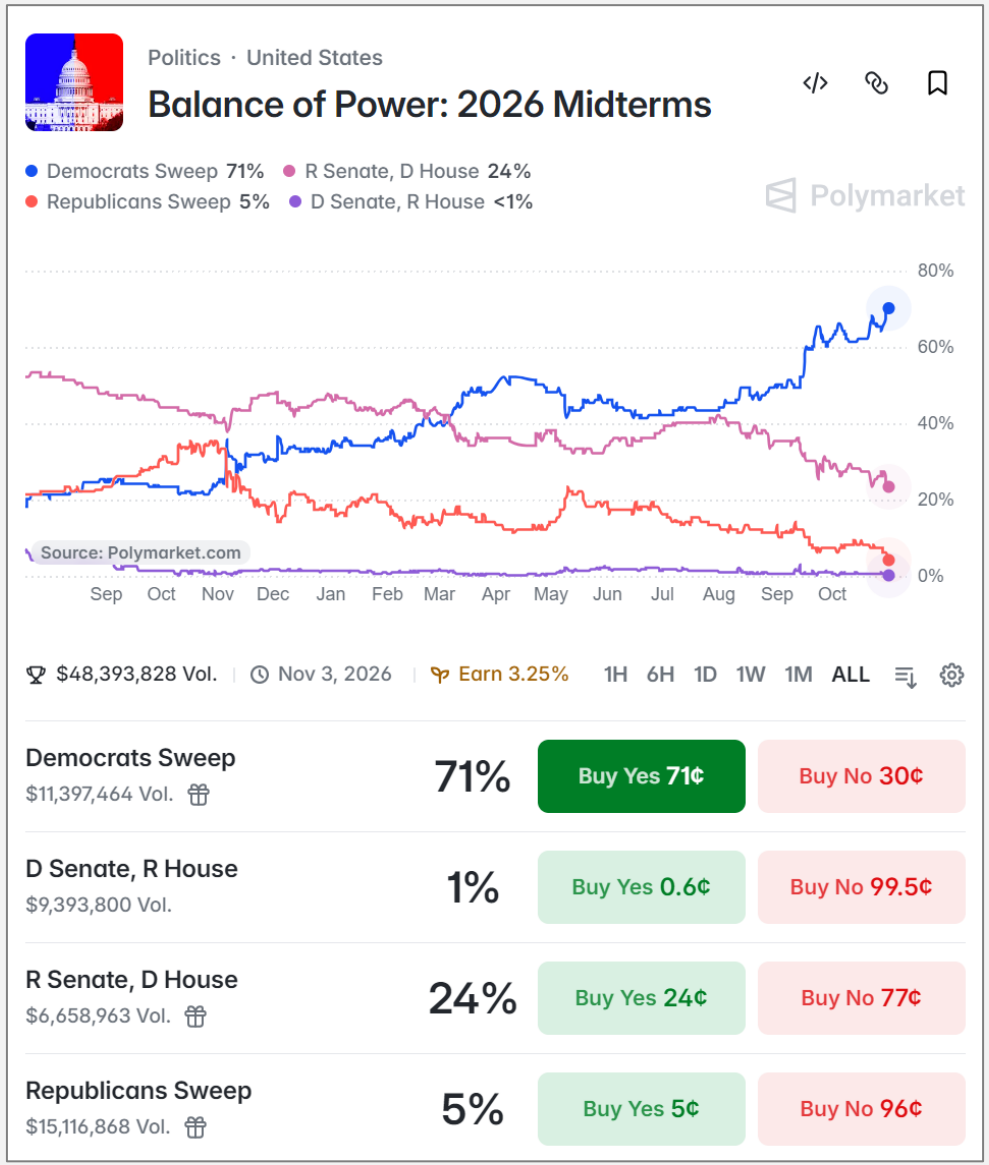


My model:

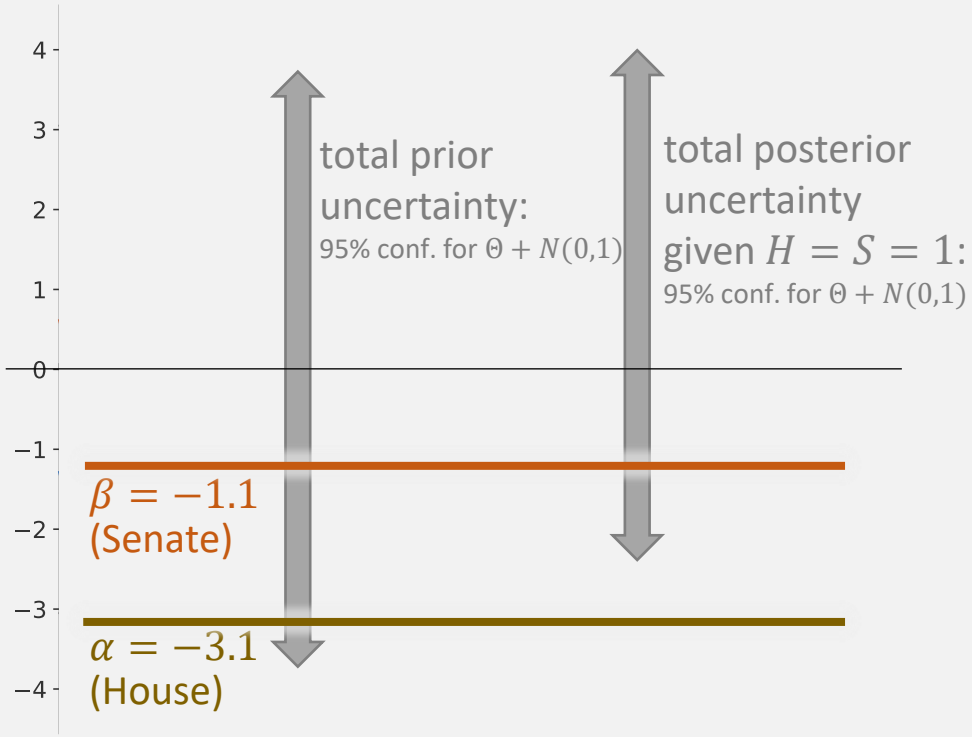
I believe there is a "national sentiment" Θ that describes swing voters, voter turnout, etc.

$$\alpha = -3.1$$
$$\beta = -1.1$$
$$\tau = 1.61$$

$$\Theta \sim N(0, \tau^2)$$

$$H = 1_{\Theta + N(0,1) > \alpha}$$

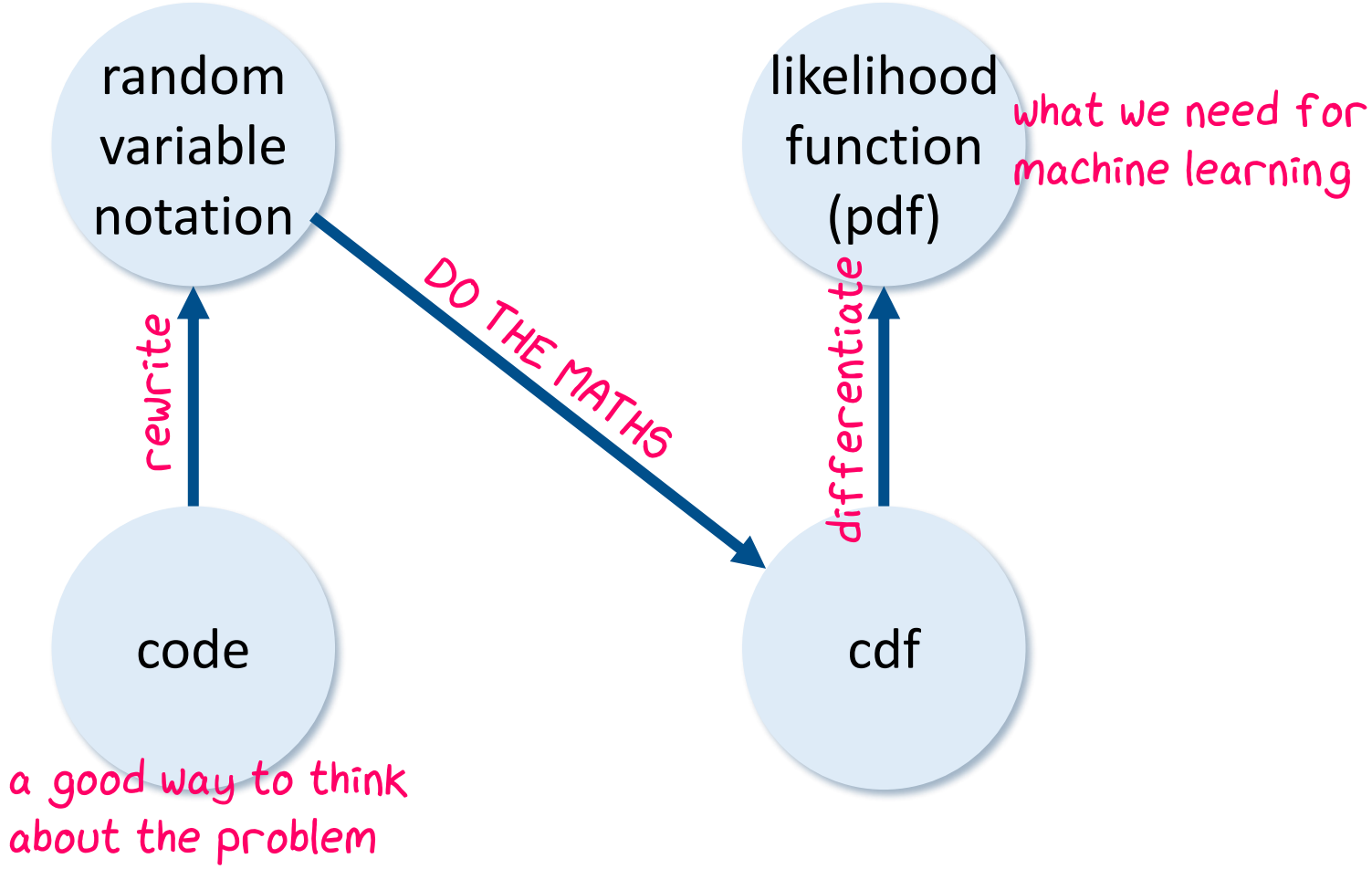
$$S = 1_{\Theta + N(0,1) > \beta}$$



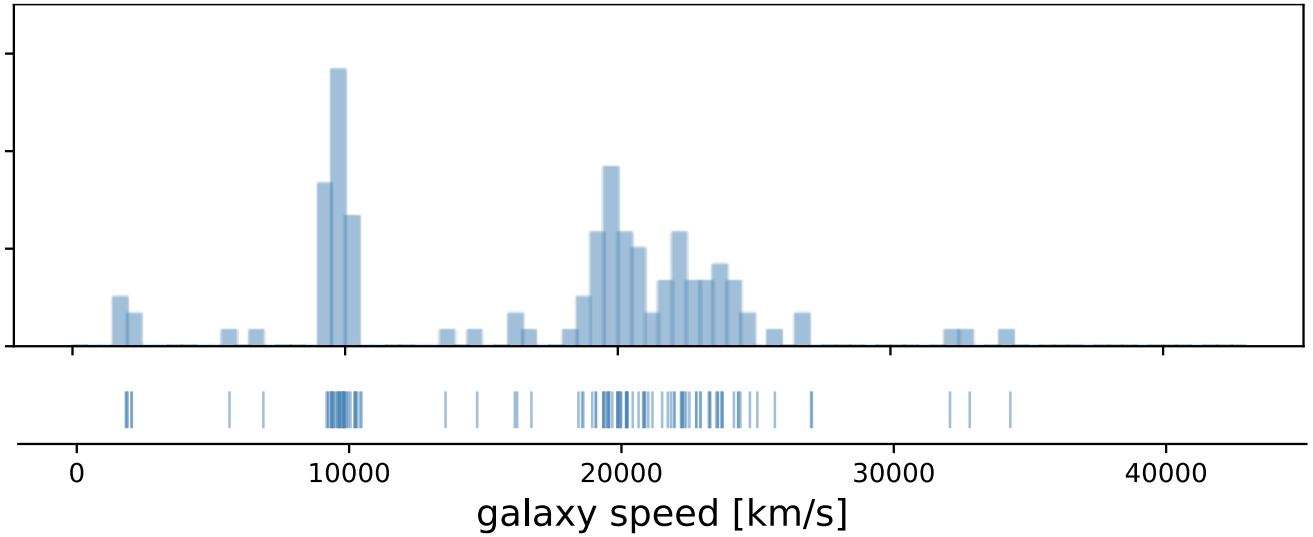
LAST LECTURE

Bespoke probability distributions: from code to likelihood

There are four ways to specify a distribution:



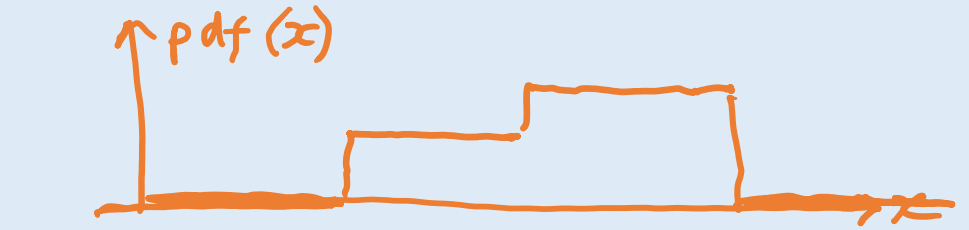
Our goal:
to find the best distribution we can to fit this dataset.

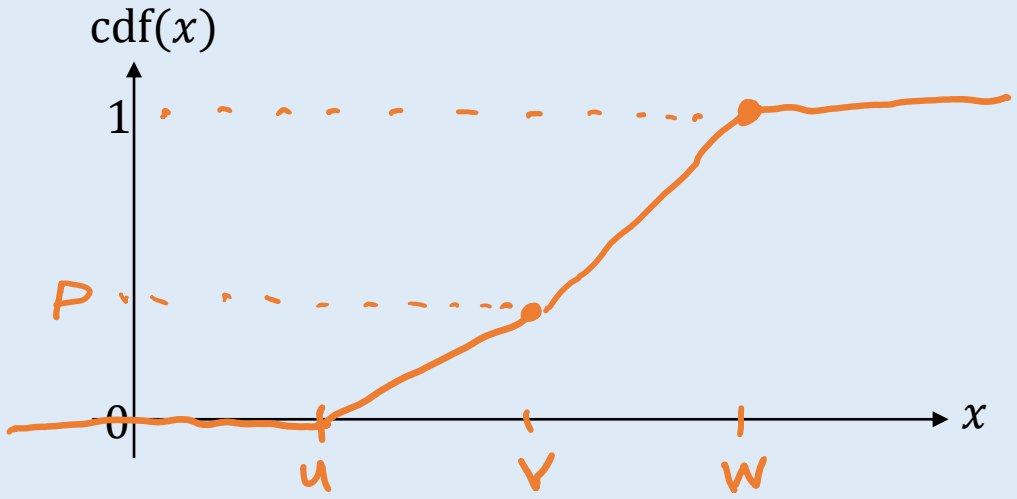


EXERCISE

What's the pdf for this random variable?

```
def rx(u,v,w,p):
    # preconditions: u < v < w, and 0 < p < 1
    k = np.random.choice(["left","right"], [p,1-p])
    if k == "left":
        return np.random.uniform(u,v)
    else:
        return np.random.uniform(v,w)
```

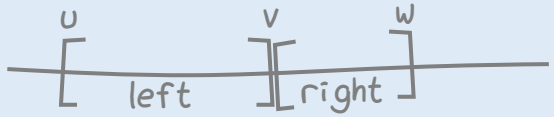




① Rewrite it in random variable notation

$$\text{Let } K = \begin{cases} \text{left} & \text{with prob. } p \\ \text{right} & \text{with prob. } 1-p \end{cases}$$

$$\text{Let } X \sim \begin{cases} U[u,v] & \text{if } K = \text{left} \\ U[v,w] & \text{if } K = \text{right} \end{cases}$$



② Do the maths: simplify the cdf into elementary building blocks

$$\mathbb{P}(X \le x) = \mathbb{P}(X \le x | K = \text{left}) \times \mathbb{P}(K = \text{left}) + \mathbb{P}(X \le x | K = \text{right}) \times \mathbb{P}(K = \text{right})$$

$$= p\,\mathbb{P}(U[u,v] \le x) + (1-p)\,\mathbb{P}(U[v,w] \le x)$$

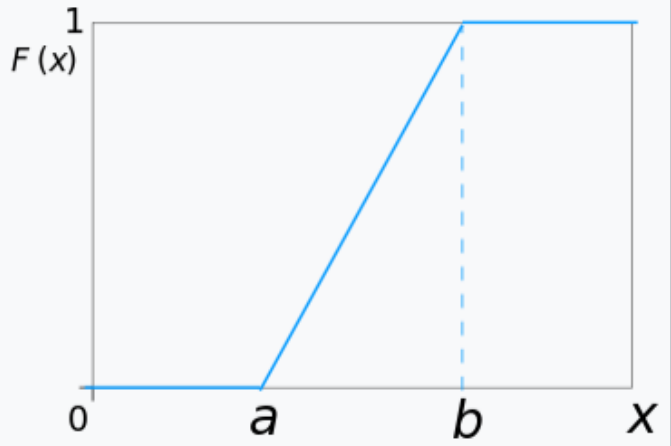
$$= \begin{cases} \text{if } x < u: & = 0 \\ \text{if } x \in [u,v]: & p \cdot \frac{x-u}{v-u} + (1-p)\,0 = p\,\frac{x-u}{v-u} \\ \text{if } x \in [v,w]: & p + (1-p)\frac{x-v}{w-v} \\ \text{if } x > w: & = 1 \end{cases}$$

sanity check
- $= p$ at $x = v$
- $= p$ at $x = v$ ✓

by the Law of Total Probability

$$\mathbb{P}(X \in A) = \sum_y \mathbb{P}(X \in A | Y = y)\mathbb{P}(Y = y)$$

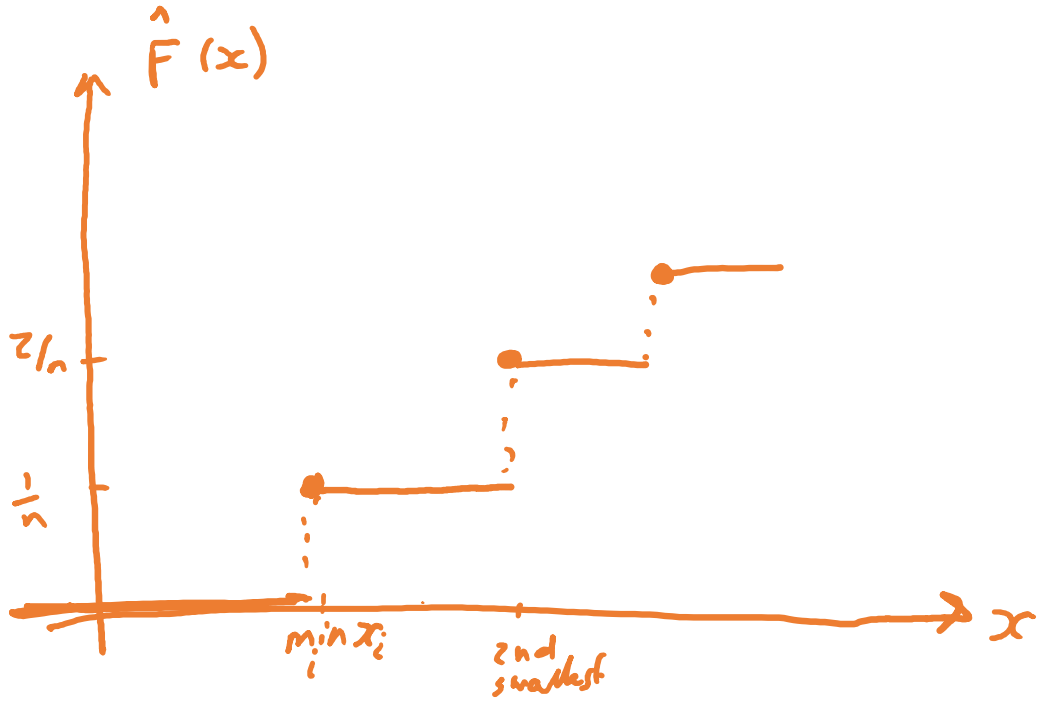
Wikipedia: Uniform distribution

Cumulative distribution function	
	
Notation	$\mathcal{U}_{[a,b]}$
Parameters	$-\infty < a < b < \infty$
Support	$x \in [a, b]$
PDF	$\begin{cases} \frac{1}{b-a} & \text{for } x \in [a, b] \\ 0 & \text{otherwise} \end{cases}$
CDF	$\begin{cases} 0 & \text{for } x < a \\ \frac{x-a}{b-a} & \text{for } x \in [a, b] \\ 1 & \text{for } x > b \end{cases}$

ECDF

Given a dataset of numerical values $[x_1, x_2, \ldots, x_n]$, the empirical cumulative distribution function or ecdf is

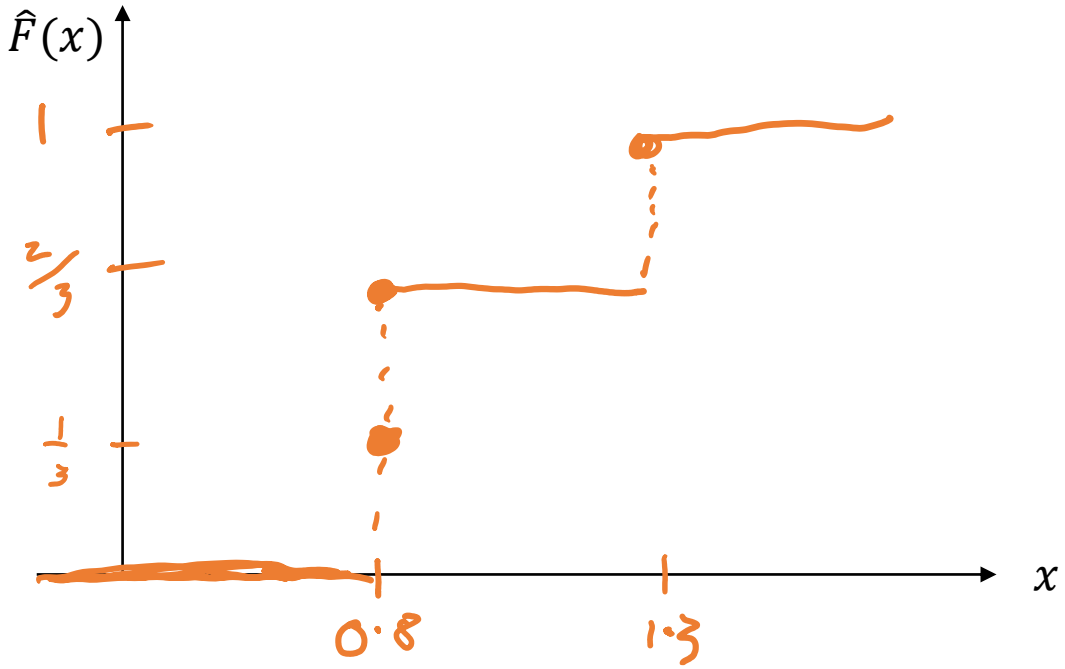
$$\hat{F}(x) = \frac{1}{n}\begin{pmatrix}\text{how many datapoints}\\ \text{there are } \leq x\end{pmatrix}$$




```
x = [...]
F = np.arange(1, len(x)+1) / len(x)
plt.plot(np.sort(x), F, drawstyle='steps-post')
```

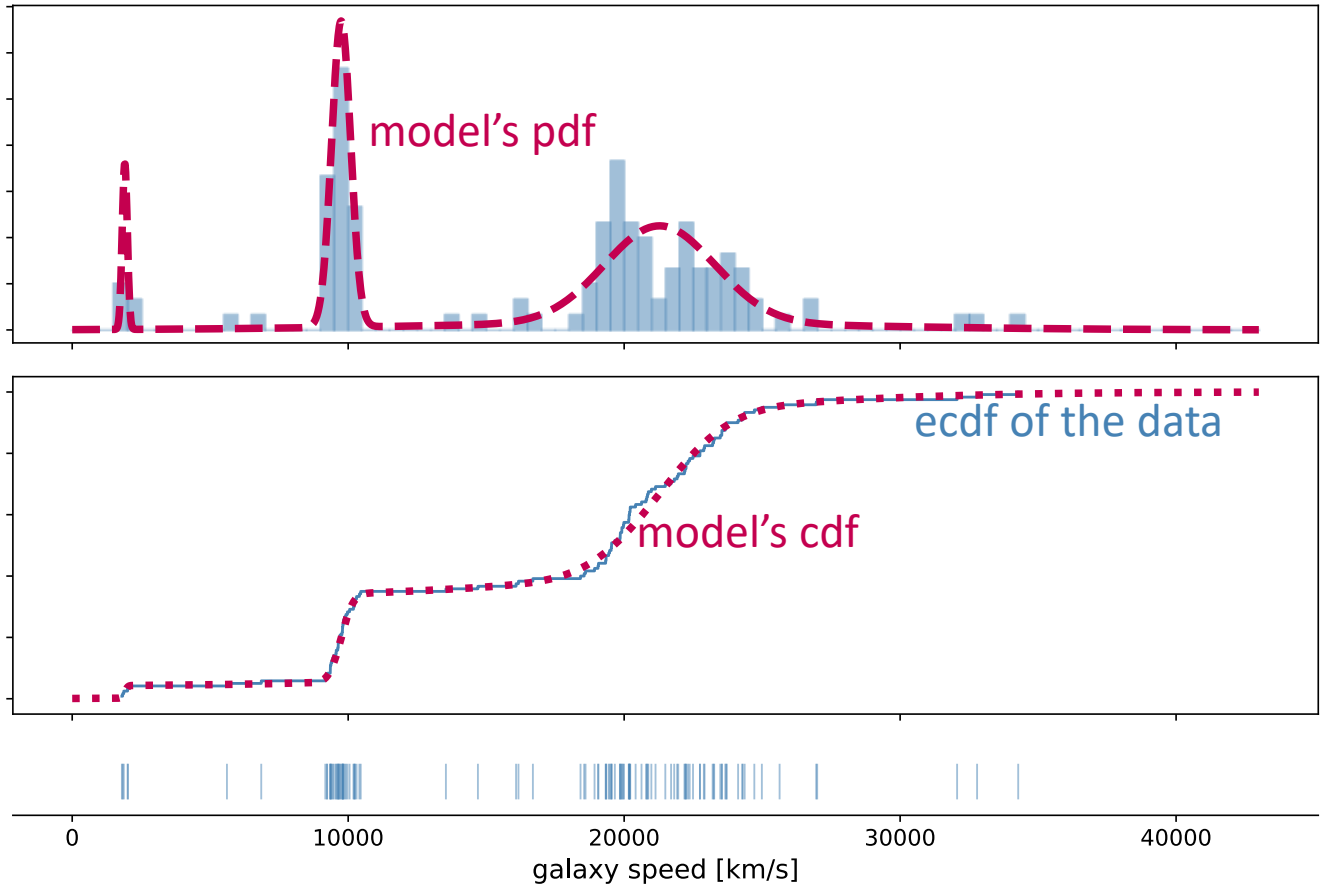
What if there are repeated values in the dataset, e.g.

```
x = [0.8, 0.8, 1.3]
```



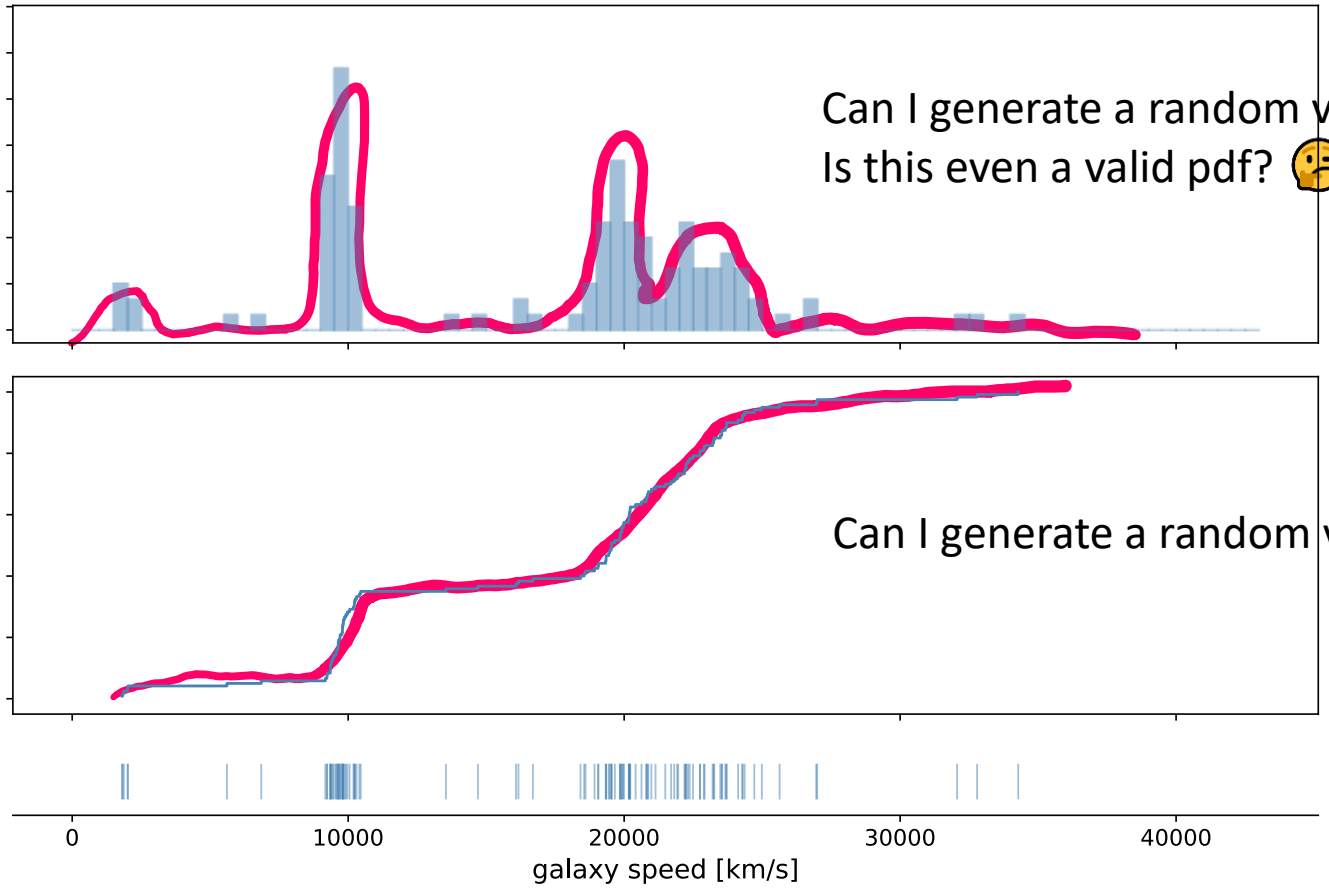

```
x = [...]
F = np.arange(1, len(x)+1) / len(x)
plt.plot(np.sort(x), F, drawstyle='steps-post')
```

(This code will plot an extra point at (0.8, 1/3), but who cares? The plot is still correct.)



fitted Gaussian mixture model

But can I find a better-fitting distribution?



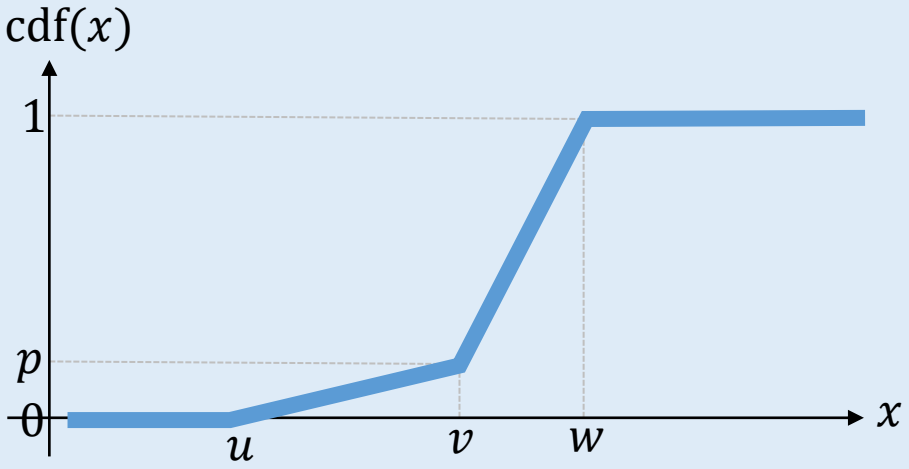
Can I generate a random variable with this pdf?
Is this even a valid pdf? 🤔 🤷‍♀️

Can I generate a random variable with this cdf?

It's certainly a valid cdf:
it starts at 0, goes to 1,
and is non-decreasing.

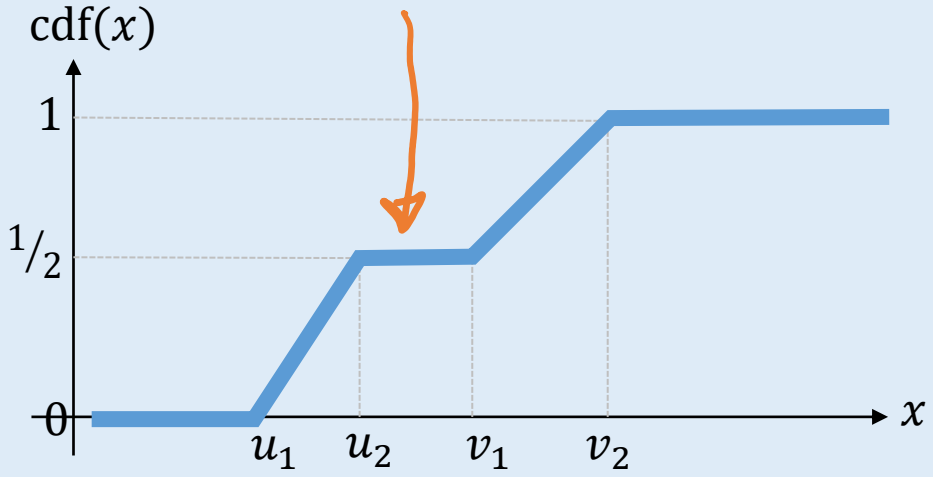
Let's build up our skills at turning cdf plots into code.

But can I find a better-fitting distribution?




```
def rx(u,v,w,p):
    k = np.random.choice(["left","right"], [p,1-p])
    if k == "left":
        return np.random.uniform(u,v)
    else:
        return np.random.uniform(v,w)
```

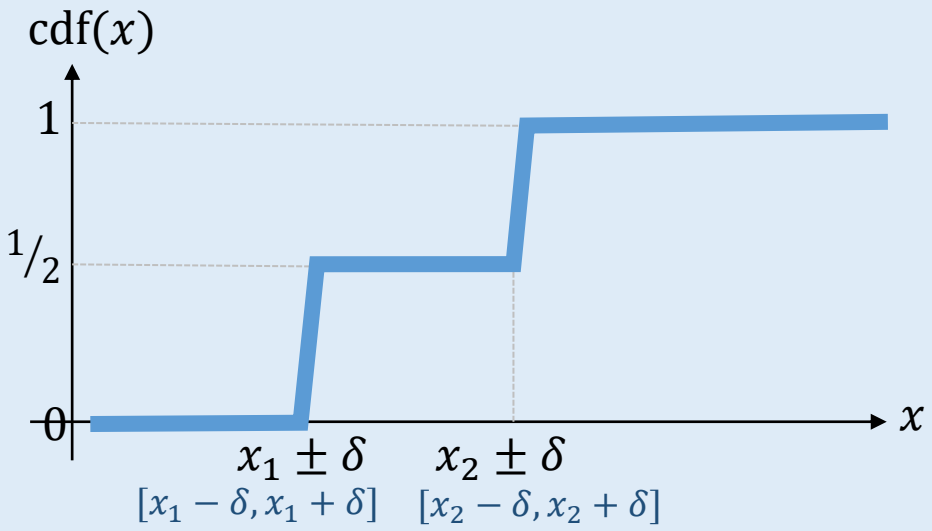
cdf flat ⇒ pdf = 0 ⇒ don't generate values in $[u_2, v_1]$



QUESTION. How would you complete this code?

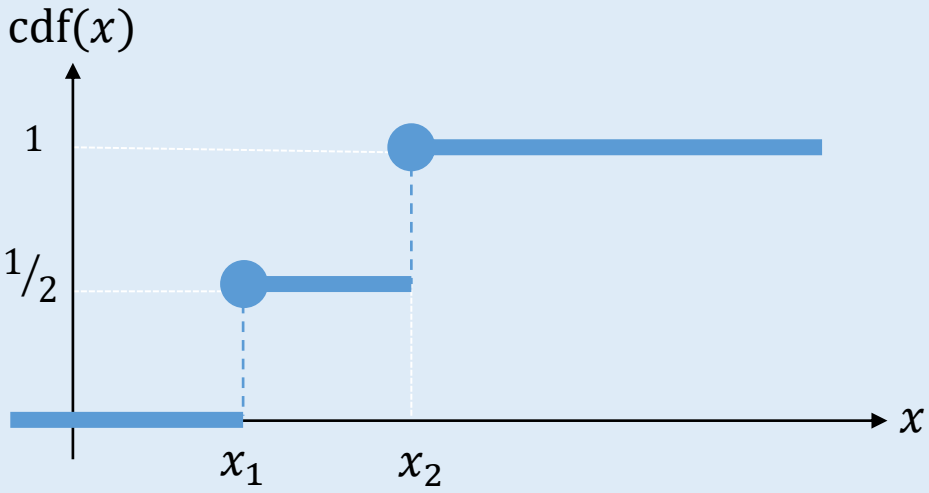
```
def rx(u1,u2,v1,v2):
    # pick either left or right, with equal probability
    k = np.random.choice(["left","right"])
    if k == "left":
        return np.random.uniform(u1,u2)
    else:
        return np.random.uniform(v1,v2)
```

fix $p = \frac{1}{2}$




```
def rx(x1,x2,δ):
    k = np.random.choice(["left","right"])
    if k == "left":
        return np.random.uniform(x1 − δ,x1 + δ)
    else:
        return np.random.uniform(x2 − δ,x2 + δ)
```

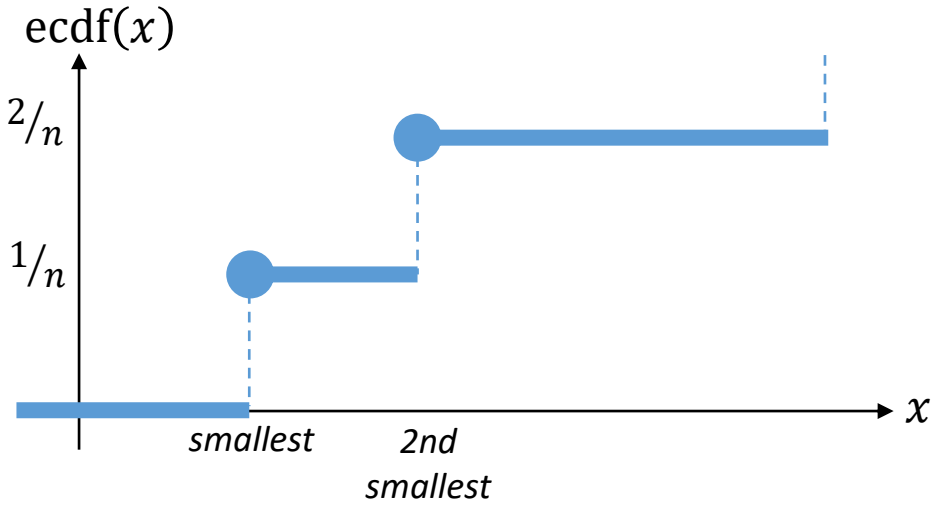
Extreme case: limit as $\delta \to 0$




```
def rx(x1,x2):
    k = np.random.choice(["left","right"])
    if k == "left":
        return x1
    else:
        return x2
```

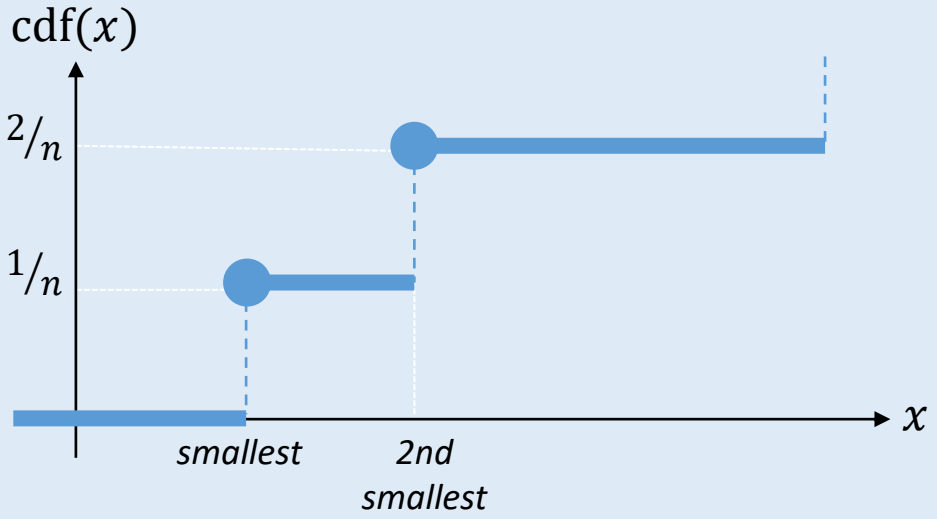
Or, more simply, just use np.random.choice([x1, x2])

This obviously generalizes to lists.



Recall the empirical distribution for a dataset $\vec{x} = (x_1, x_2, \ldots, x_n)$:

$$\text{ecdf}(x) = \frac{1}{n}(\#\text{points} \leq x)$$



To generate a random variable $\hat{X}$ whose cdf matches exactly this step function:

```
def rxhat([x_1, ..., x_n]):
    return np.random.choice([x_1, ..., x_n])
```

This is a perfect fit to the dataset!

The empirical distribution

Given a dataset $[x_1, x_2, \dots, x_n]$ let $\hat{X}$ be the random variable obtained by picking one of the x_i at random. (This is a discrete random variable.) It satisfies

$$\mathbb{P}(\hat{X} \in A) = \frac{1}{n}\sum_{i=1}^{n} 1_{x_i \in A}$$

$$\mathbb{E}\, h(\hat{X}) = \frac{1}{n}\sum_{i=1}^{n} h(x_i)$$

We say this random variable has *the empirical distribution of the dataset*.

This is a general definition that makes sense for any type of data (text, images, etc.), whereas the ecdf only applies to real-valued random variables.

Let $I \in \{1, \dots, n\}$ be the index we pick, i.e. $\hat{X} = x_I$. Then

$$\mathbb{E}h(\hat{X}) = \mathbb{E}h(x_I) = \sum_{i=1}^{n} h(x_i)\, \mathbb{P}(I = i) = \frac{1}{n}\sum_{i} h(x_i)$$

APPLICATION 1

If we're looking for a distribution to fit our dataset, just use $\hat{X}$ — the empirical distribution is a perfect fit, no need to bother fitting the parameters of an off-the-shelf random variable.

QUESTION. Is there any need at all for fitting / mle?

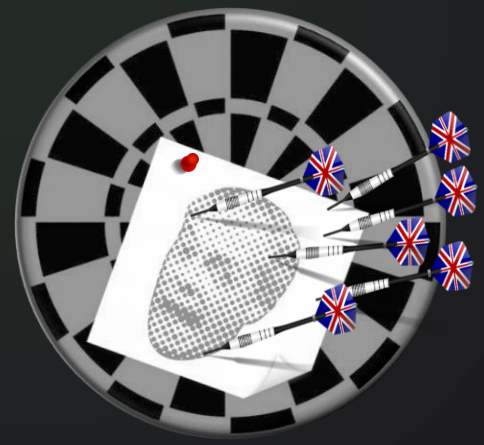


APPLICATION 2

If there's a random variable X that we'd like to work with, we can just take samples from X then use $\hat{X}$ instead — this saves us the bother of tricky integrals, by Monte Carlo approximation.

Monte Carlo

$$\mathbb{E}\, h(X) \approx \frac{1}{n}\sum_{i=1}^{n} h(x_i) = \mathbb{E}\, h(\hat{X})$$



PART III

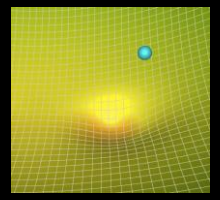
Inference / generalization

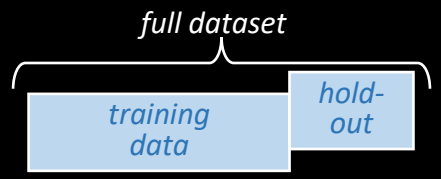


I tossed four coins and got $x = 1$ head. My data model is

$$X \sim \text{Bin}(4, \theta)$$

What is it legitimate for me to infer?

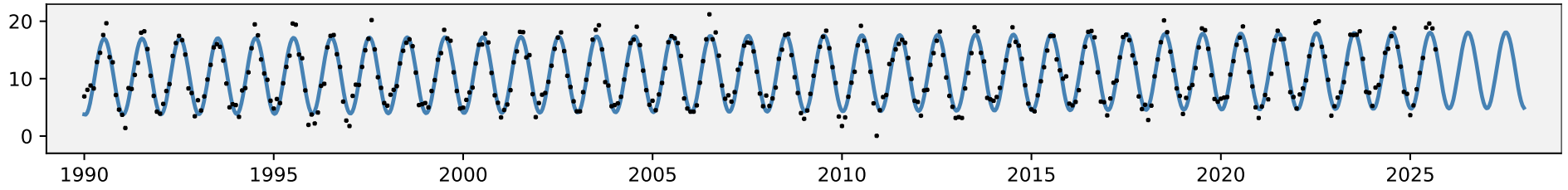


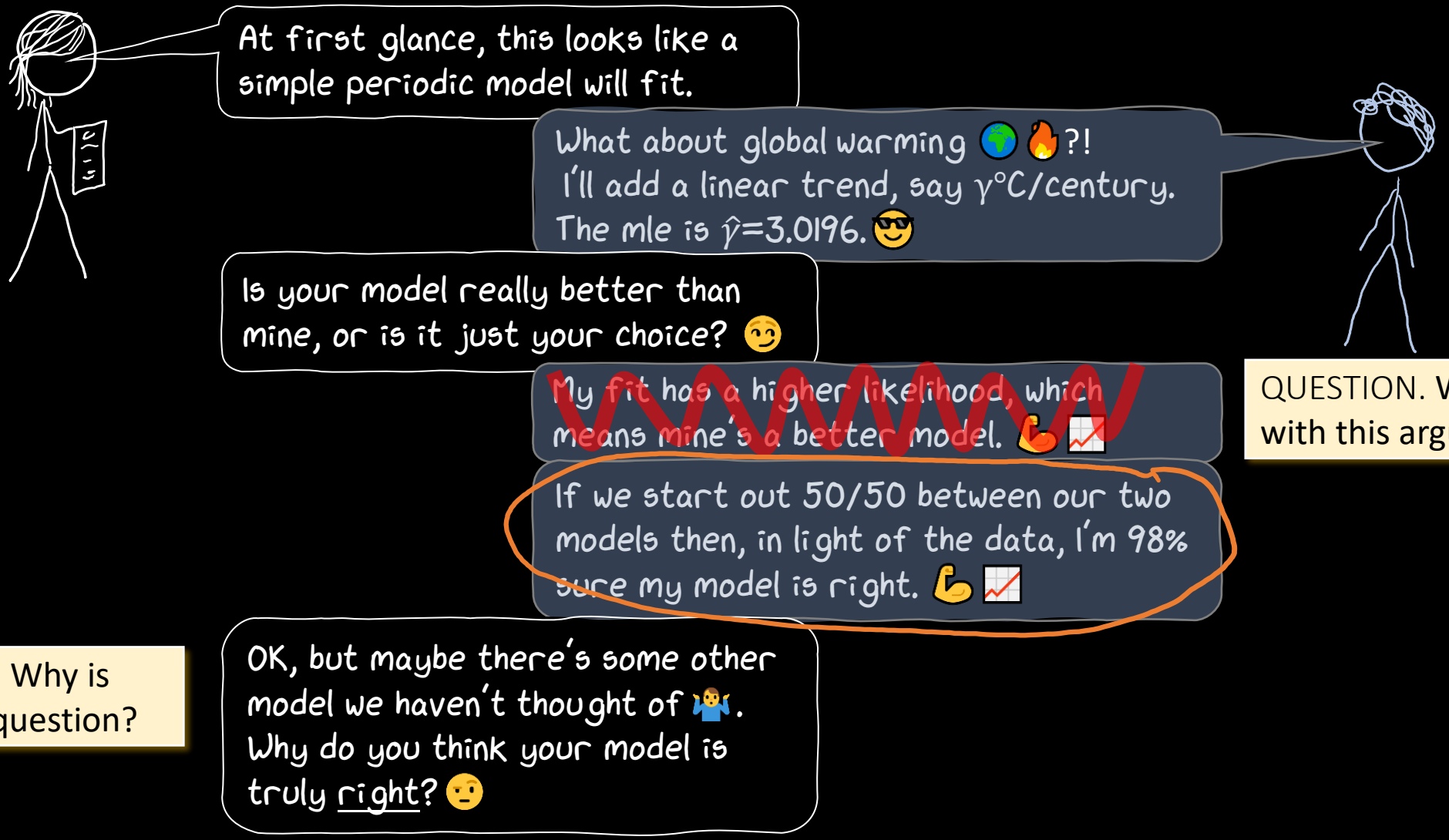




What can I say about ...	FITTING	HOLDOUT	BAYESIANISM	FREQUENTISM
... which model to choose?	≈	👍	👍	
... parameter values?	≈	≈	👍	
... predicted outcomes?			👍	
... whether my model is adequate?	✗	✗	✗	

Iterative model development





QUESTION. What's wrong with this argument?

QUESTION. Why is this a silly question?



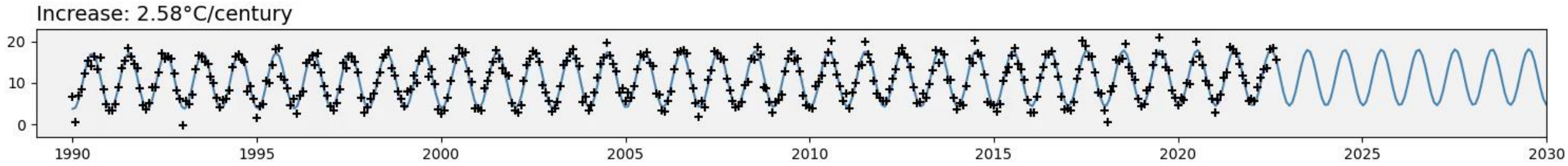
I saw x=1. Let me go figure out how likely is each possible explanation $\Theta=\theta$.

Bayes's rule:

$$\Pr_\Theta(\theta|x) = \kappa \Pr_\Theta(\theta) \Pr_X(x|\Theta = \theta)$$

I saw x=1, $\hat{\theta}=1/4$, **IN THIS REALITY.** What was $\hat{\theta}$ in other dimensions of the multiverse?







Climate confidence challenge.

Find a 95% confidence interval for γ, for Cambridge from 1985 to the present. (It's your choice how to simulate the multiverse.)

Please submit your answer on Moodle by Tuesday 10 November

Confidence intervals via resampling

A full dataset, not just a single reading

Given a dataset x,

1. Decide on a readout function $t(x)$

   e.g. $t(x)$ = my estimate $\hat{y}$ when I fit my model to the dataset x.

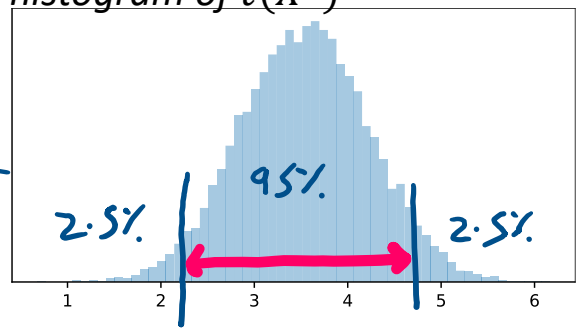
   We can use any readout we like, but it does have to be a function only of the dataset (i.e. with no unknown parameters). Such a function is called a statistic.

2. *"Simulate a multiverse of datasets."*
   - Fit a model for the dataset
   - Let X^* be a random synthetic dataset, generated from the fitted model
   - Simulate many synthetic datasets

3. Compute t for each dataset, and report the spread of t

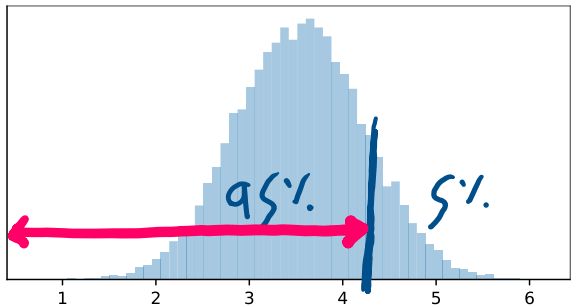
   for example with a histogram or a confidence interval

histogram of $t(X^*)$



Two-sided 95% confidence interval

```
np.quantile(tsamples, [.025, .975])
```



One-sided 95% confidence interval

```
np.quantile(tsamples, [0,.95])
```

Example 9.2.1.

We are given a dataset

$$x = [4.3, 5.1, 6.1, 6.8, 7.4, 8.8, 9.9]$$

which we decide to model as independent samples from $N(\mu, \sigma^2)$. Find a 95% confidence interval for $\hat{\mu}$.

QUESTION. What readout statistic is the question asking about?

the question wants the readout $\hat{\mu}$, and for this model $\hat{\mu}$ = sample mean.

```
# 1. Define a readout statistic
def t(x): return np.mean(x)

# 2. To generate a synthetic dataset ...
μhat = np.mean(x)
σhat = np.sqrt(np.mean((x-μhat)**2))
def rx_star():
    return np.random.normal(loc=μhat, scale=σhat, size=len(x))

# 3. Sample the readout statistic, and report its spread
t_ = [t(rx_star()) for _ in range(10000)]
lo,hi = np.quantile(t_, [.025, .975])
```

Fit a $N(\mu, \sigma^2)$ model to my dataset

synthesize a new dataset by sampling the right number of points from my fitted $N(\hat{\mu}, \hat{\sigma}^2)$

Example 9.2.1.

We are given a dataset

$$x = [4.3, 5.1, 6.1, 6.8, 7.4, 8.8, 9.9]$$

which we decide to model as independent samples from $N(\mu, \sigma^2)$. Find a 95% confidence interval for $\hat{\mu}$.

```
# 1. Define a readout statistic
def t(x): return np.mean(x)

# 2. To generate a synthetic dataset ...
def rx_star():
    return np.random.choice(x, size=len(x))
```

i.e. to simulate what the dataset might have been, simply sample n values from the empirical distribution (which is a perfect fit to the data)

```
# 3. Sample the readout statistic, and report its spread
t_ = [t(rx_star()) for _ in range(10000)]
lo,hi = np.quantile(t_, [.025, .975])
```

Confidence intervals via parametric resampling

Given a dataset x
and a parametric probability model $\Pr(x;\theta)$

1. Decide on a readout function $t(x)$
2. *"Simulate a multiverse of datasets."*
   - Fit a model for the dataset i.e. estimate $\hat{\theta}$
   - Let X^* be a random synthetic dataset, generated from the fitted model
   - Simulate many synthetic datasets
3. Compute t for each dataset, and report the spread of t
   for example with a histogram
   or a confidence interval

Confidence intervals via non-parametric resampling

Given a dataset x,

1. Decide on a readout function $t(x)$
2. *"Simulate a multiverse of datasets."*
   - Use the empirical distribution as a model
   - Let X^* be a random synthetic dataset, generated from the fitted model
   - Simulate many synthetic datasets
3. Compute t for each dataset, and report the spread of t
   for example with a histogram
   or a confidence interval

Exercise 9.2.3 (Comparing groups).

We are given data $x = [x_1, \dots, x_m]$ which we believe is $N(\mu, \sigma^2)$ and further data $y = [y_1, \dots, y_n]$ which we believe is $N(\mu + \delta, \sigma^2)$. Find a 95% confidence interval for $\hat{\delta}$.

The MLEs for μ, δ, σ are what you calculated in Example Sheet 1 question 4:

$\hat{\mu} = \bar{x}$

$\hat{\delta} = \bar{y} - \bar{x}$

$\hat{\sigma} = \cdots$

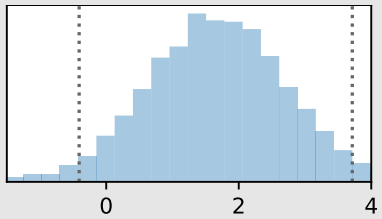
```
1  x = [4.3, 5.1, 6.1, 6.8, 7.4, 8.8, 9.9]
2  y = [8.3, 8.5, 8.9]
3  m,n = len(x), len(y)

4  # 1. Define the readout statistic
5  def t(x,y): return np.mean(y) - np.mean(x)

6

7  # 2. To generate a synthetic dataset ...
8  μ̂,δ̂ = np.mean(x), np.mean(y) - np.mean(x)
9  σ̂ = np.sqrt((np.sum((x-μ̂)**2 + np.sum((y-μ̂-δ̂)**2))/(m+n))
10 def rxy_star():
11     return (np.random.normal(loc=μ̂, scale=σ̂, size=m),
12             np.random.normal(loc=μ̂ + δ̂, scale=σ̂, size=n))

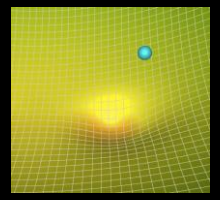
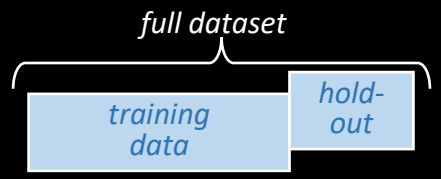
13 # 3. Sample the readout statistic, and report its spread
14 t_ = [t(*rx_star()) for _ in range(10000)]
15 lo,hi = np.quantile(t_, [.025, .975])
16 plt.hist(t_)
```



THE TOTALITARIAN RULE

When you fit a model by maximizing $\Pr(\text{data}; \text{params})$ use ALL the data and ALL the parameters.

When you resample, resample the FULL data.

What can I say about ...	 FITTING	 HOLDOUT	 BAYESIANISM	 FREQUENTISM
... which model to choose?	≈	👍	👍	
... parameter values?	≈	≈	👍	Canvas estimates across the multiverse
... predicted outcomes?			👍	
... whether my model is adequate?	✗	✗	✗	?