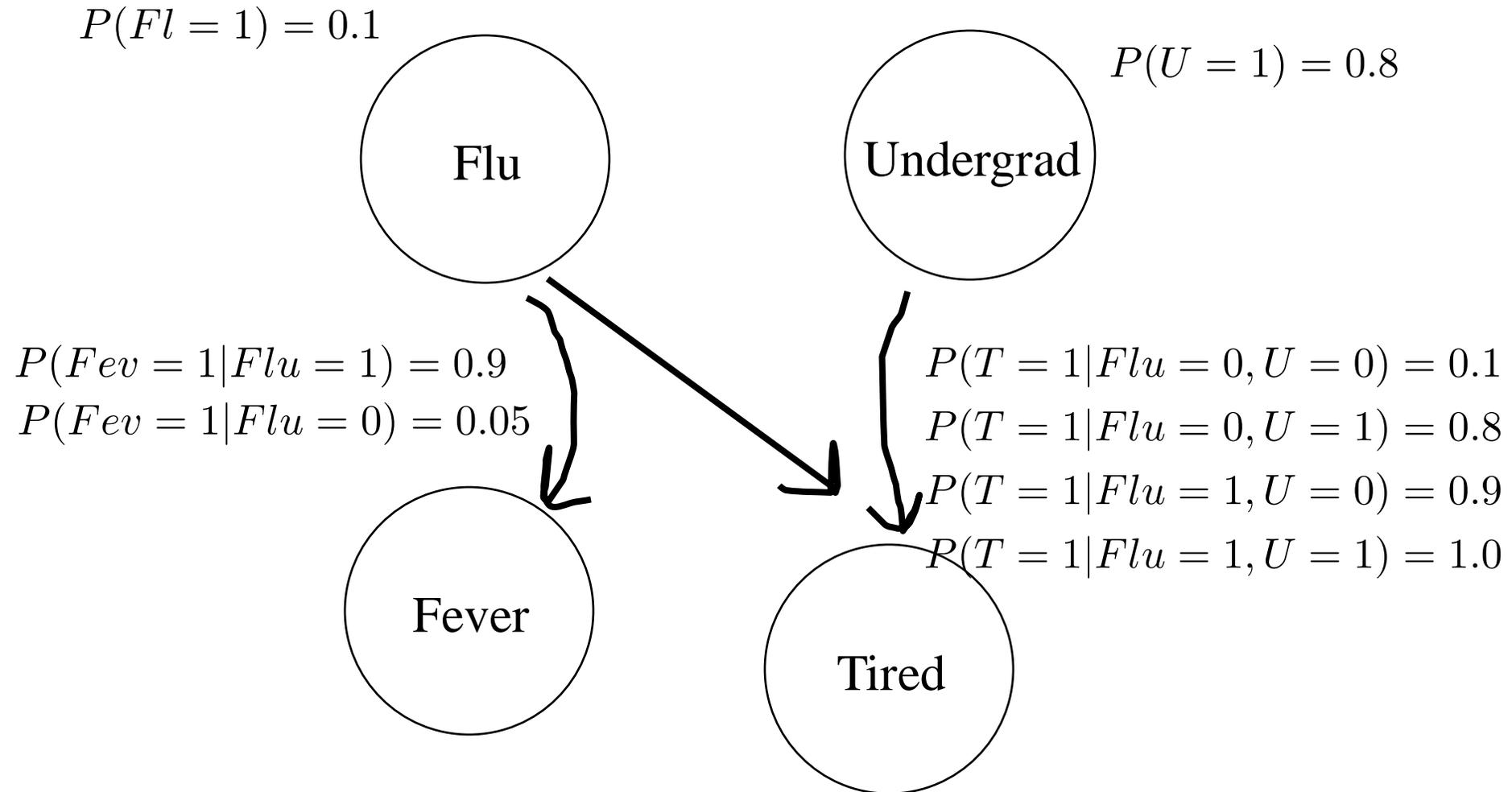
If you can sample enough from the joint distribution, you can answer any probability question



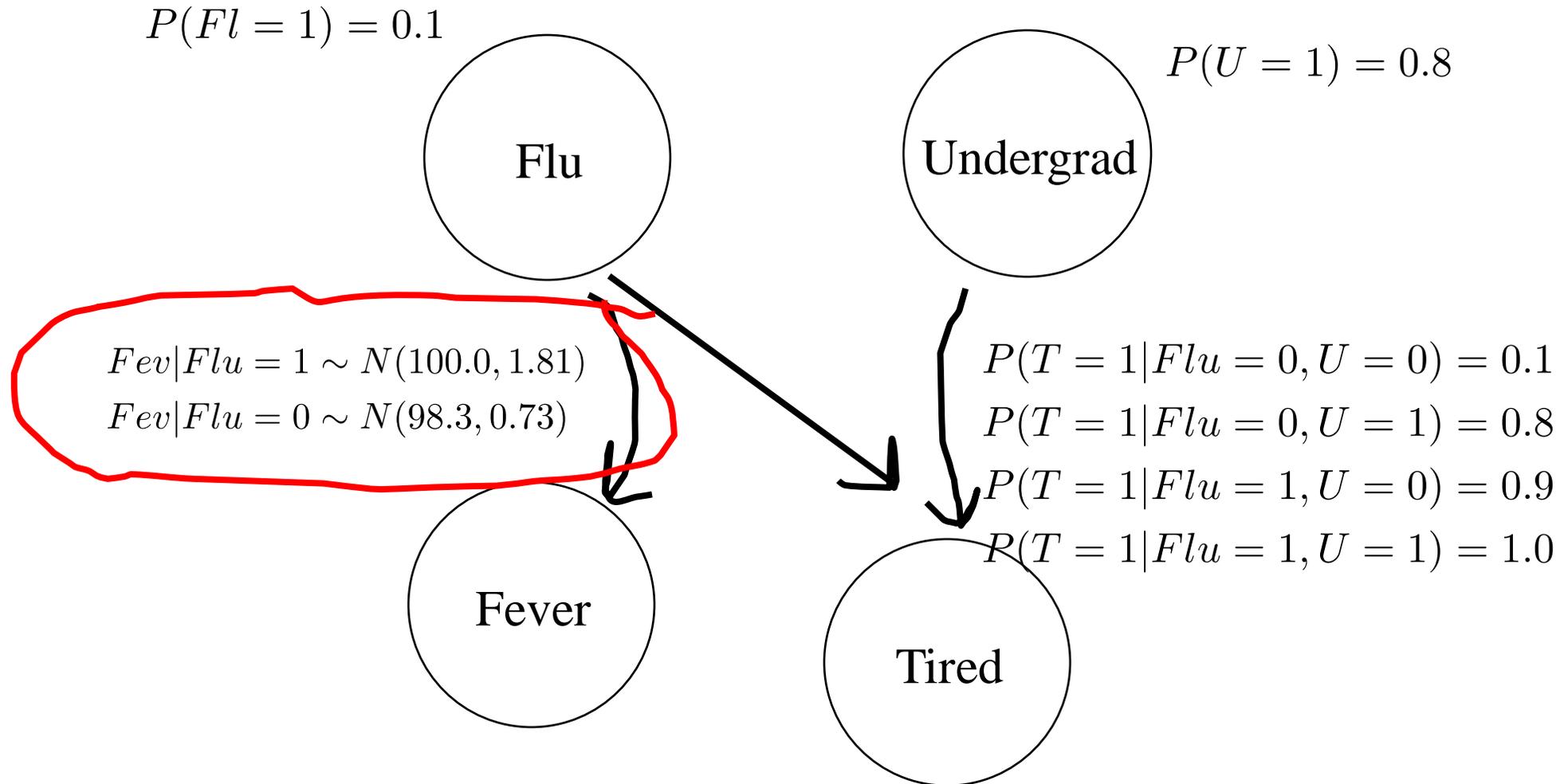
Each one of these is one joint sample

What's the matter with
rejection sampling?

Probabilistic Model



Probabilistic Model



The Magic School Bus™

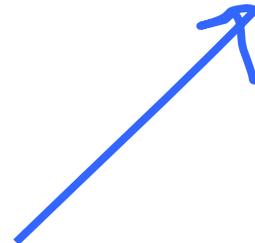


Many Algorithms

Markov Chain



MCMC



Monte Carlo

Many Algorithms

```
webmd -- -bash -- 20x20
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[0, 1, 101.0, 0]
[0, 0, 101.0, 0]
[1, 0, 101.0, 1]
[1, 0, 101.0, 0]
[1, 0, 101.0, 1]
[1, 0, 101.0, 1]
[1, 0, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
[1, 0, 101.0, 1]
[1, 1, 101.0, 1]
[1, 1, 101.0, 1]
Pr(Flu) = 0.9773
>
```

MCMC is a way to sample
with conditioned variables
fixed

Each one of these
is one posterior
sample:

[Flu, Undergrad, Fever, Tired]



Many Algorithms

Rejection
Sampling



MCMC



Pyro



Idea2Text



Three Guiding Questions

1. How do people define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

Three Guiding Questions

1. How do people define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

ROCK

The Sound: Vigorous, defiant, energetic, inventive

The Roots: Rhythm & blues, country

The Pioneers: Bill Haley, Chuck Berry, Fats Domino, Little Richard, Buddy Holly, Elvis Presley

The Places: Cleveland, New Orleans, Detroit, New York City

The Ensemble: Electric guitar, bass, drums, keyboard, vocals

"We're a rock group. We're noisy, rowdy, emotional and wild."

— Roger Young Jr., 1955
Lead guitarist of rock band AC/DC

HIP-HOP RAP

The Sound: Rhythmic, unvarnished, adaptable, streetwise

The Roots: Rhythm & blues, soul, funk, reggae

The Pioneers: Afrika Bambaataa, Kool Herc, DJ Hollywood, Grandmaster Flash, Kurtis Blow, Grandmaster Caz

The Places: New York City (South Bronx)

The Ensemble: Vinyl, turntable, vocals

"The beautiful thing about hip-hop is it's the one music collage. You can take any form of music and do it in a hip-hop way and it'll be a hip-hop song."

— Tom Tompkins, 1983
MC of Run-DMC

LATIN American

The Sound: Syncopated, enthusiastic, diverse, vibrant

The Roots: Spain, Africa, Caribbean, South America

The Pioneers: Arsenio Rodriguez, Machito, Pérez Prado, Tito Puente, Celia Cruz, Johnny Pacheco

The Places: Cuba, Puerto Rico, Mexico, Miami, New York

The Ensemble: Congas, bongos, maracas, güiro, guitar, vocals

"The emphasis was dancing and rhythm. I came in with an emphasis on lyrics, telling stories that were familiar to people in Latin America—and everybody thought with the songs."

— Roberto Roena Jr., 1980
Lead singer of the group

Folk

The Sound: Grassroots, narrative, sincere, lyrical

The Roots: Ballads, immigrant folklore, spirituals, cowboy songs

The Pioneers: Lead Belly, Odetta, Woody Guthrie, Pete Seeger, Bob Dylan, Joan Baez

The Places: Appalachia, Deep South, Western frontier

The Ensemble: Guitar, banjo, fiddle, accordion, vocals

"I find the rhythms [of folk music], I find the melodies, time tested by generations of singers. Above all, I find the words... they seemed fresh, straightforward, honest."

— Peter Dinklage Jr., 1960
Folk musician

COUNTRY Western

The Sound: Genuine, uncomplicated, nostalgic, informal

The Roots: European ballads, folk and gospel songs

The Pioneers: Uncle Dave Macon, the Carter Family, Jimmie Rodgers, Roy Acuff, Gene Autry, Bill Monroe

The Places: Appalachia, Nashville, Chicago, Western U.S.

The Ensemble: Fiddle, banjo, guitar, harmonica, accordion, vocals

"Country music is very crude and the best."

— Hank Williams (1917–1953)
Country musician

CLASSICAL

The Sound: Intricate, polished, structured, harmonious

The Roots: Sacred music, choral chants, madrigals, dance rhythms

The Pioneers: J.S. Bach, Handel, Haydn, Mozart, Beethoven, Brahms

The Places: Austria, Germany, France, Italy

The Ensemble: Strings, woodwinds, brass, percussion, vocals

"I carry my thoughts about with me a long time... before writing them down. I change many things, discard others, and try again and again until I am satisfied."

— Ludwig van Beethoven (1770–1827)
Composer

AutoSave OFF | Search Sheet | Home | Insert | Page Layout | Formulas | Data | Share

Clipboard | Font | Alignment | Number | Conditional Formatting | Format as Table | Cell Styles | Cells | Editing

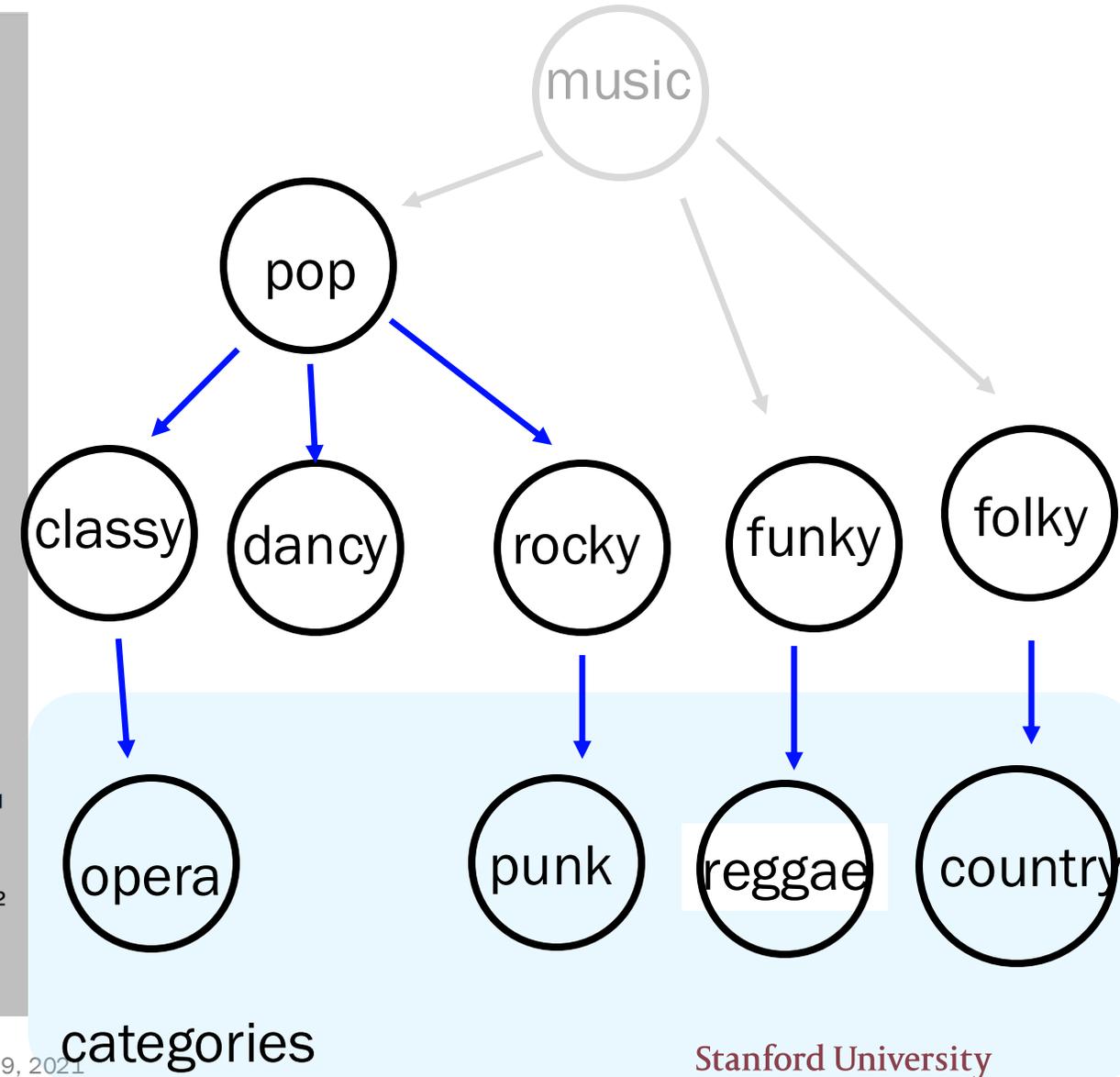
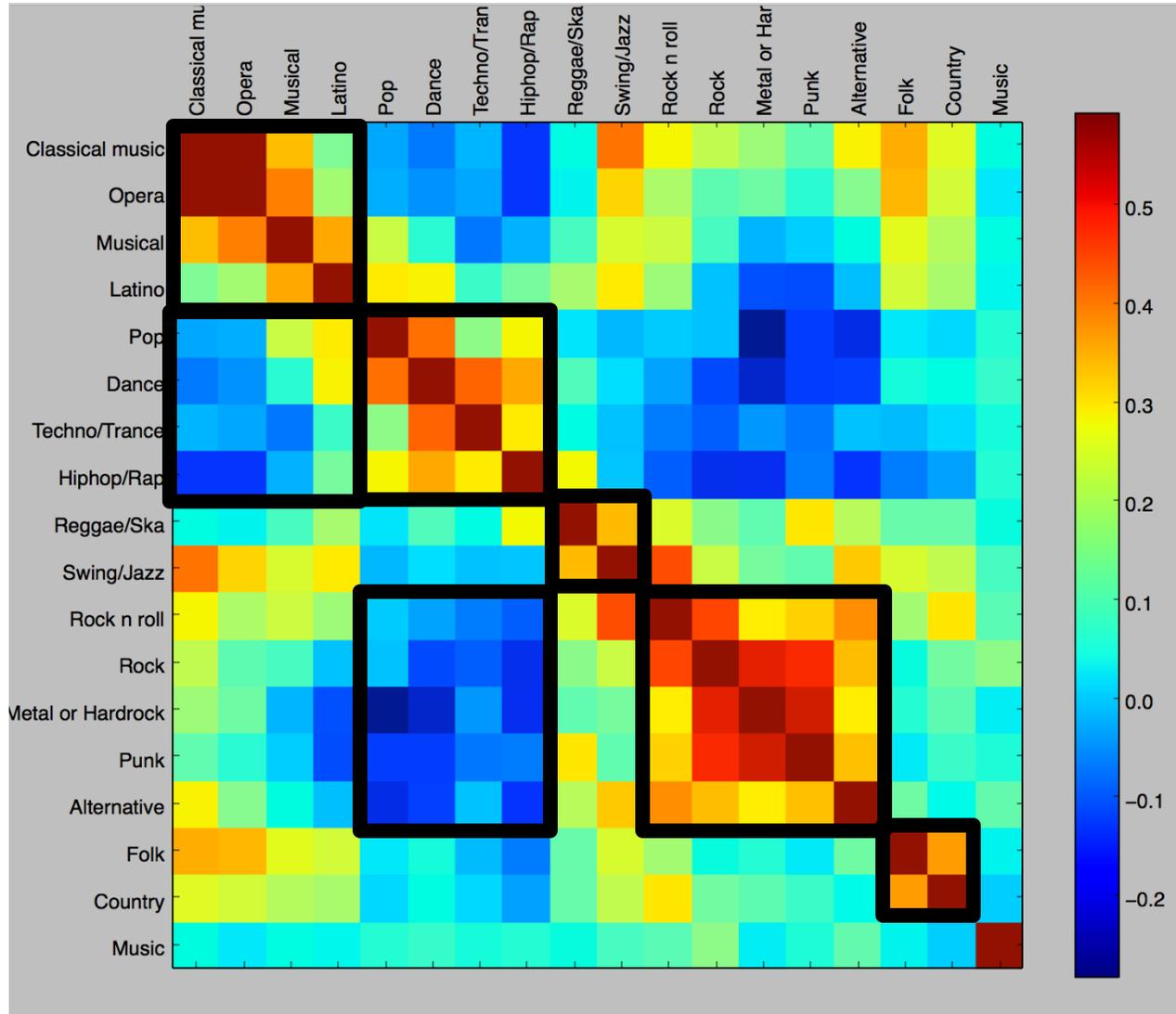
C15 | fx | 3

	A	B	C	D	E	F	G	H	I
1	Music	Dance	Folk	Country	Classical music	Musical	Pop	Rock	Me
2	5	2	1	2	2	1	5	5	
3	4	2	1	1	1	2	3	5	
4	5	2	2	3	4	5	3	5	
5	5	2	1	1	1	1	2	2	
6	5	4	3	2	4	3	5	3	
7	5	2	3	2	3	3	2	5	
8	5	5	3	1	2	2	5	3	
9	5	3	2	1	2	2	4	5	
10	5	3	1	1	2	4	3	5	
11	5	2	5	2	2	5	3	5	
12	5	3	2	1	2	3	4	3	
13	5	1	1	1	4	1	2	5	
14	5	1	2	1	4	3	3	5	
15	5	5	3	2	1	5	5	2	
16	5	2	1	1	2	3	4	5	
17	1	2	2	3	4	3	3	5	
18	5	3	1	1	1	2	4	4	
19	5	3	3	3	2	2	4	4	
20	5	5	4	3	4	5	5	4	
21	5	3	3	2	4	2	2	4	
22	5	3	2	3	4	3	2	5	
23	5	1	1	3	2	2	2	5	
24	5	3	2	3	3	3	4		
25	5	4	2	2	2	4	4	5	
26	5	3	1	1	4	3	3	5	
27	5	4	2	1	2	3	5	1	
28	5	5	5	4	5	3	4	4	
29	4	3	4	1	3	2	2	4	
30	5	5	1	1	1	1	3	4	
31	5	3	4	2	3	3	3	4	
32	4	4	3	3	3	3	4	4	
33	4	4	1	3	2	3	5	3	
34	5	3	1	3	2	3	3	4	
35	5	2	2	3	4	5	4	3	

music +

Ready | 100%

From Correlation to Bayes Net!



Why is it harder to find independences here than for bat DNA expression?

	A	B	C	D	E	F	G	H	Me
1	Music	Dance	Folk	Country	Classical music	Musical	Pop	Rock	Me
2	5	2	1	2	2	1	5	5	
3	4	2	1	1	1	2	3	5	
4	5	2	2	3	4	5	3	5	
5	5	2	1	1	1	1	2	2	
6	5	4	3	2	4	3	5	3	
7	5	2	3	2	3	3	2	5	
8	5	5	3	1	2	2	5	3	
9	5	3	2	1	2	2	4	5	
10	5	3	1	1	2	4	3	5	
11	5	2	5	2	2	5	3	5	
12	5	3	2	1	2	3	4	3	
13	5	1	1	1	4	1	2	5	
14	5	1	2	1	4	3	3	5	
15	5	5	3	2	1	5	5	2	
16	5	2	1	1	2	3	4	5	
17	1	2	2	3	4	3	3	5	
18	5	3	1	1	1	2	4	4	
19	5	3	3	3	2	2	4	4	
20	5	5	4	3	4	5	5	4	
21	5	3	3	2	4	2	2	4	
22	5	3	2	3	4	3	2	5	
23	5	1	1	3	2	2	2	5	
24	5	3	2	3	3	3	4		
25	5	4	2	2	2	4	4	5	
26	5	3	1	1	4	3	3	5	
27	5	4	2	1	2	3	5	1	
28	5	5	5	4	5	3	4	4	
29	4	3	4	1	3	2	2	4	
30	5	5	1	1	1	1	3	4	
31	5	3	4	2	3	3	3	4	
32	4	4	3	3	3	3	4	4	
33	4	4	1	3	2	3	5	3	
34	5	3	1	3	2	3	3	4	
35	5	2	2	3	4	5	4	3	

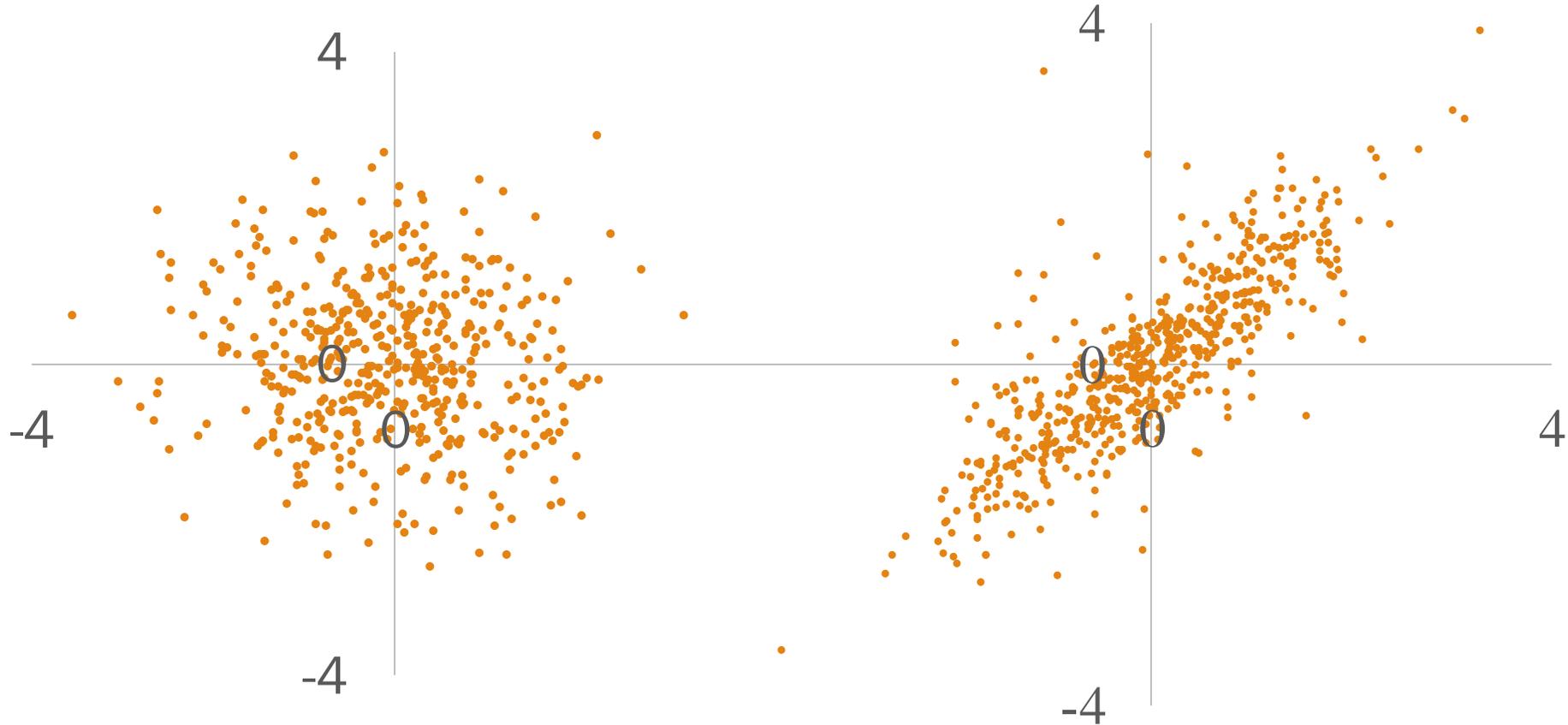
Bat Data

Gene1	Gene2	Gene3	Gene4	Gene5	Trait
TRUE	FALSE	TRUE	TRUE	FALSE	FALSE
FALSE	FALSE	TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	TRUE	FALSE	FALSE	FALSE
TRUE	FALSE	TRUE	TRUE	TRUE	FALSE
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE
TRUE	FALSE	TRUE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	TRUE	FALSE	FALSE
TRUE	TRUE	FALSE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE
TRUE	FALSE	TRUE	TRUE	TRUE	FALSE
FALSE	FALSE	TRUE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE
		...			
TRUE	FALSE	FALSE	TRUE	FALSE	FALSE

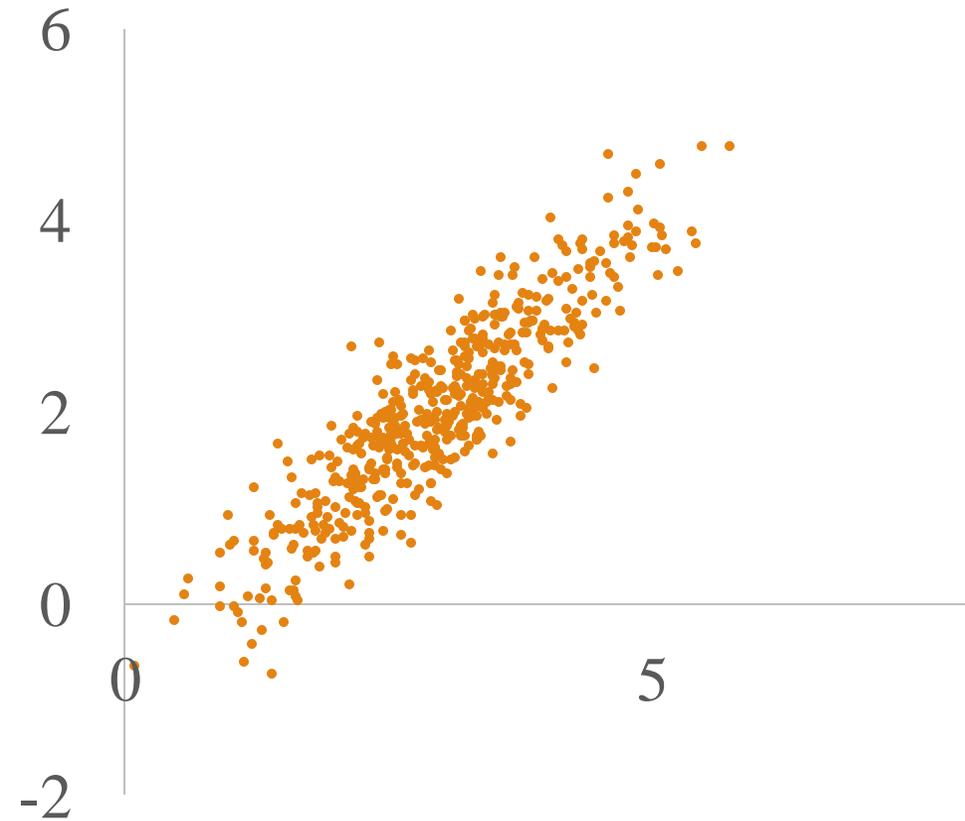
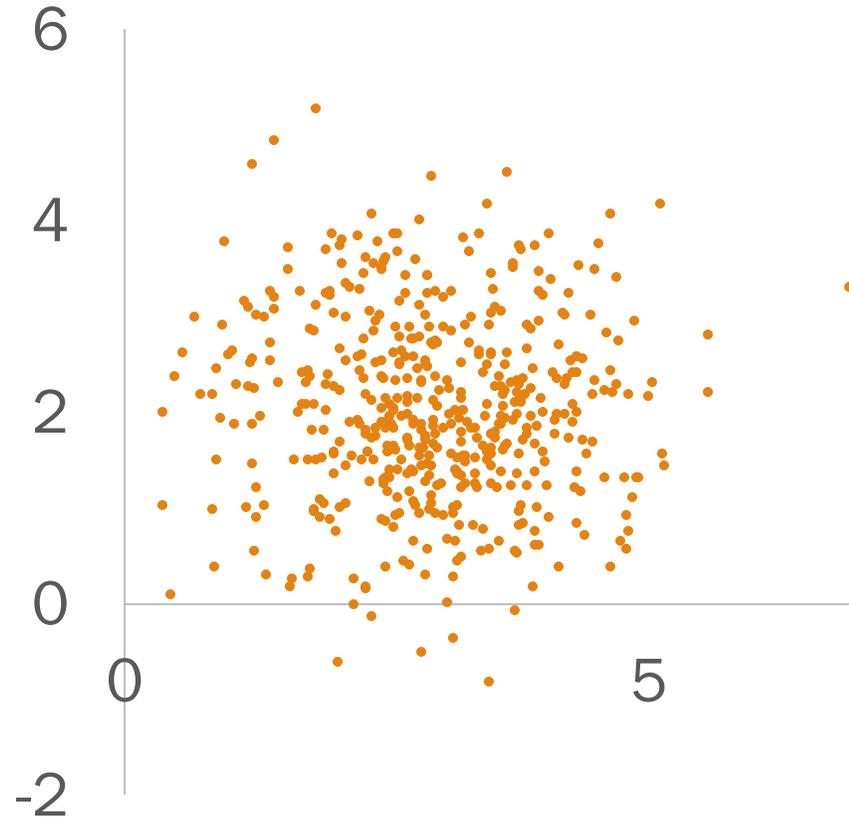
Expression Amount

Gene5	Trait
0.76	0.83
0.94	0.85
0.82	0.03
0.94	0.32
0.50	0.10
0.40	0.53
0.90	0.67
0.29	0.71
0.72	0.25
0.15	0.24
0.79	0.98
0.68	0.77
0.71	0.37
0.36	0.18
0.62	0.08
0.59	0.38
0.82	0.76

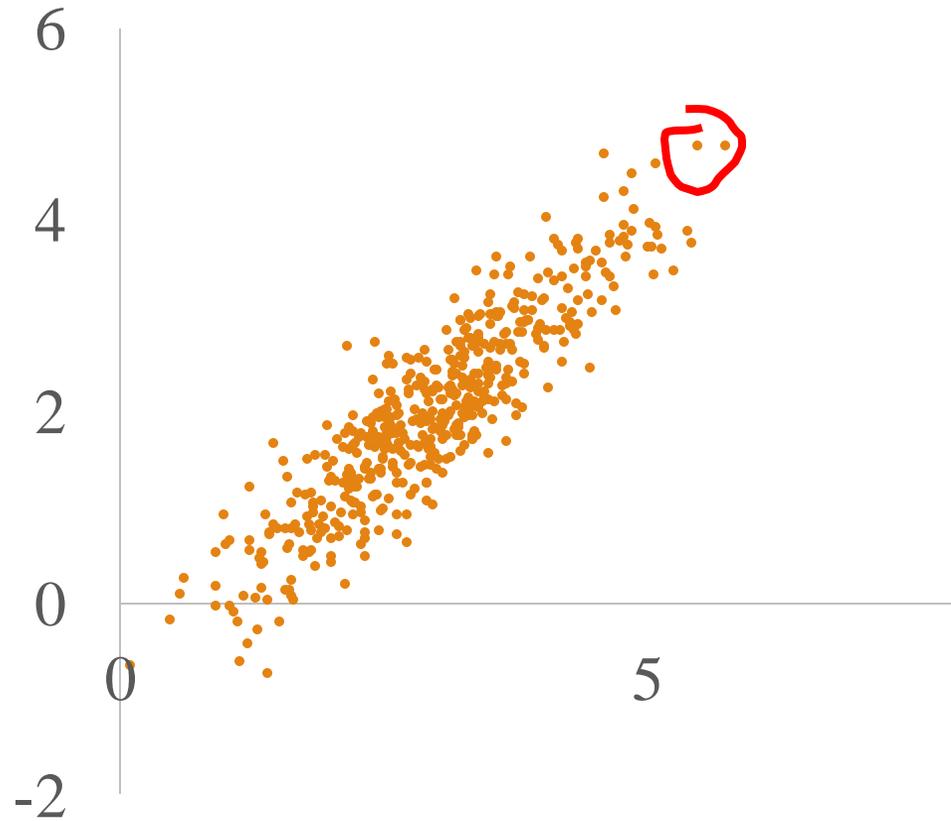
Spot The Difference



Spot The Difference



Vary Together

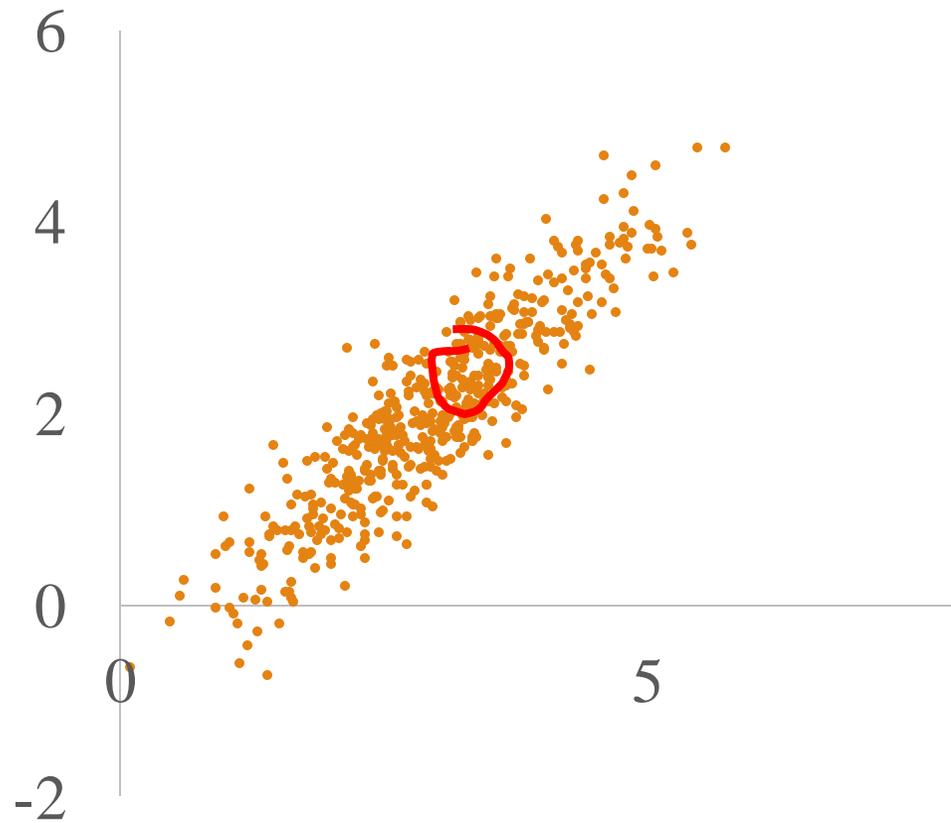


$$x - E[x] = 3$$

$$y - E[y] = 2.6$$

$$(x - E[x])(y - E[y]) = 7.8$$

Vary Together

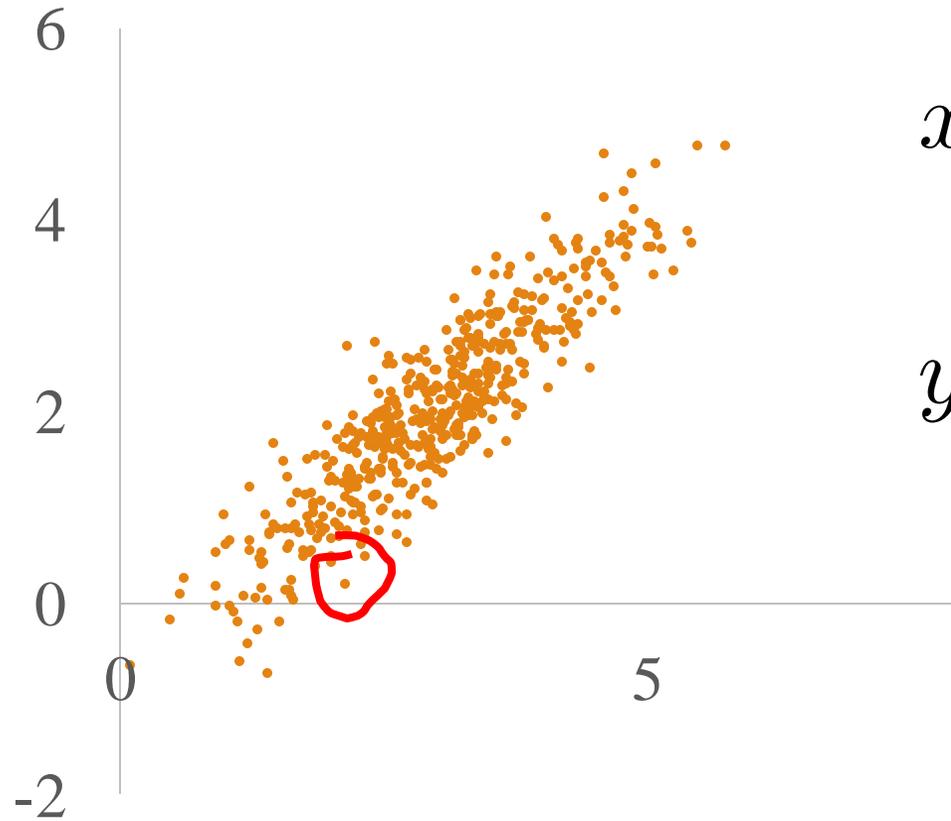


$$x - E[x] \approx 0$$

$$y - E[y] \approx 0$$

$$(x - E[x])(y - E[y]) = 0$$

Vary Together

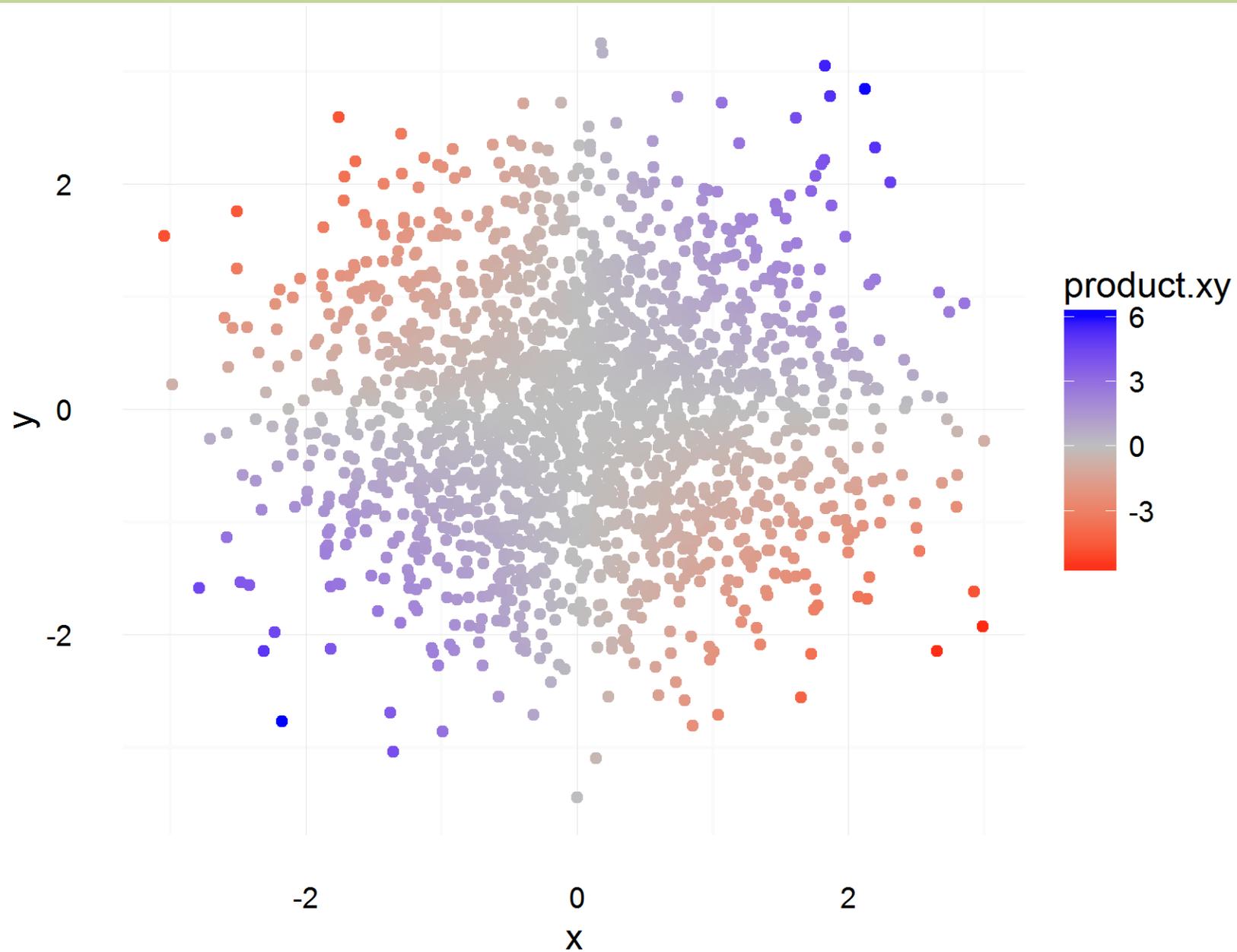


$$x - E[x] = -1.1$$

$$y - E[y] = -2.8$$

$$(x - E[x])(y - E[y]) \approx 3.1$$

Understanding Covariance



The Dance of the Covariance

Say X and Y are arbitrary random variables

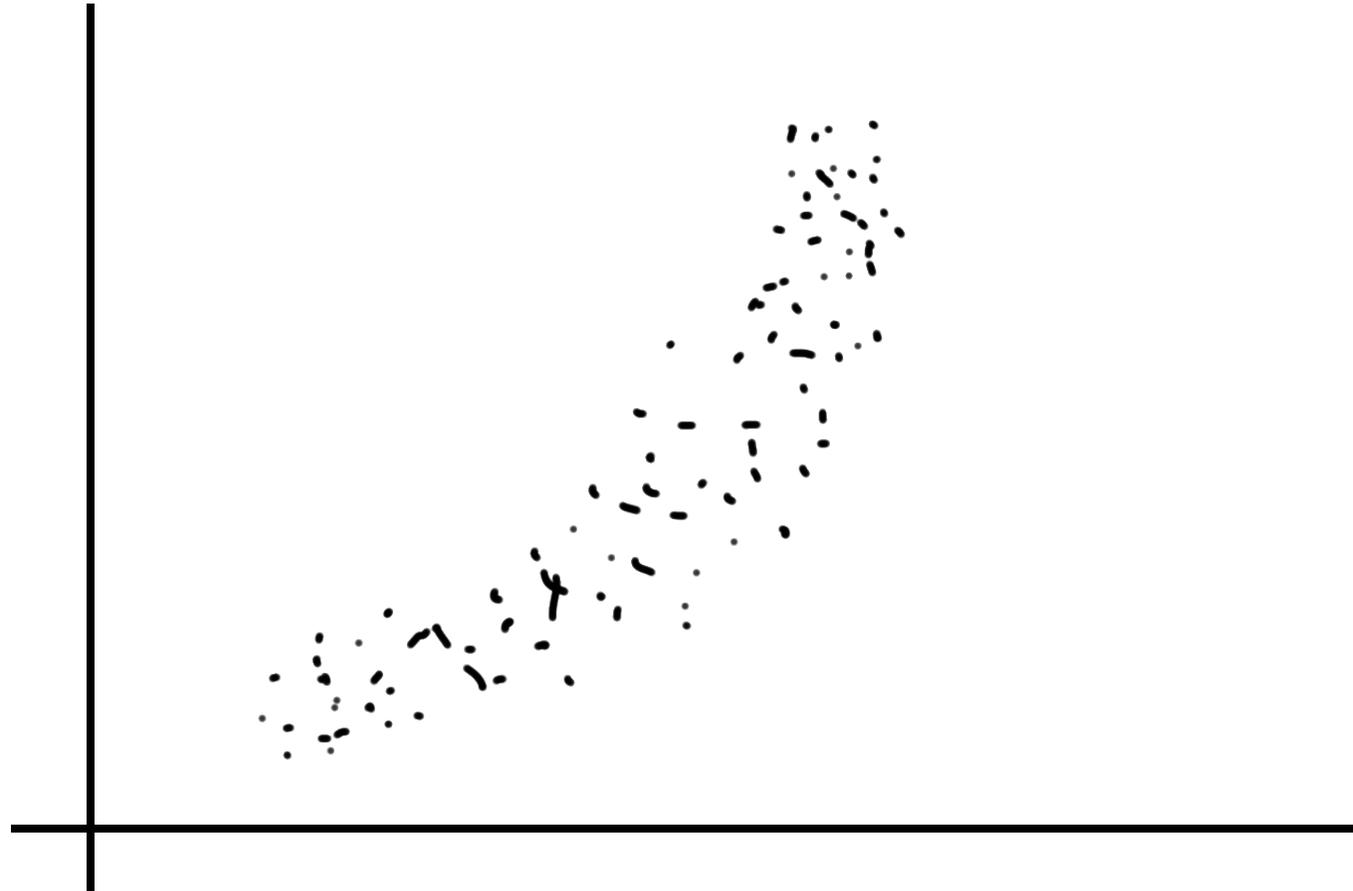
Covariance of X and Y :

$$\text{Cov}(X, Y) = E[(X - E[X])(Y - E[Y])]$$

x	y	$(x - E[X])(y - E[Y])p(x,y)$
Above mean	Above mean	Positive
Bellow mean	Bellow mean	Positive
Bellow mean	Above mean	Negative
Above mean	Bellow mean	Negative

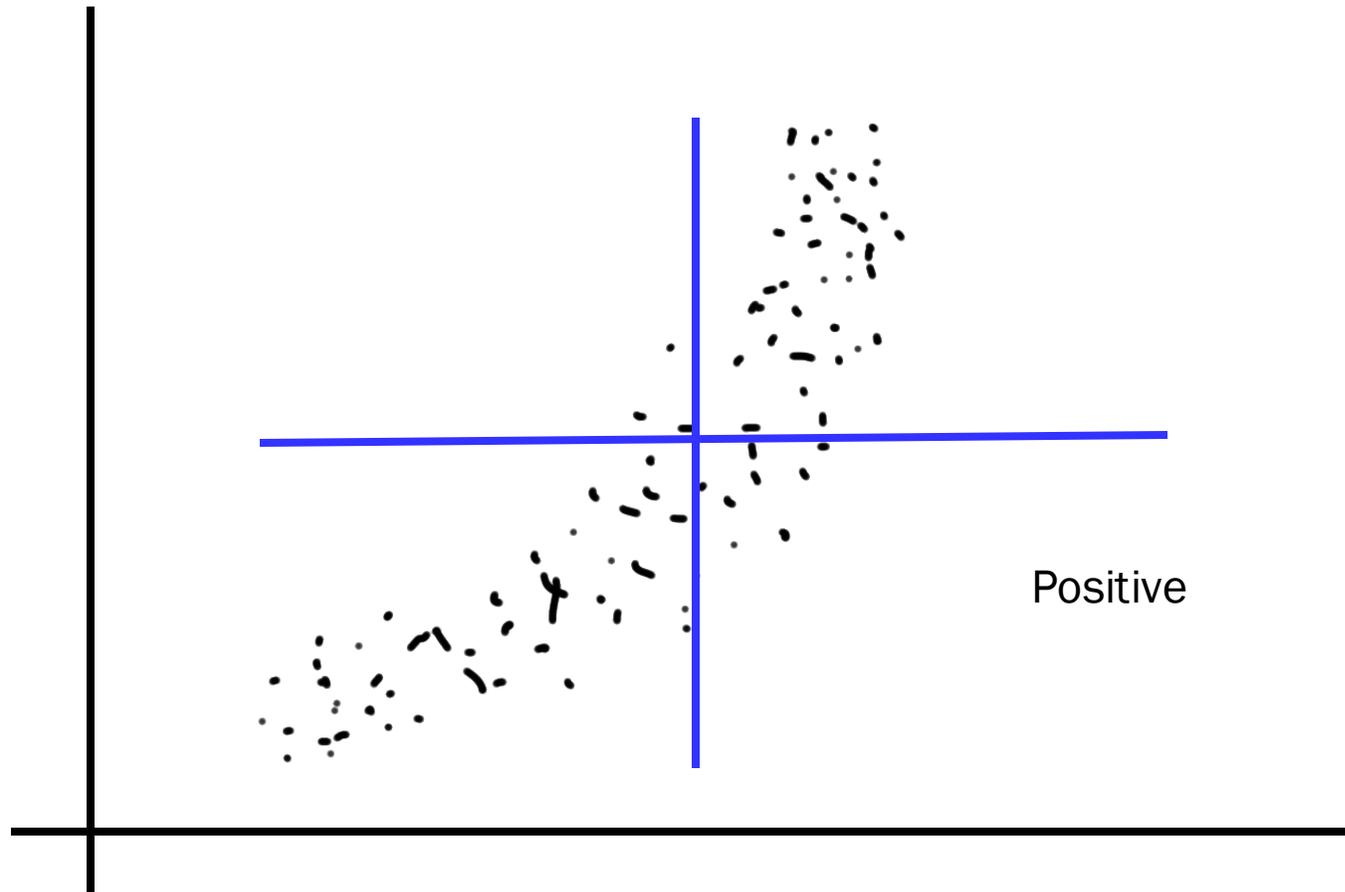
Covariance

Poll: (a) positive, (b) negative, (c) zero



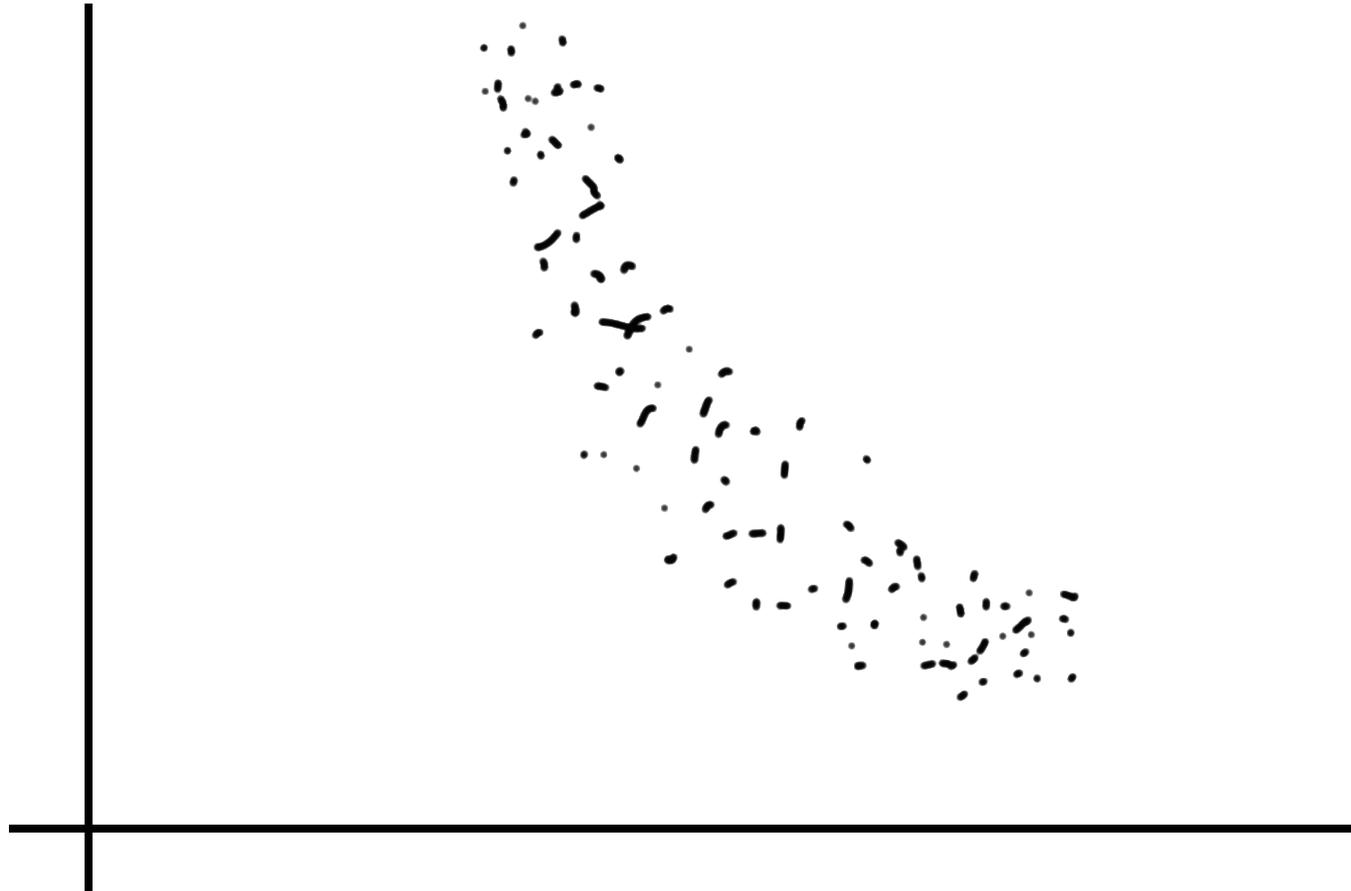
Covariance

Is the Covariance: (a) positive, (b) negative, (c) zero



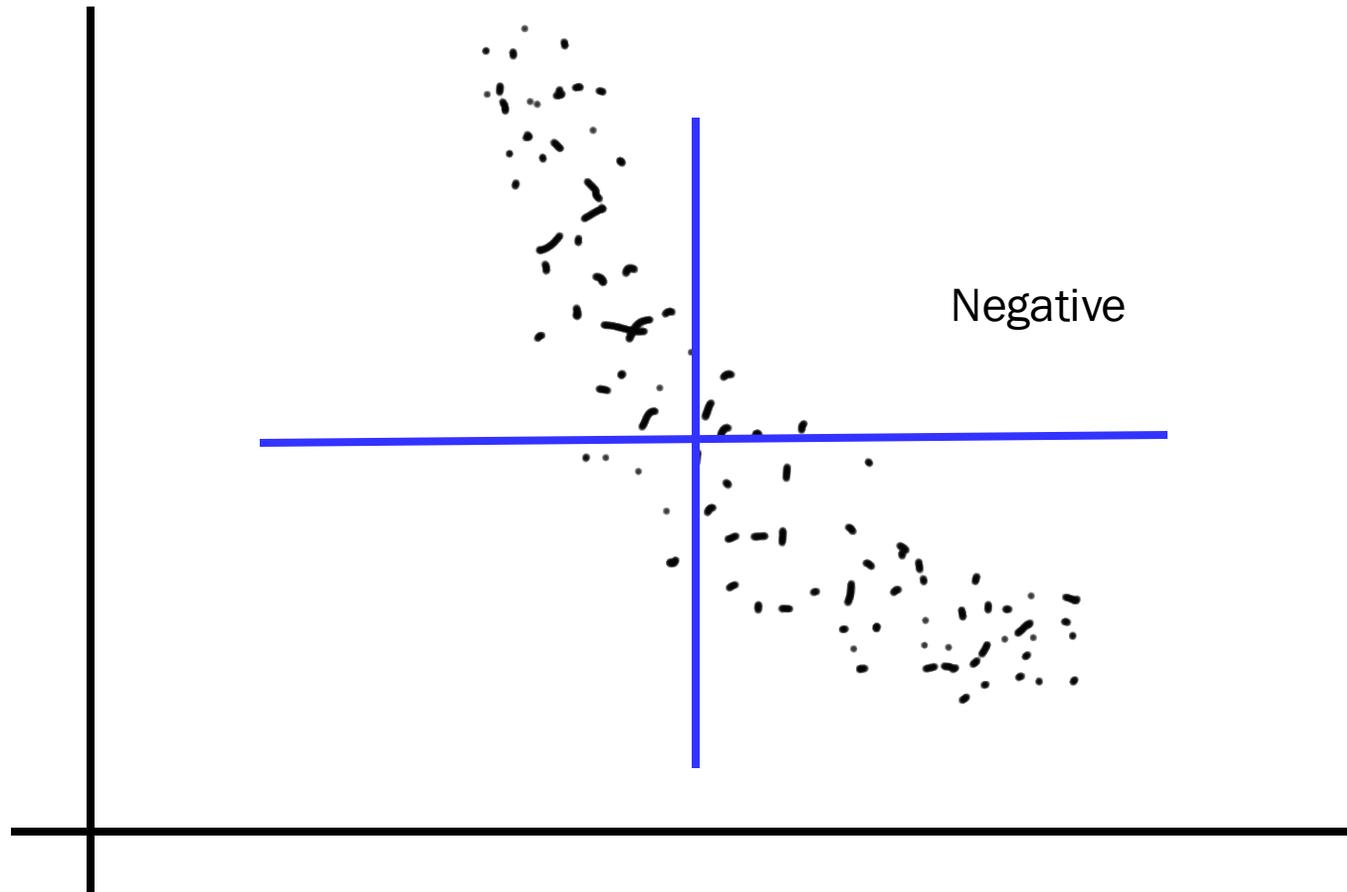
Covariance

Is the Covariance: (a) positive, (b) negative, (c) zero



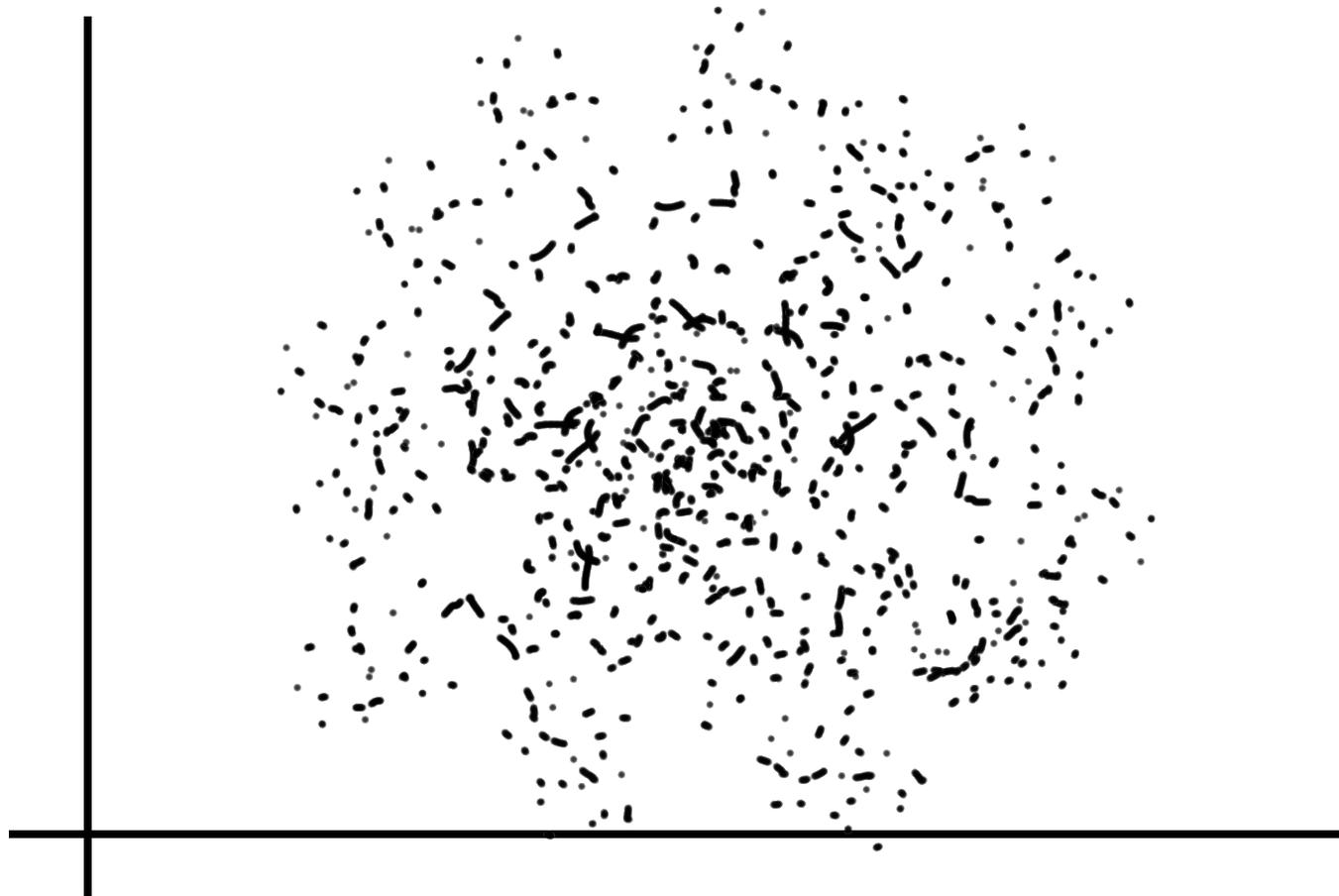
Covariance

Is the Covariance: (a) positive, (b) negative, (c) zero



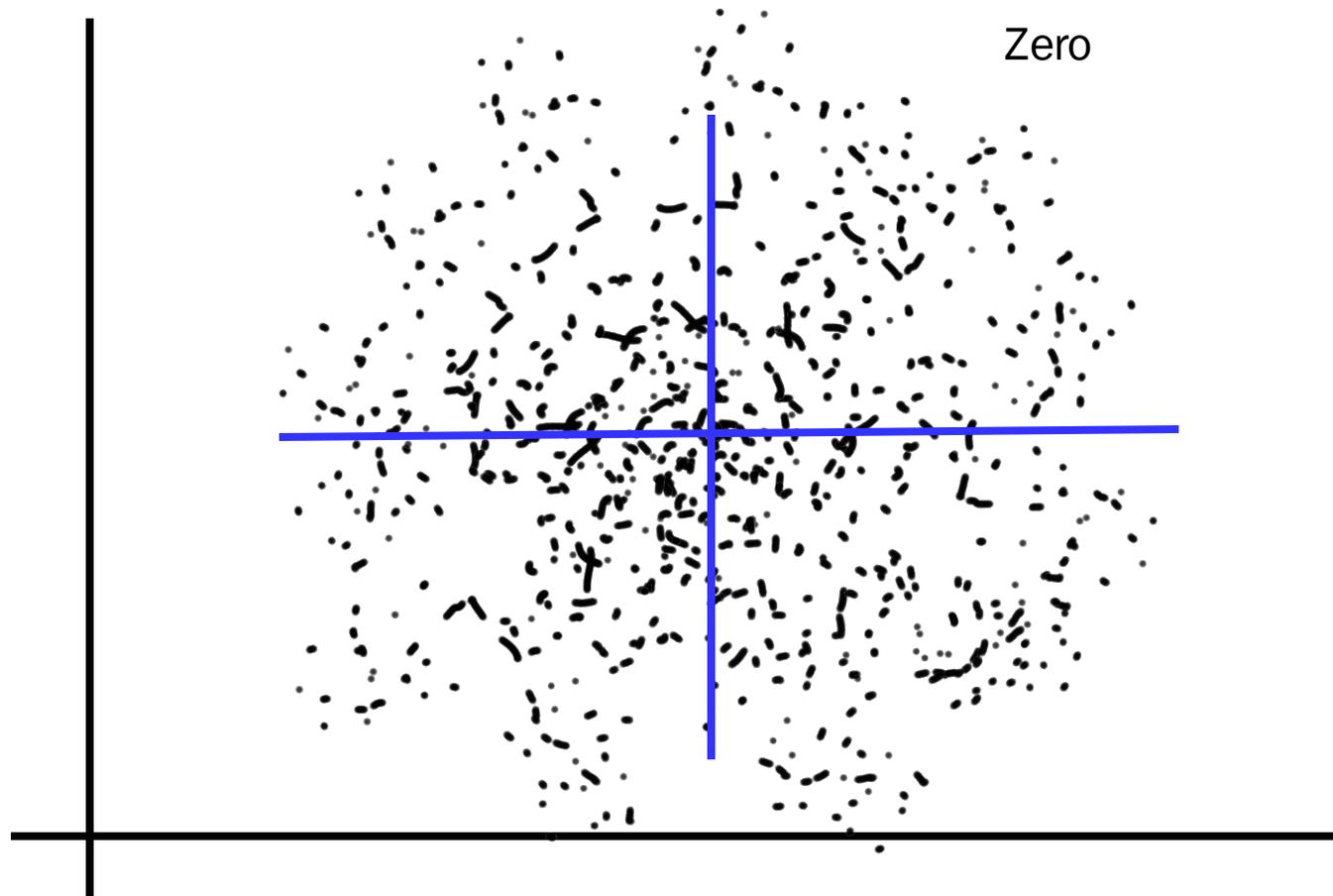
Covariance

Is the Covariance: (a) positive, (b) negative, (c) zero



Covariance

Is the Covariance: (a) positive, (b) negative, (c) zero



The Dance of the Covariance

Say X and Y are arbitrary random variables

Covariance of X and Y :

$$\text{Cov}(X, Y) = E[(X - E[X])(Y - E[Y])]$$

Equivalently:

$$\begin{aligned}\text{Cov}(X, Y) &= E[XY - E[X]Y - XE[Y] + E[Y]E[X]] \\ &= E[XY] - E[X]E[Y] - E[X]E[Y] + E[X]E[Y] \\ &= E[XY] - E[X]E[Y]\end{aligned}$$

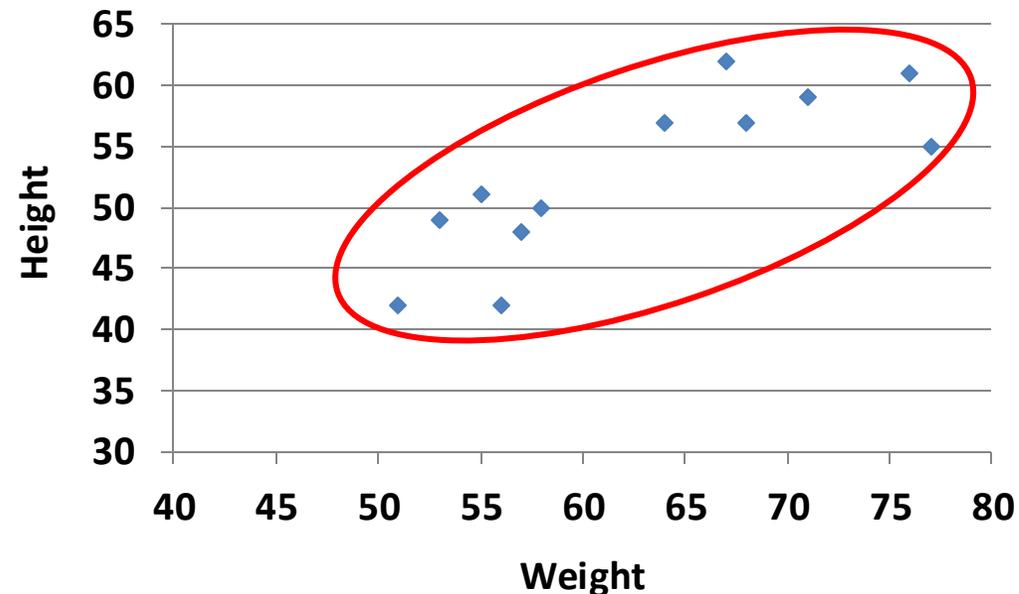
- X and Y independent $\rightarrow \text{Cov}(X, Y) = 0$
- But $\text{Cov}(X, Y) = 0$ does **not** imply X and Y independent!

Covariance and Data

Consider the following data:

Weight	Height	Weight * Height
64	57	3648
71	59	4189
53	49	2597
67	62	4154
55	51	2805
58	50	2900
77	55	4235
57	48	2736
56	42	2352
51	42	2142
76	61	4636
68	57	3876

$$\begin{array}{lll} E[W] & E[H] & E[W*H] \\ = 62.75 & = 52.75 & = 3355.83 \end{array}$$



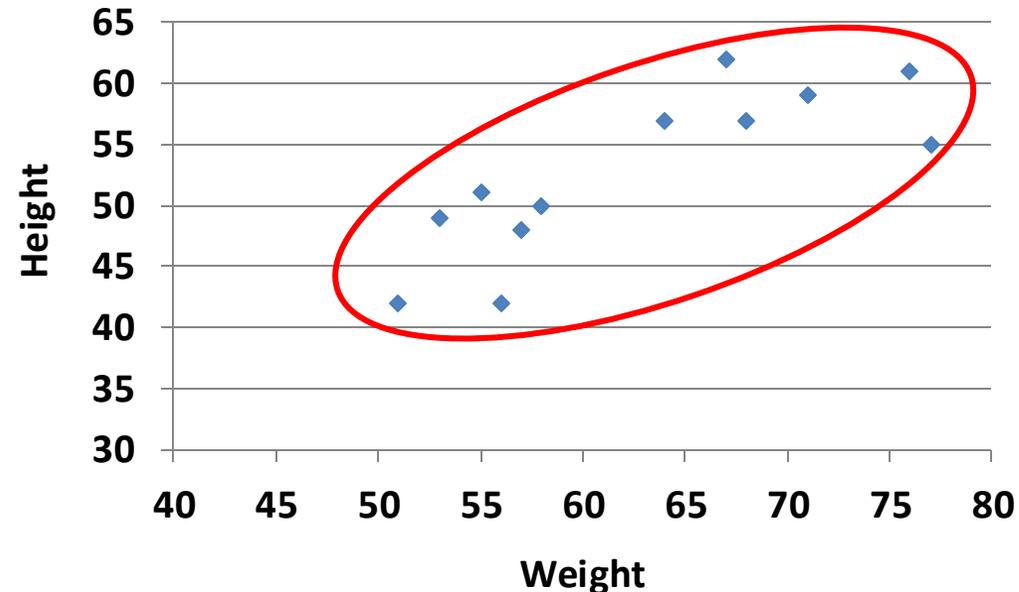
$$\begin{aligned} \text{Cov}(W, H) &= E[W*H] - E[W]E[H] \\ &= 3355.83 - (62.75)(52.75) \\ &= 45.77 \end{aligned}$$

Correlation

What is Wrong With This?

Consider the following data:

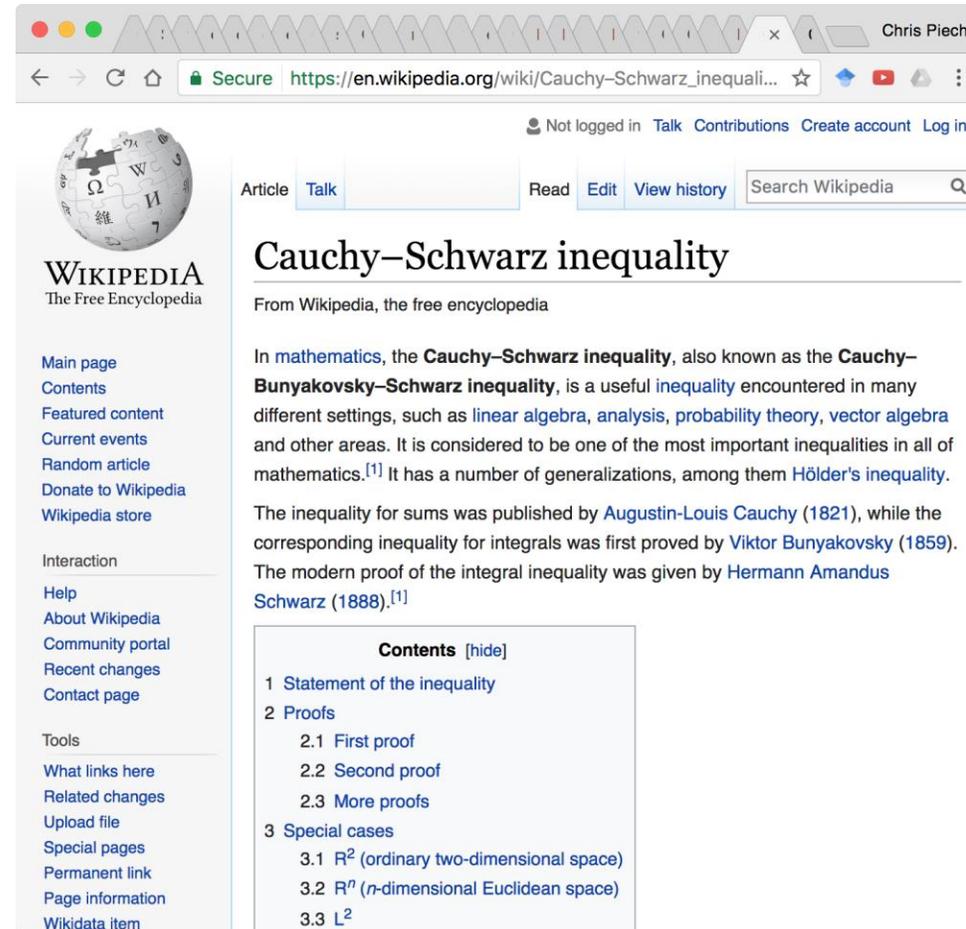
Weight	Height	Weight * Height
64	57	3648
71	59	4189
53	49	2597
67	62	4154
55	51	2805
58	50	2900
77	55	4235
57	48	2736
56	42	2352
51	42	2142
76	61	4636
68	57	3876



$$\begin{aligned} E[W] &= 62.75 \\ E[H] &= 52.75 \\ E[W*H] &= 3355.83 \end{aligned}$$

$$\begin{aligned} \text{Cov}(W, H) &= E[W*H] - E[W]E[H] \\ &= 3355.83 - (62.75)(52.75) \\ &= 45.77 \end{aligned}$$

Cauchy Schwarz, a great way to normalize!



The image shows a screenshot of a web browser displaying the Wikipedia article for the Cauchy–Schwarz inequality. The browser's address bar shows the URL https://en.wikipedia.org/wiki/Cauchy–Schwarz_inequality. The page title is "Cauchy–Schwarz inequality". The article text states: "In **mathematics**, the **Cauchy–Schwarz inequality**, also known as the **Cauchy–Bunyakovsky–Schwarz inequality**, is a useful **inequality** encountered in many different settings, such as **linear algebra**, **analysis**, **probability theory**, **vector algebra** and other areas. It is considered to be one of the most important inequalities in all of mathematics.^[1] It has a number of generalizations, among them **Hölder's inequality**. The inequality for sums was published by **Augustin-Louis Cauchy** (1821), while the corresponding inequality for integrals was first proved by **Viktor Bunyakovsky** (1859). The modern proof of the integral inequality was given by **Hermann Amandus Schwarz** (1888).^[1]" Below the text is a "Contents" section with the following items: 1 Statement of the inequality, 2 Proofs (2.1 First proof, 2.2 Second proof, 2.3 More proofs), and 3 Special cases (3.1 \mathbb{R}^2 (ordinary two-dimensional space), 3.2 \mathbb{R}^n (n -dimensional Euclidean space), 3.3 L^2).

$$-\text{Std}(X)\text{Std}(Y) \leq \text{Cov}(X, Y) \leq \text{Std}(X)\text{Std}(Y)$$

Viva La Correlación

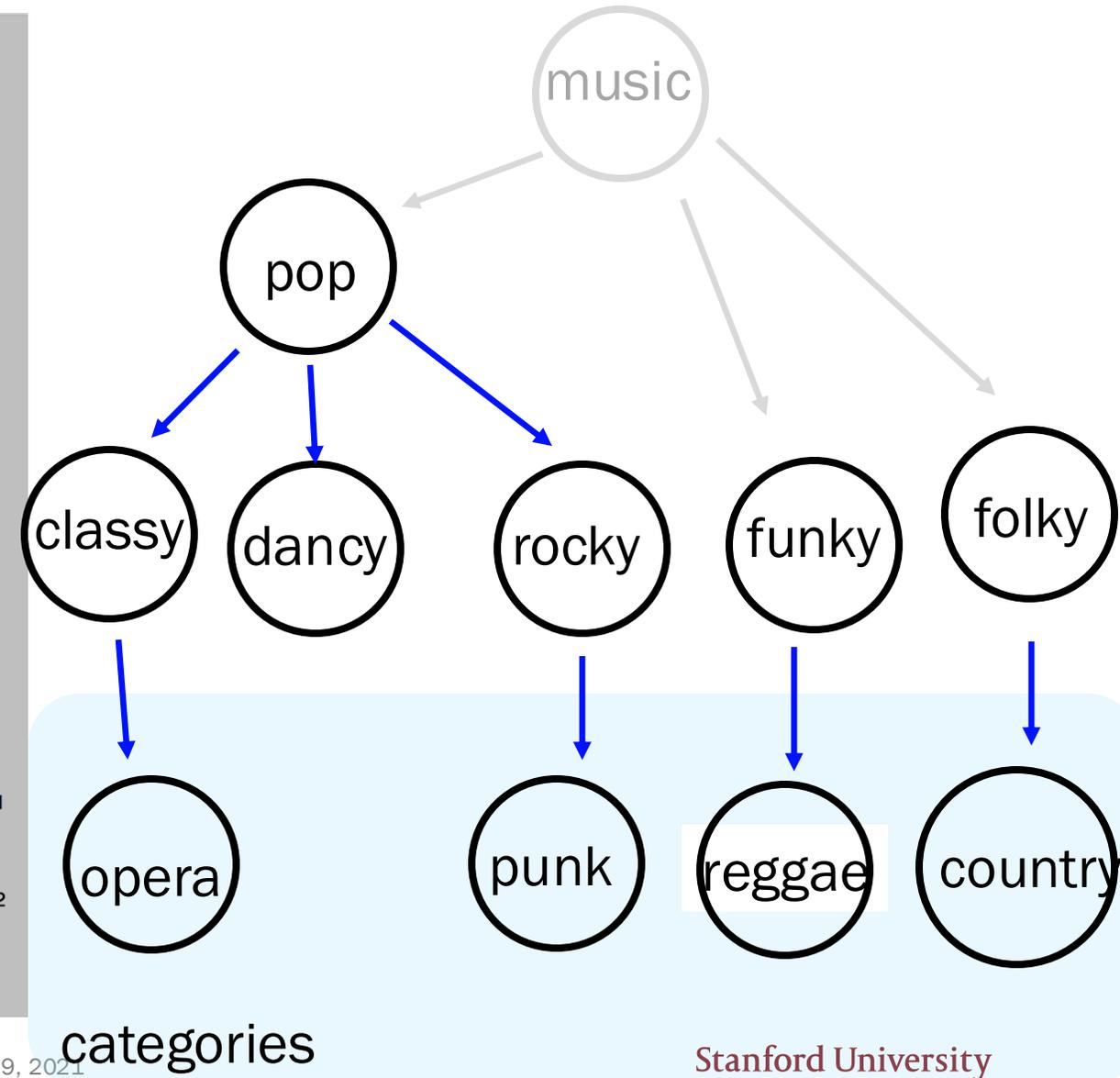
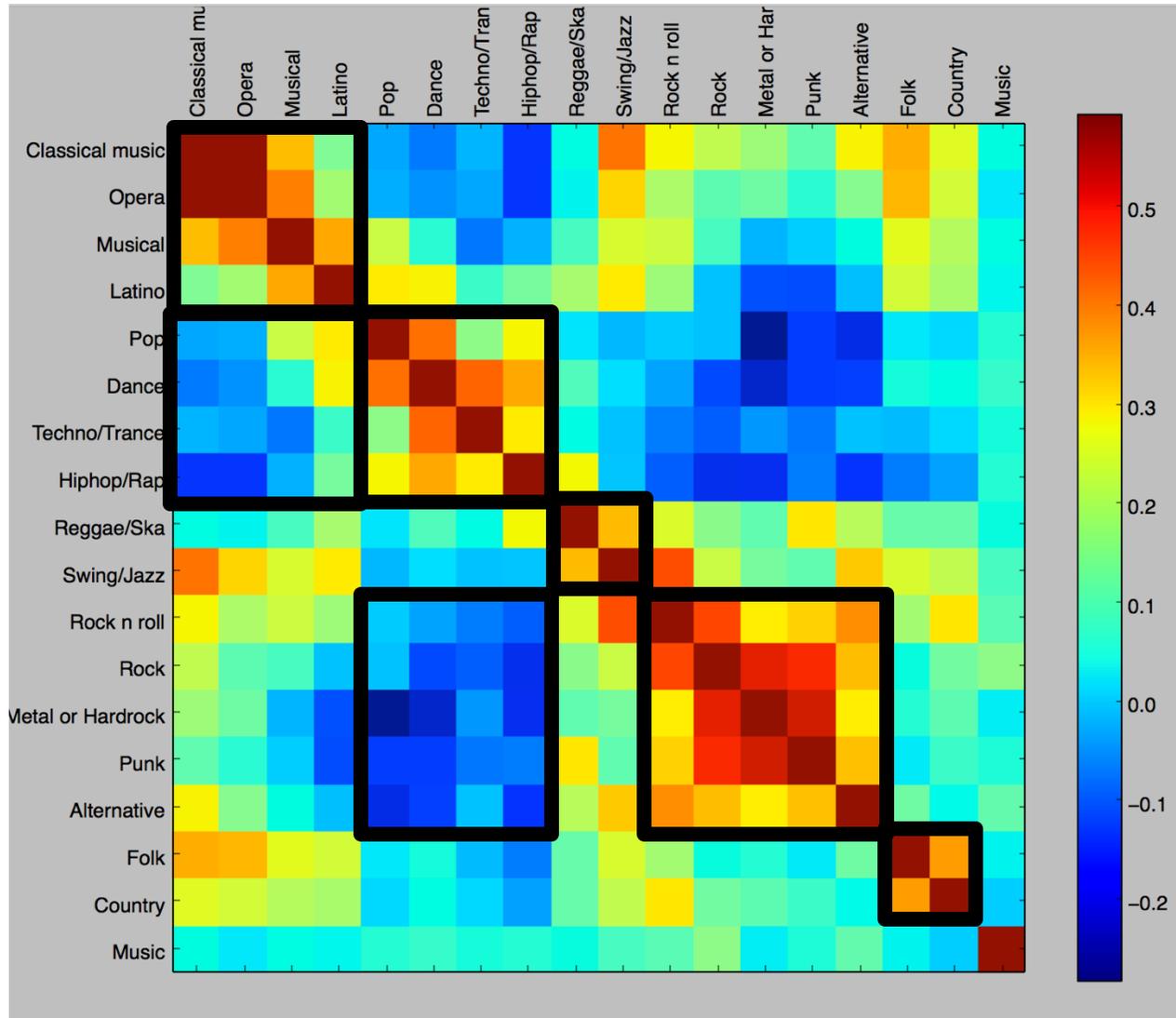
Say X and Y are arbitrary random variables

- Correlation of X and Y , denoted $\rho(X, Y)$:

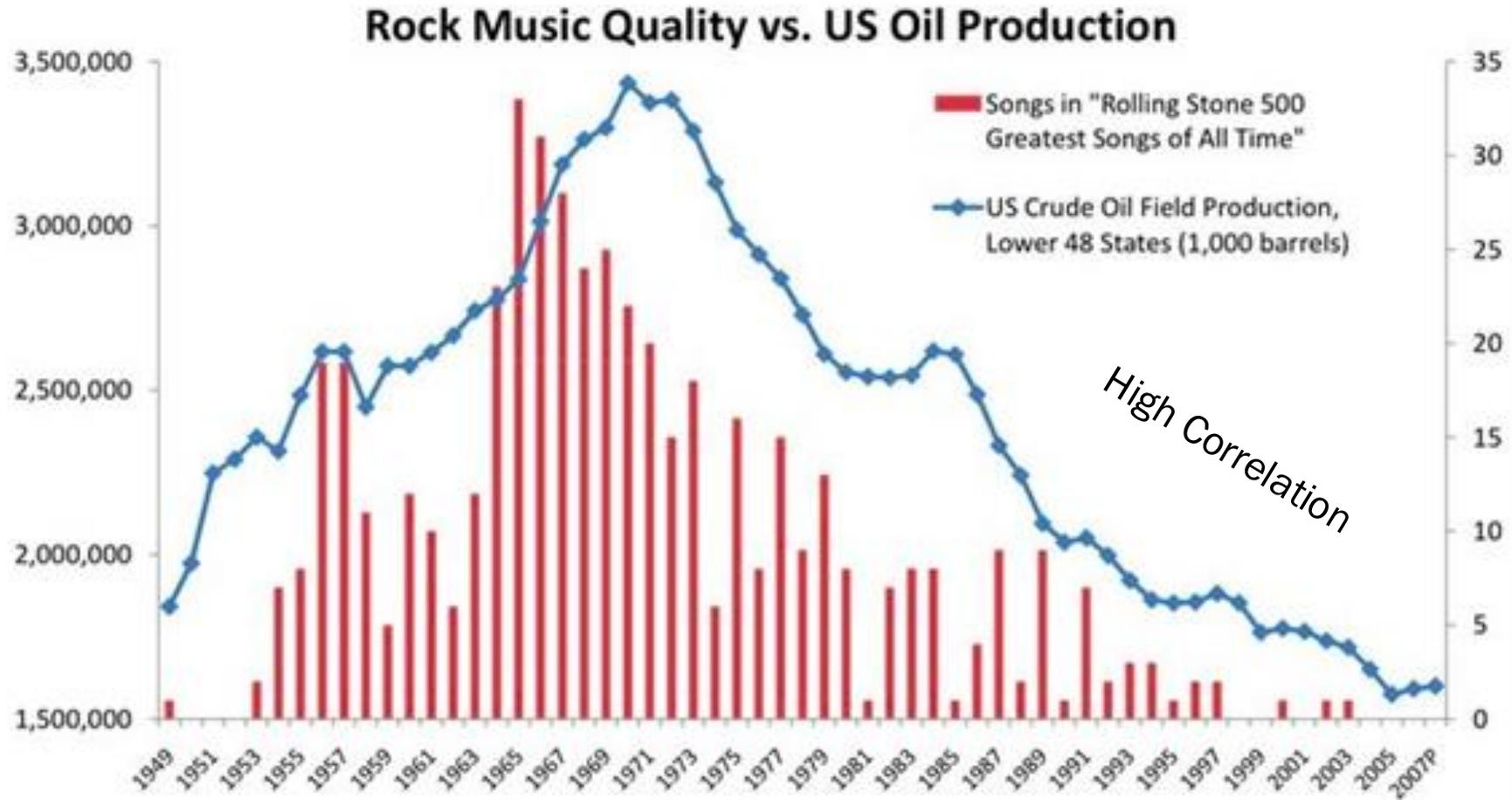
$$\rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}}$$

- Note: $-1 \leq \rho(X, Y) \leq 1$
- $\rho(X, Y) = 1 \quad \Rightarrow$ perfectly correlated
- $\rho(X, Y) = -1 \quad \Rightarrow$ perfectly negatively correlated
- $\rho(X, Y) = 0 \quad \Rightarrow$ absence of linear relationship
 - But, X and Y can still be related in some other way!
- If $\rho(X, Y) = 0$, we say X and Y are “uncorrelated”

Recall: It is a useful starting point



Rock Music Vs Oil?

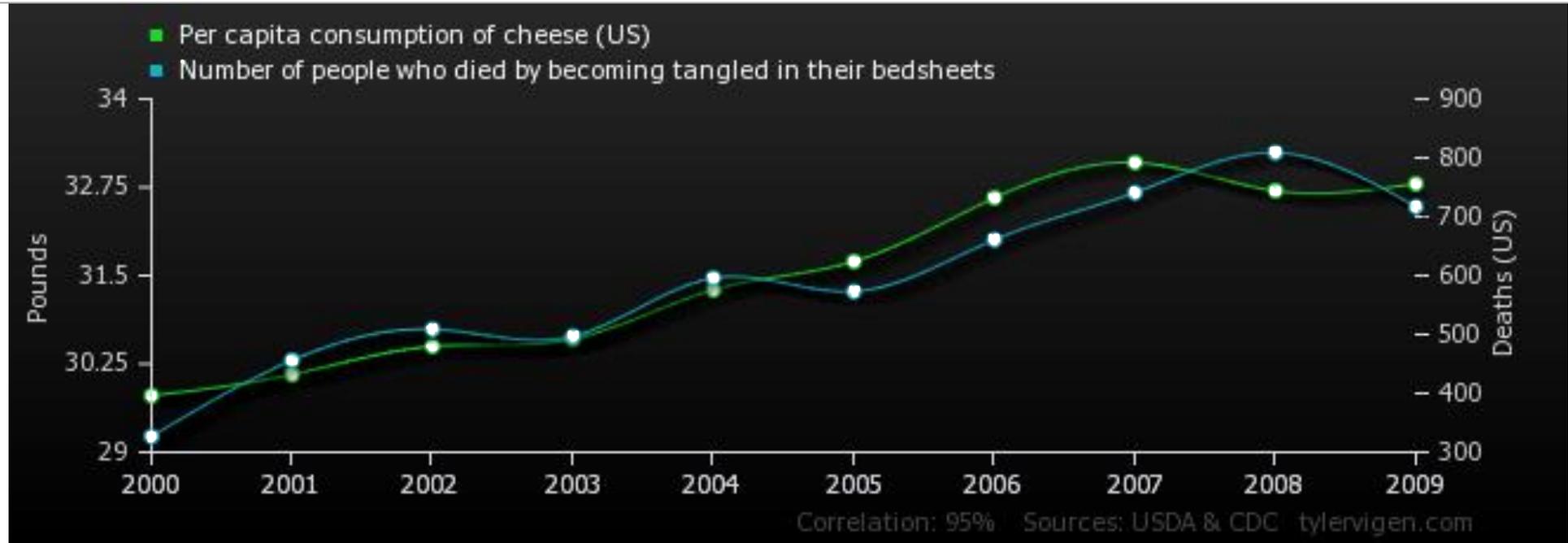


Hubbert Peak Theory

<http://www.aei.org/publication/blog/>

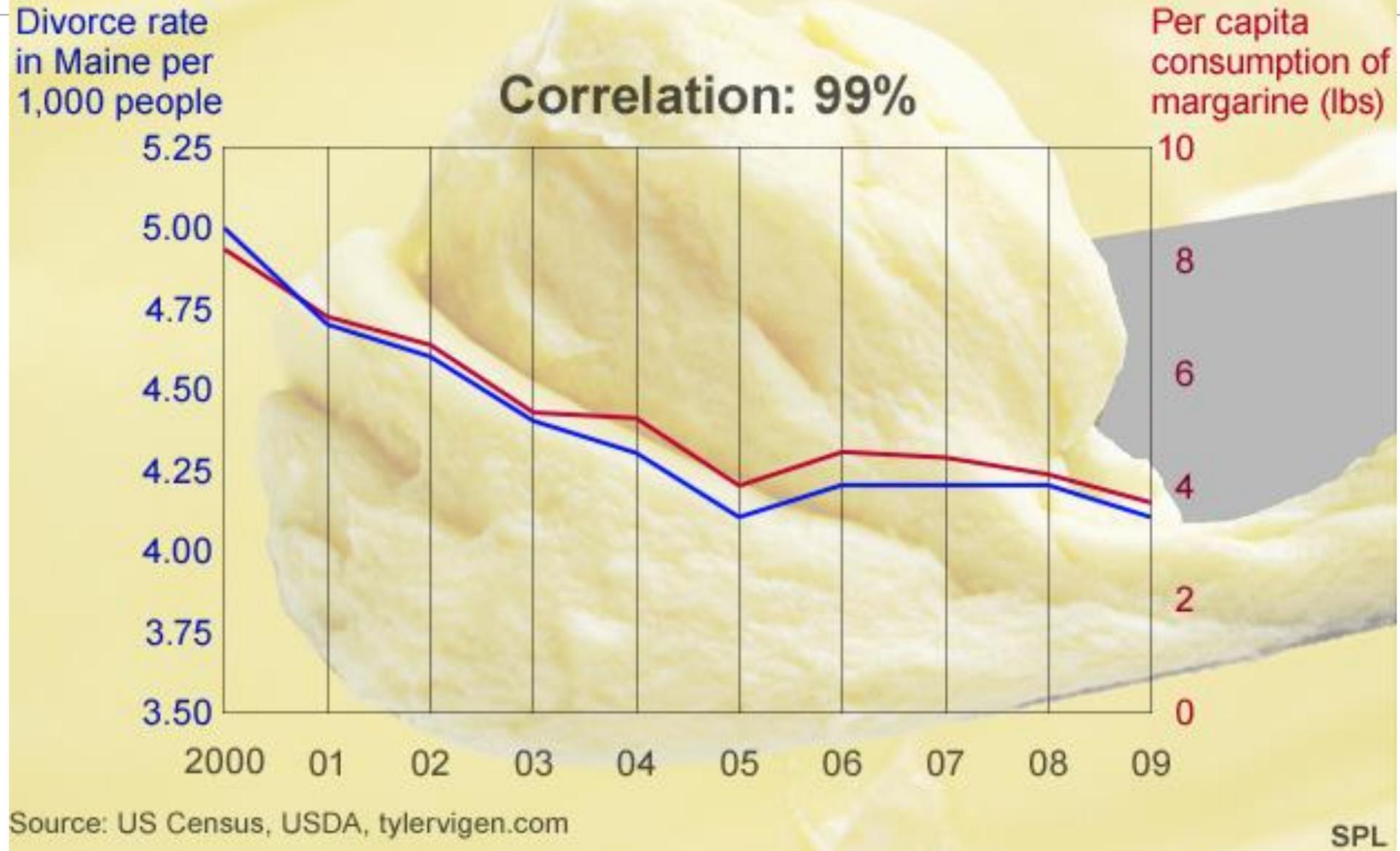
Stanford University

Tell your friends!



	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
<i>Per capita consumption of cheese (US) Pounds (USDA)</i>	29.8	30.1	30.5	30.6	31.3	31.7	32.6	33.1	32.7	32.8
<i>Number of people who died by becoming tangled in their bedsheets Deaths (US) (CDC)</i>	327	456	509	497	596	573	661	741	809	717
Correlation: 0.947091										

Divorce Vs Butter?



Three Guiding Questions

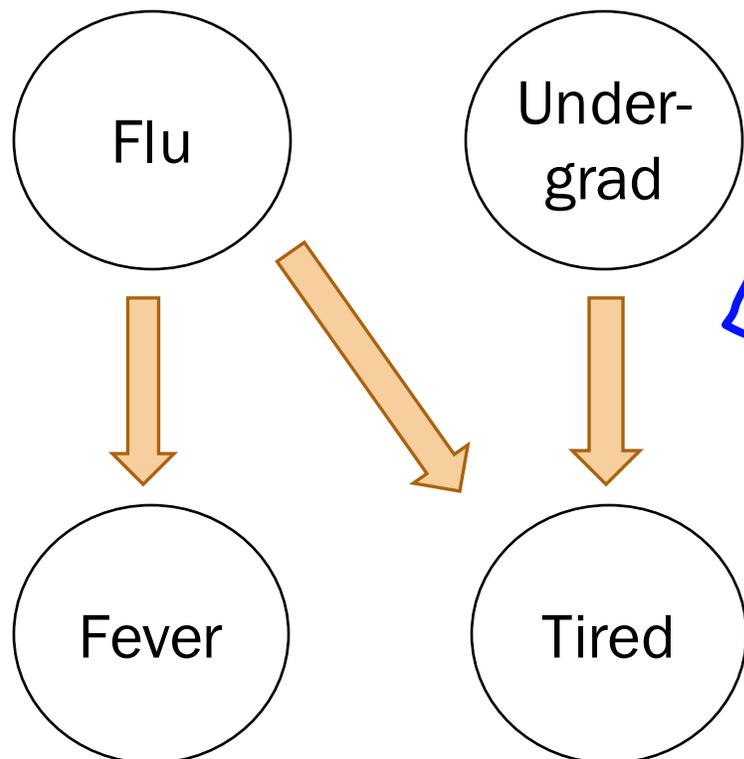
1. How do people actually define large models?
2. How can we do inference in large models?
3. What data can inform the design process?

What haven't we talked about?

Machine Learning (last section of CS109)

$$P(F_{lu} = 1) = 0.1$$

$$P(U = 1) = 0.8$$



1. Learn this from data

2. Learn this from data

$$P(F_{ev} = 1 | F_{lu} = 1) = 0.9$$
$$P(F_{ev} = 1 | F_{lu} = 0) = 0.05$$

$$P(T = 1 | F_{lu} = 0, U = 0) = 0.1$$
$$P(T = 1 | F_{lu} = 0, U = 1) = 0.8$$
$$P(T = 1 | F_{lu} = 1, U = 0) = 0.9$$
$$P(T = 1 | F_{lu} = 1, U = 1) = 1.0$$