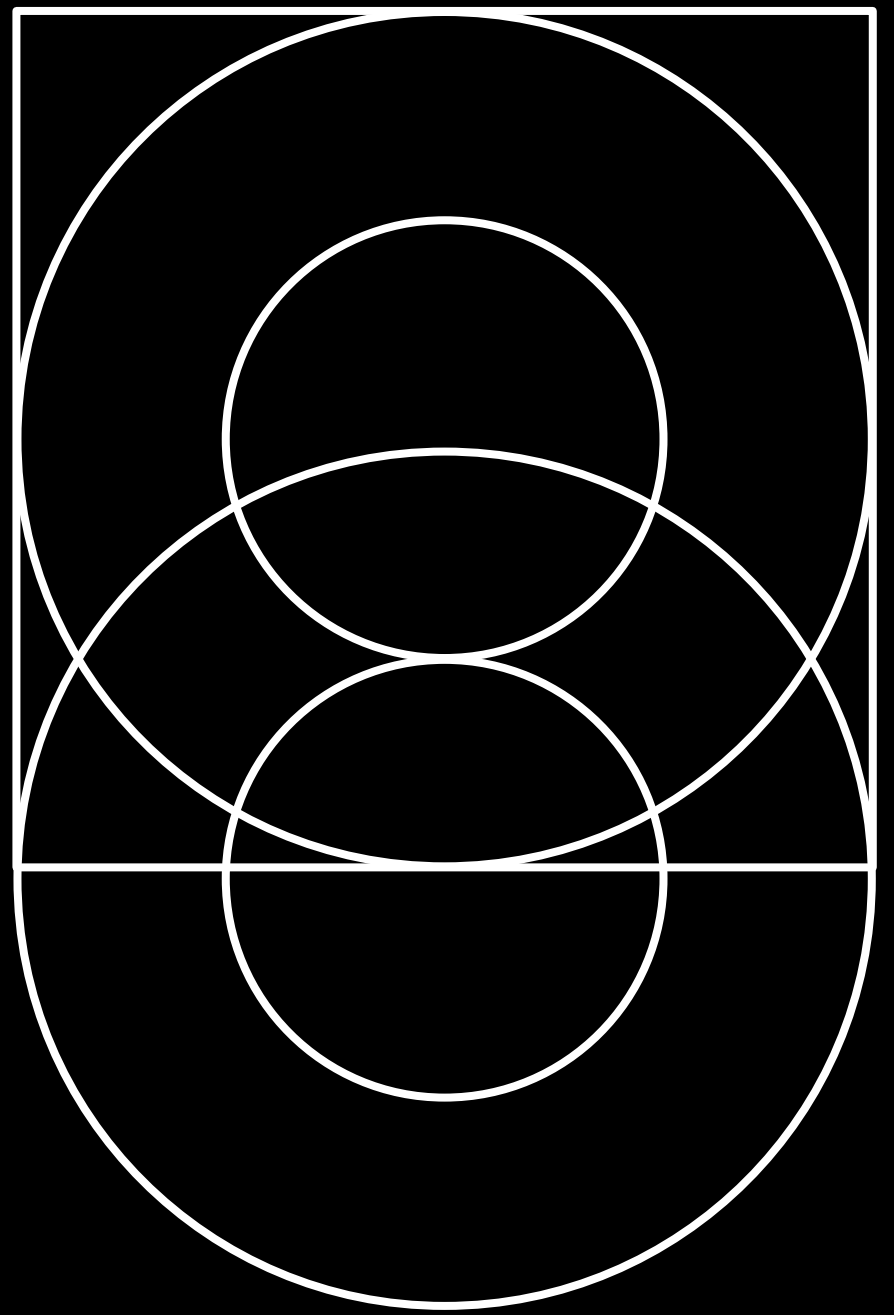
Unit8™

Bayesian inference and PyMC3

Sebastian Gay

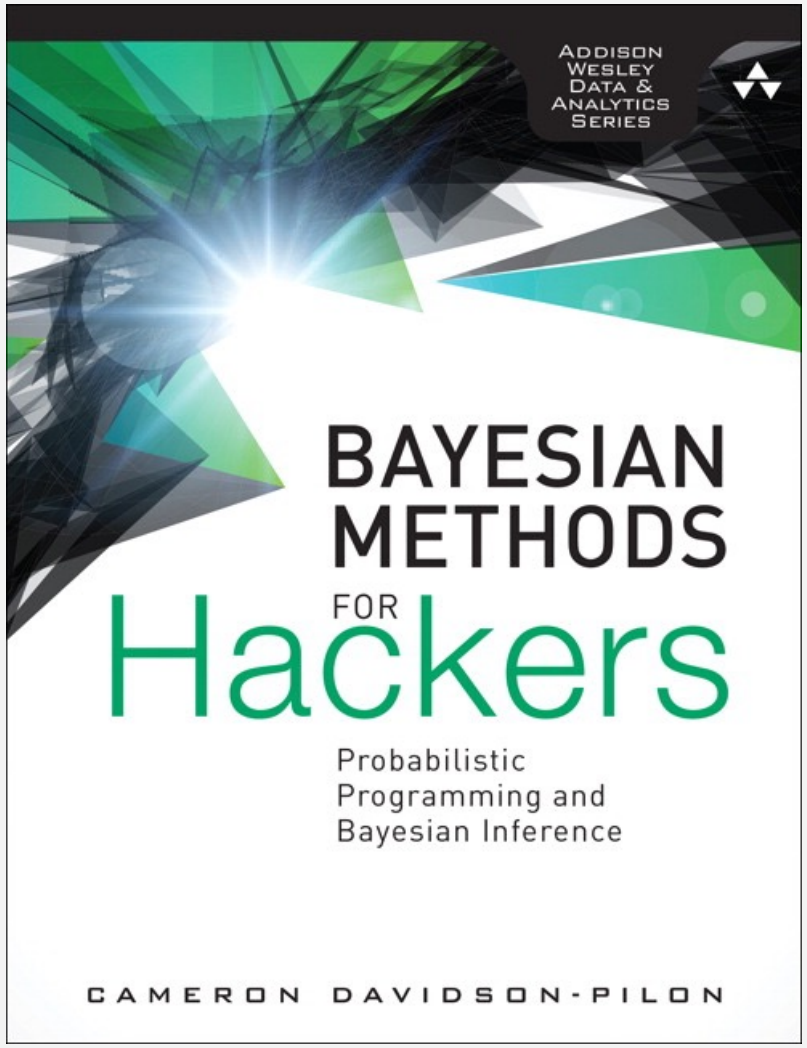
19/05/2022

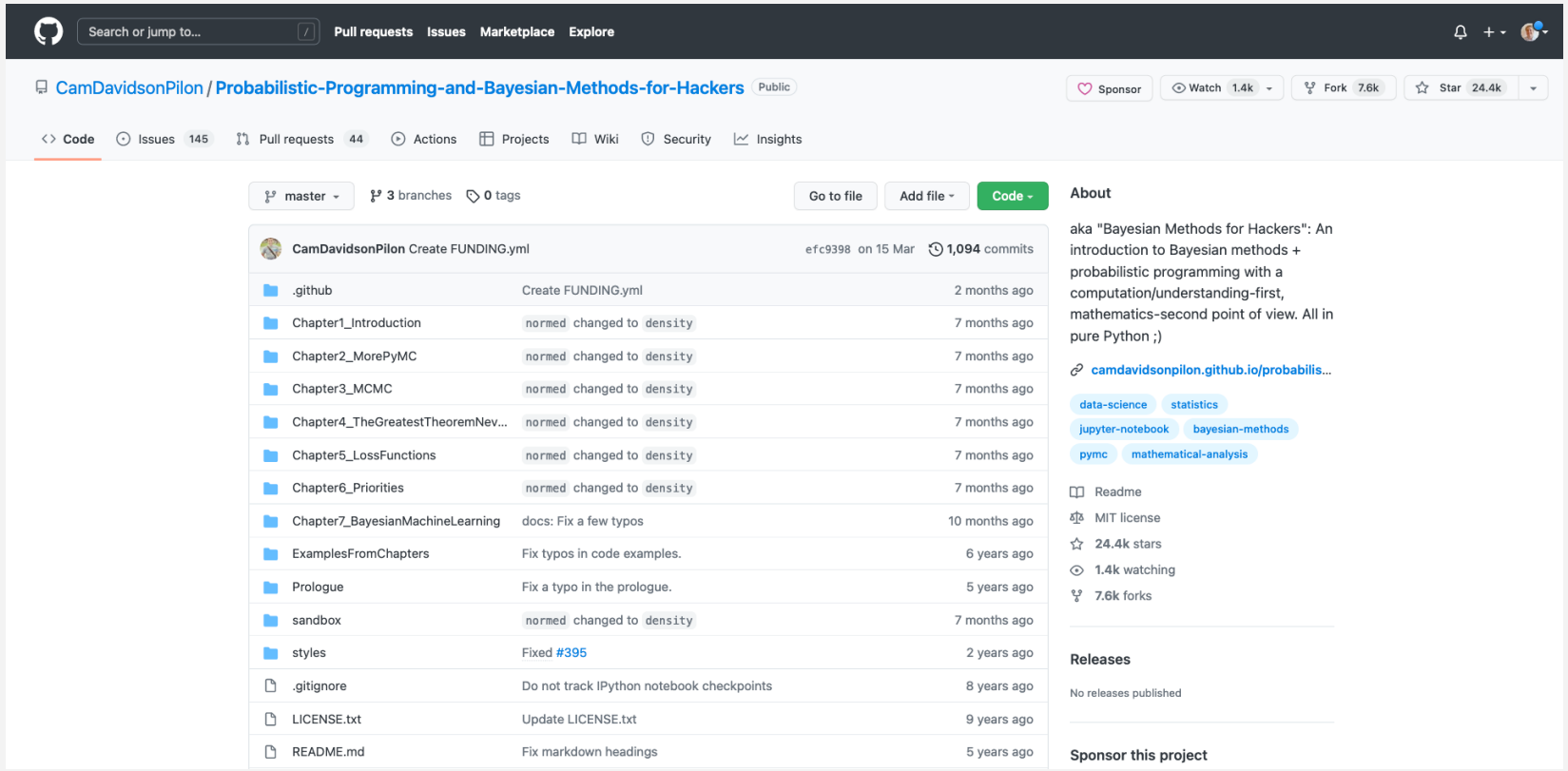


Outline

- Bayesian vs. Frequentist statistics
- Bayes' formula, interpretation, and dropping the denominator
- Introducing PyMC3 and text messaging rates
- Understanding Markov Chain Monte Carlo (MCMC)
- Applying MCMC to text messaging rates
- Industry applications of Bayesian inference

Check it out:



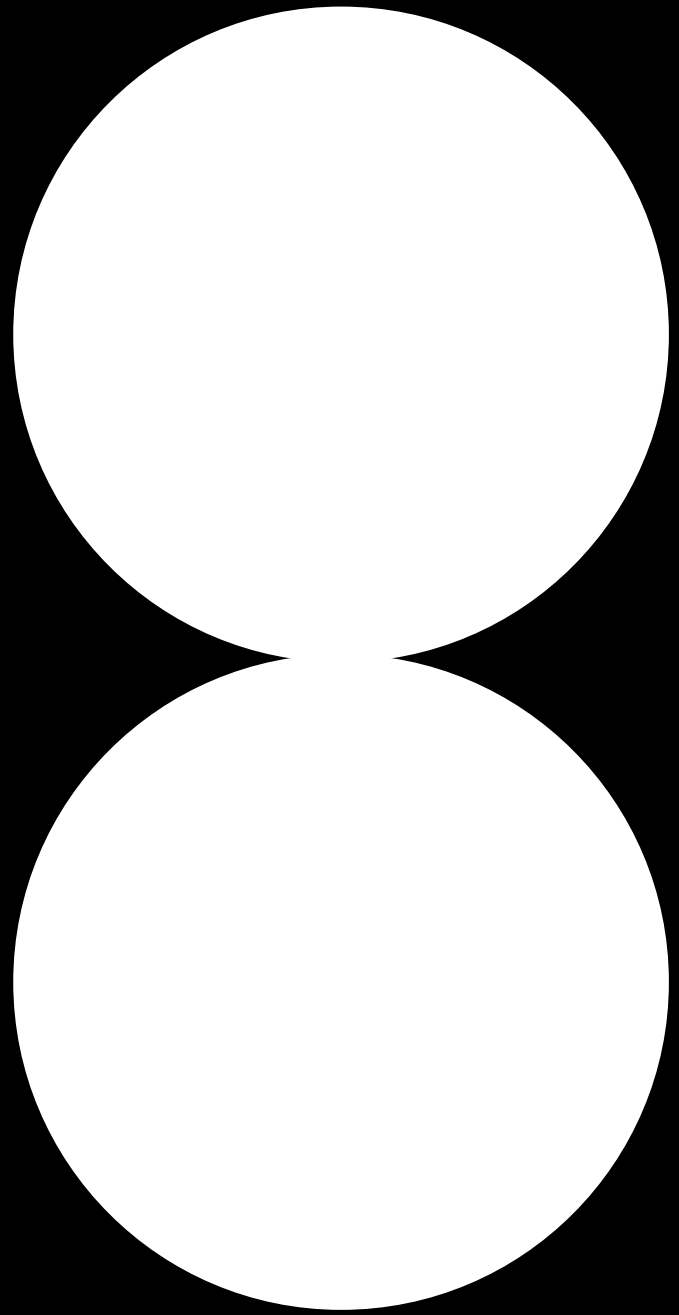


Unit8

unit8.com

Unit8™

Bayesian vs Frequentist Statistics



digital natives

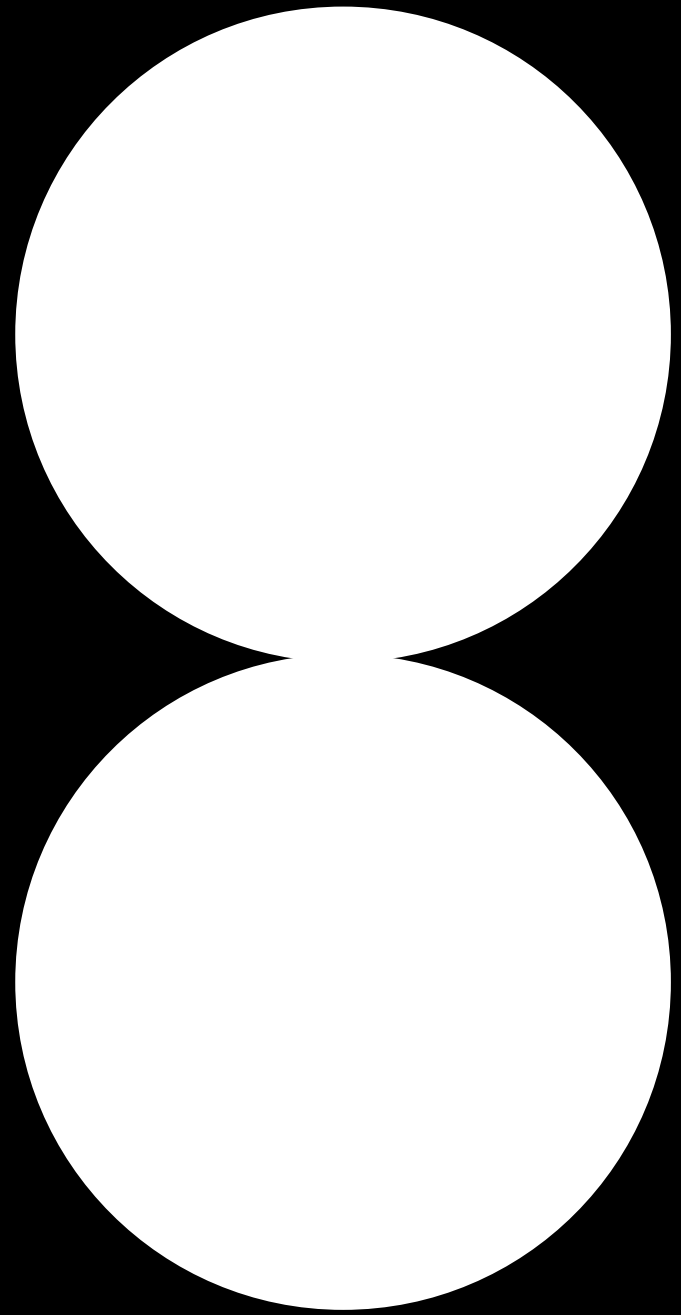
unit8.com

The 5 CHF coin

Which to use?

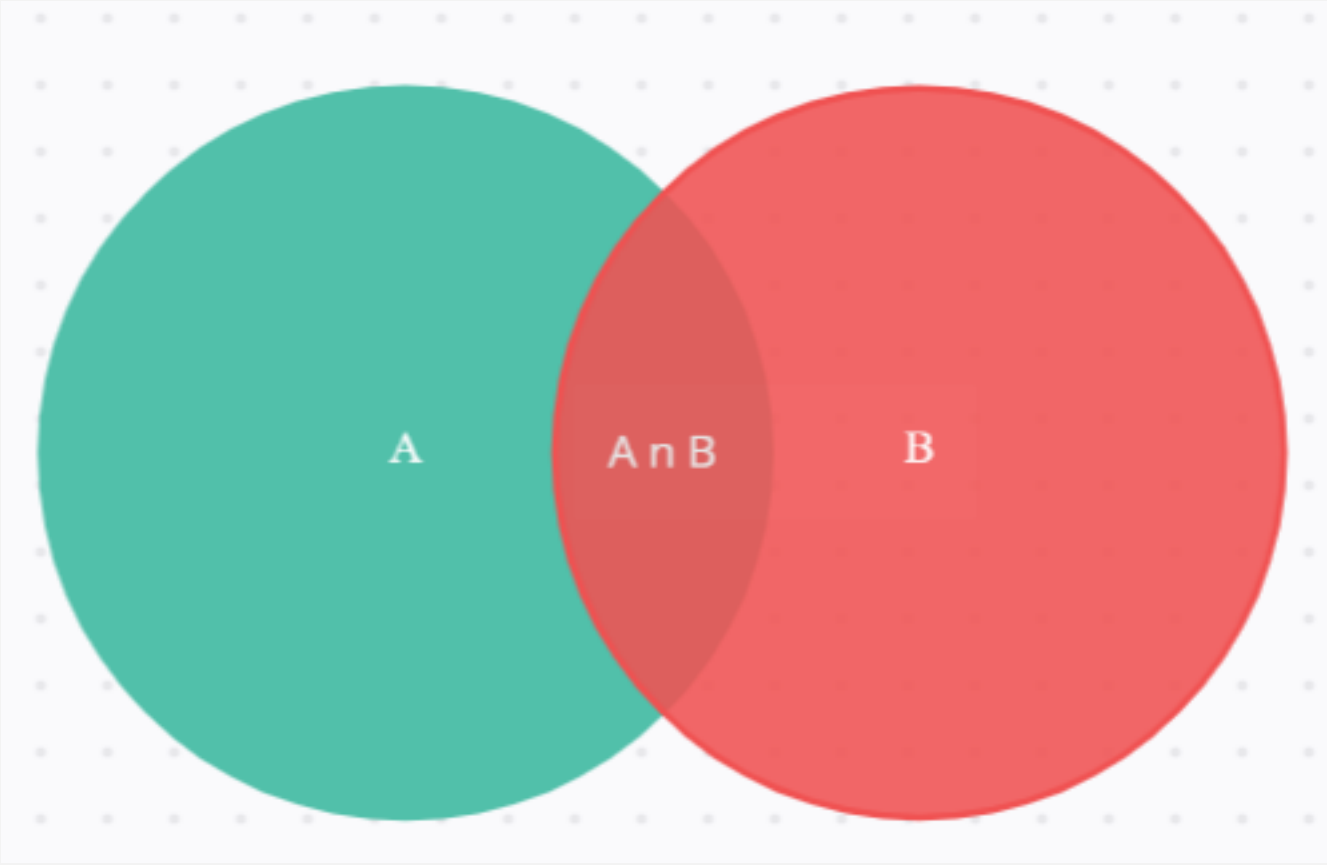
- What is fixed?
    - Frequentist: Parameters are fixed
    - Bayesian: Data is fixed
- When to use what?

Bayes’ Formula



Deriving Bayes Formula

$$P(A \cap B) = P(A)\ P(B|A)$$
$$= P(B)\ P(A|B)$$
$$\Downarrow$$
$$P(B)\ P(A|B) = P(A)\ P(B|A)$$
$$P(A|B) = \frac{P(A)\ P(B|A)}{P(B)}$$



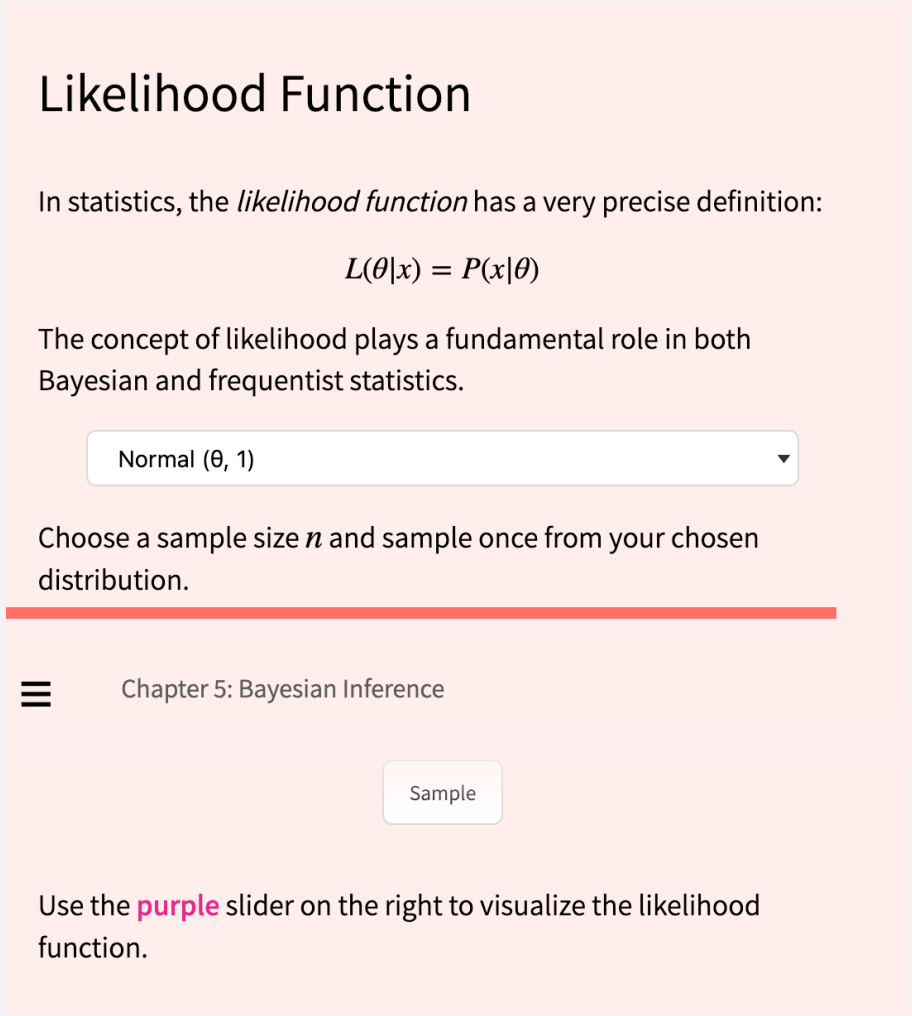
How does it help us?

$$\text{Model parameterised by } \theta$$
$$\text{Data } \boldsymbol{x}$$

$$P(\text{model}|\text{data}) = \frac{P(\text{model})\ P(\text{data}|\text{model})}{P(\text{data})}$$

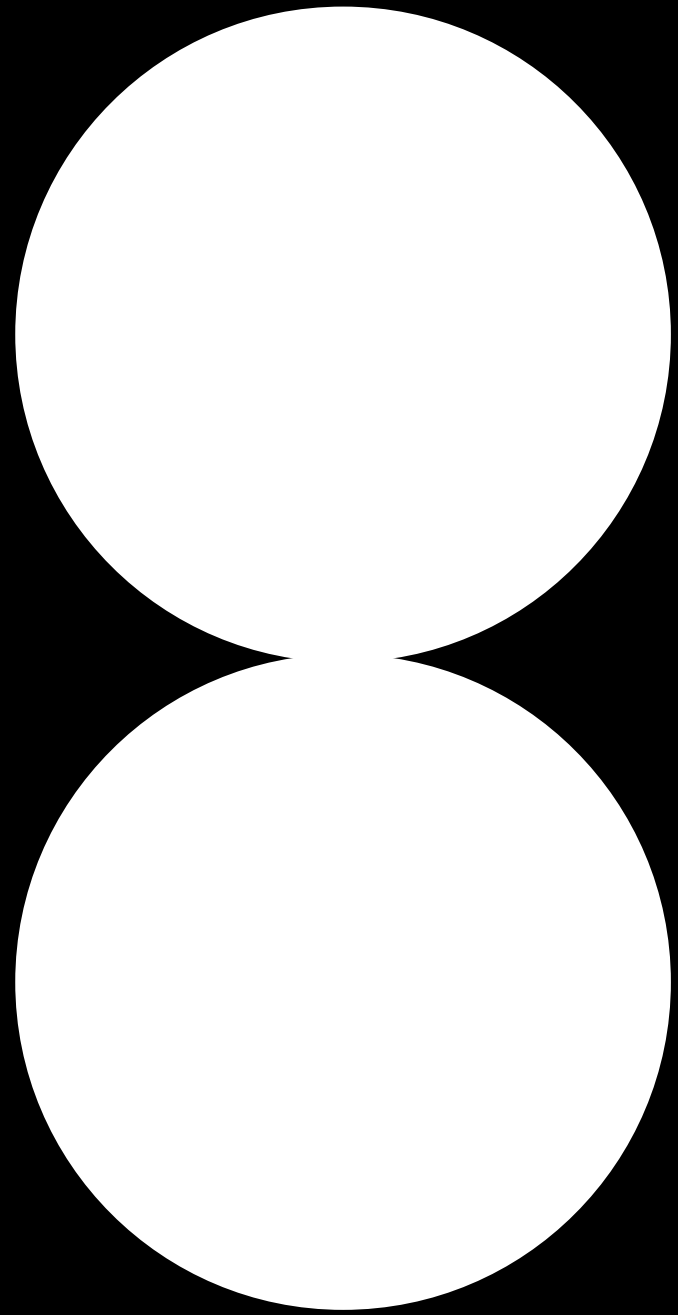
$$P(\theta|\boldsymbol{x}) = \frac{P(\theta)\ P(\boldsymbol{x}|\theta)}{P(\boldsymbol{x})}$$
$$\propto P(\theta)\ P(\boldsymbol{x}|\theta)$$

Visualising likelihood



Unit8™

PyMC



digital natives

unit8.com

Background

- PyMC: Initial development in 2003
- PyMC2: 2006-2010 (release). Fortran used for computation.
- PyMC3: 2013 release. Python + Theano (CUDA).
- PyMC3 → PyMC.
- Aesara backend: 2020: JAX/Numba

PyMC3

NUMFOCUS

OPEN CODE = BETTER SCIENCE

What does it look like?

```
import pymc3 as pm

with pm.Model() as model:
    alpha = 1.0/count_data.mean()  # Recall count_data is the
                                   # variable that holds our txt counts
    lambda_1 = pm.Exponential("lambda_1", alpha)
    lambda_2 = pm.Exponential("lambda_2", alpha)

    tau = pm.DiscreteUniform("tau", lower=0, upper=n_count_data - 1)
```

```
with model:
    step = pm.Metropolis()
    trace = pm.sample(10000, tune=5000, step=step, return_inferencedata=False)
```

```
lambda_1_samples = trace['lambda_1']
lambda_2_samples = trace['lambda_2']
tau_samples = trace['tau']
```

`model` is clearly not just an alias... So what is the `with` about?

Whats with `with`?

The `with` statement is called a context manager

```
with open('some_file', 'w') as opened_file:
    opened_file.write('Hola!')
```

An equivalent class to a context manager would look like this.

1. Init
2. Enter
3. Exit

```
with File('demo.txt', 'w') as opened_file:
    opened_file.write('Hola!')
```

Biggest advantage: error handling.
Also: Object persistence.

```
class File(object):
    def __init__(self, file_name, method):
        self.file_obj = open(file_name, method)
    def __enter__(self):
        return self.file_obj
    def __exit__(self, type, value, traceback):
        self.file_obj.close()
```

Context Manager in PyMC

- The bottom line: pythonic way to associate variables with independent models

- Each model's variables are defined as distributions. The goal is to improve our estimates of these variables.

```python
class Model(WithMemoization, metaclass=ContextMeta):
    """Encapsulates the variables and likelihood factors of a model.
```

```python
class WithMemoization:
    def __hash__(self):
        return hash(id(self))

    def __getstate__(self):
        state = self.__dict__.copy()
        state.pop("_cache", None)
        return state

    def __setstate__(self, state):
        self.__dict__.update(state)
```

```python
class ContextMeta(type):
    """Functionality for objects that put themselves in a context using
    the `with` statement.
    """

    def __new__(cls, name, bases, dct, **kwargs):  # pylint: disable=unused-arg
        "Add __enter__ and __exit__ methods to the class."

        def __enter__(self):
            self.__class__.context_class.get_contexts().append(self)
            # self._aesara_config is set in Model.__new__
            self._config_context = None
            if hasattr(self, "_aesara_config"):
                self._config_context = aesara.config.change_flags(**self._aesar
                self._config_context.__enter__()
            return self

        def __exit__(self, typ, value, traceback):  # pylint: disable=unused-ar
            self.__class__.context_class.get_contexts().pop()
            # self._aesara_config is set in Model.__new__
            if self._config_context:
                self._config_context.__exit__(typ, value, traceback)

        dct[__enter__.__name__] = __enter__
        dct[__exit__.__name__] = __exit__
```

Community Contributing About Examples Blog Search...

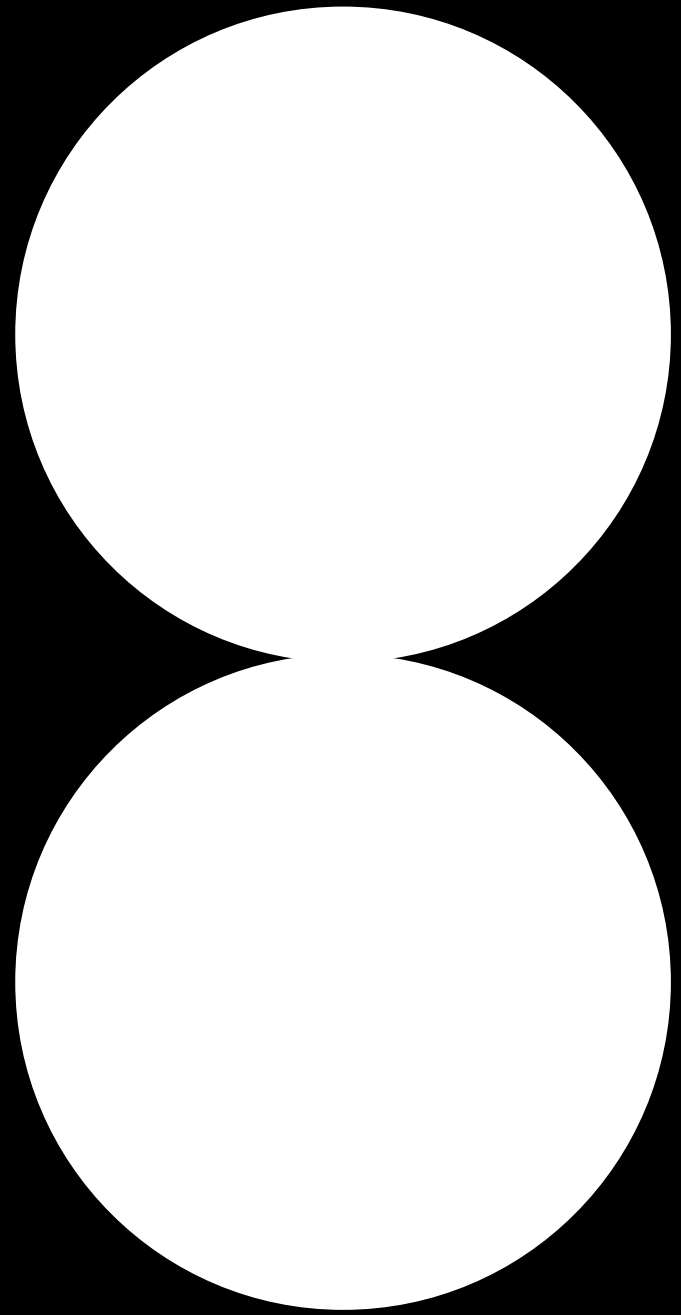
```python
        # you'll get a "TypeError: type() takes 1 or 3 arguments" exception.
        return super().__new__(cls, name, bases, dct)

    # FIXME: is there a more elegant way to automatically add methods to the cl
    # are instance methods instead of class methods?
    def __init__(
        cls, name, bases, nmspc, context_class: Optional[Type] = None, **kwargs
    ):  # pylint: disable=unused-argument
        """Add ``__enter__`` and ``__exit__`` methods to the new class automati
        if context_class is not None:
            cls._context_class = context_class
        super().__init__(name, bases, nmspc)
```

Text message problem

- Notebook 1: Model Creation
- Notebook 2a: Text message problem setup (from Bayesian methods for hackers)

Unit8™

Markov Chain Monte Carlo



digital natives

unit8.com

Generating a parameter distribution

How can we generate a parameter distribution?

- Symbolically:
    - Advantage – accurate
    - Disadvantage- incredibly difficult, often not possible for posterior distributions
- Numerically:
    - Advantage – easy for a computer to do
    - Disadvantage – requires sampling to converge to actual distribution

Markov Chain Monte Carlo

We take a position in parameter space.

1. Start at current position
2. Propose a new position
3. Accept/Reject the new position based on the position's adherence to the data and prior distributions.
4. Return to step 1.

After a large number of iterations, return all accepted positions.

Metropolis–Hastings algorithm

Take the target density (i.e. posterior distribution) $f(x)$

Start at the value $x^{(0)}$

Take a proposal distribution $q\left(x \mid x^{(i)}\right)$ (e.g. Gaussian)

1. Sample $x^* \sim q\left(x \mid x^{(j)}\right)$.
2. Calculate the acceptance probability

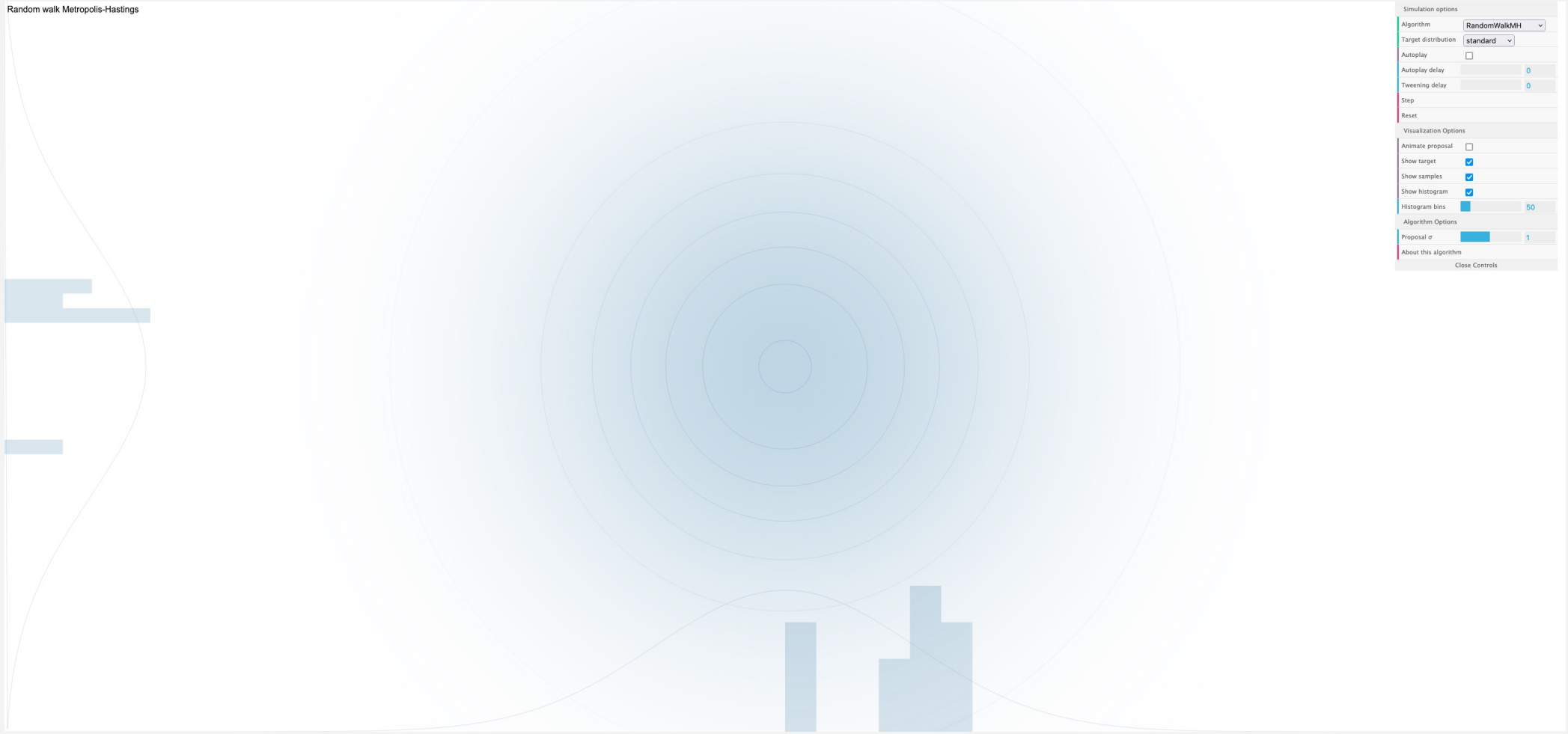
$$\rho\left(x^{(j)}, x^*\right) = \min\left\{1, \frac{f(x^*)}{f\left(x^{(j)}\right)} \frac{q\left(x^{(j)} \mid x^*\right)}{q\left(x^* \mid x^{(j)}\right)}\right\}$$

Cancels out with symmetric function

3. Let

$$x^{(i+1)} = \begin{cases} x^* & \text{with probability } \rho\left(x^{(j)}, x^*\right) \\ x^{(i)} & \text{otherwise} \end{cases}$$

Visualisating MCMC



Gibbs Sampler

- What happens when sampling from the joint distribution is difficult?
    - High dimensionality of the joint posterior
    - Unknown constants in posterior but not conditional

$$p(x,y) = \frac{1}{C} e^{-\frac{x^2y^2+x^2+y^2-8x-8y}{2}}$$

$$\mathcal{N}(\mu, \sigma)$$

$$p(y|x) = g(x) e^{-(y-\frac{4}{1+x^2})^2(\frac{1+x^2}{2})}$$

$$y|x \sim \mathcal{N}(\frac{4}{1+x^2}, \sqrt{\frac{1}{1+x^2}})$$

$$p(x|y) = g(y) e^{-(x-\frac{4}{1+y^2})^2(\frac{1+y^2}{2})}$$

$$x|y \sim \mathcal{N}(\frac{4}{1+y^2}, \sqrt{\frac{1}{1+y^2}})$$

- Enter the Gibbs Sampler- a special case of MH where the proposal is always accepted.

Gibbs sampler step

Suppose $\theta = (\theta_1, \ldots, \theta_K)$, then the following Gibbs sampler can be constructed

$$
\begin{aligned}
\theta_1^{(j)} &\sim p\left(\theta_1 \mid \theta_2^{(j-1)}, \ldots, \theta_K^{(j-1)}\right) \\
\theta_2^{(j)} &\sim p\left(\theta_2 \mid \theta_1^{(j)}, \theta_3^{(j-1)}, \ldots, \theta_K^{(j-1)}\right) \\
&\vdots \\
\theta_k^{(j)} &\sim p\left(\theta_k \mid \theta_1^{(j)} \ldots, \theta_{k-1}^{(j)}, \theta_{k+1}^{(j-1)}, \ldots, \theta_K^{(j-1)}\right) \\
&\vdots \\
\theta_K^{(j)} &\sim p\left(\theta_K \mid \theta_1^{(j)}, \ldots, \theta_{K-1}^{(j)}\right)
\end{aligned}
$$

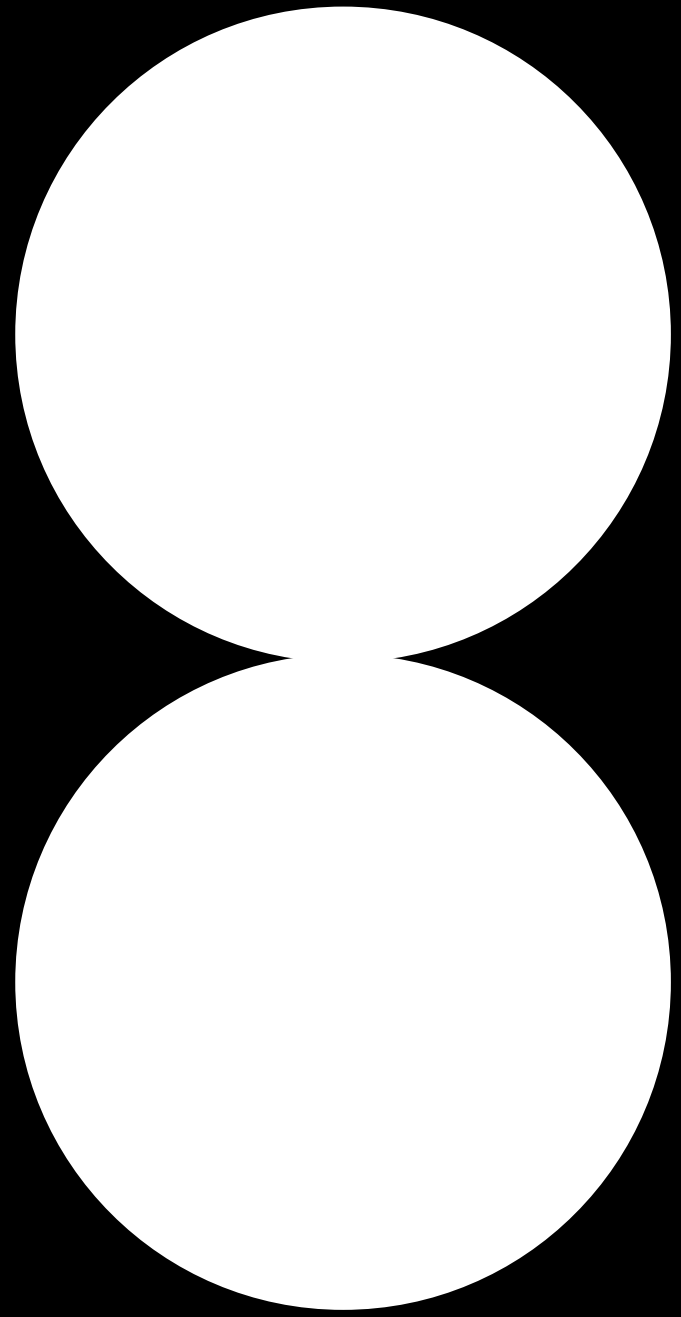
Burn in

- Our walk will take time to converge, based the starting point
- Because of this, a set of the first samples can sample from a low probabilty region
- Two solutions:
    - Run for a really long time- reduce the impact of this low probability oversampling
    - Remove these samples- eliminate their impact

Autocorrelation

- Measure of how well $\boldsymbol{\theta}_k$ predicts $\boldsymbol{\theta}_{k+t}$
- High autocorrelation means parameter search is repeating a path and not exploring parameter space
- Low autocorrelation is not necessary for convergence, but it is sufficient
- Solution is thinning- take every nth sample

Applying MCMC



Text messaging

- Notebook 2b

Generalised Linear Models –NBA fouls

Model definition

We denote by:

- N_d and N_c the number of disadvantaged and committing players, respectively,
- K the number of plays,
- k a play,
- y_k the observed call/noncall in play k,
- p_k the probability of a foul being called in play k,
- $i(k)$ the disadvantaged player in play k, and by
- $j(k)$ the committing player in play k.

We assume that each disadvantaged player is described by the latent variable:

- θ_i for $i = 1, 2, \ldots, N_d$,

and each committing player is described by the latent variable:

- b_j for $j = 1, 2, \ldots, N_c$.

Then we model each observation y_k as the result of an independent Bernoulli trial with probability p_k, wher

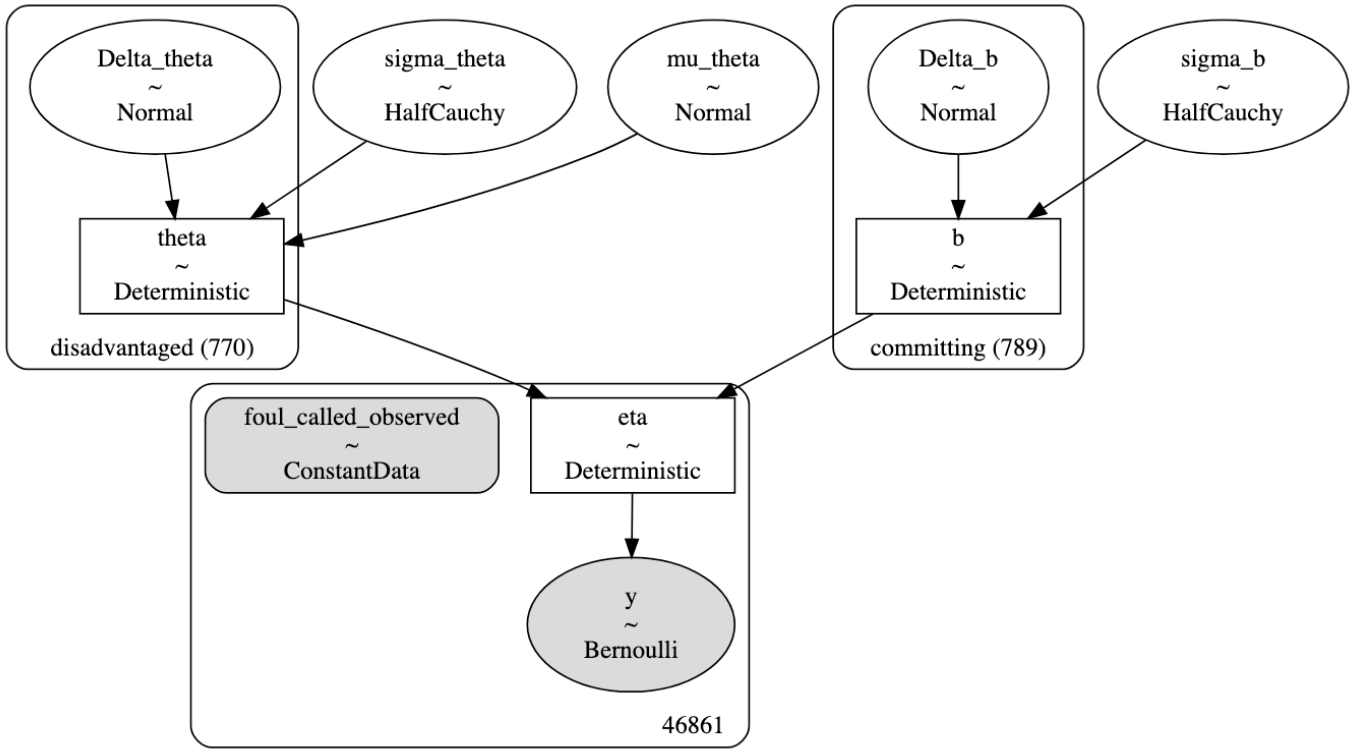
$$p_k = \text{sigmoid}(\eta_k) = \left(1 + e^{-\eta_k}\right)^{-1}, \quad \text{with} \quad \eta_k = \theta_{i(k)} - b_{j(k)},$$

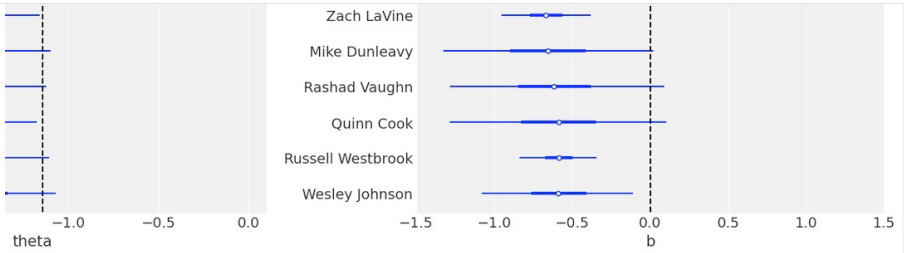
for $k = 1, 2, \ldots, K$, by defining (via a non-centered parametrisation)

$$\begin{aligned} \theta_i &= \sigma_\theta \Delta_{\theta,i} + \mu_\theta \sim \text{Normal}(\mu_\theta, \sigma_\theta^2), & i &= 1, 2, \ldots, N_d, \\ b_j &= \sigma_b \Delta_{b,j} \sim \text{Normal}(0, \sigma_b^2), & j &= 1, 2, \ldots, N_c, \end{aligned}$$

with priors/hyperpriors

$$\begin{aligned} \Delta_{\theta,i} &\sim \text{Normal}(0, 1), & i &= 1, 2, \ldots, N_d, \\ \Delta_{b,j} &\sim \text{Normal}(0, 1), & j &= 1, 2, \ldots, N_c, \\ \mu_\theta &\sim \text{Normal}(0, 100), \\ \sigma_\theta &\sim \text{HalfCauchy}(2.5), \\ \sigma_b &\sim \text{HalfCauchy}(2.5). \end{aligned}$$



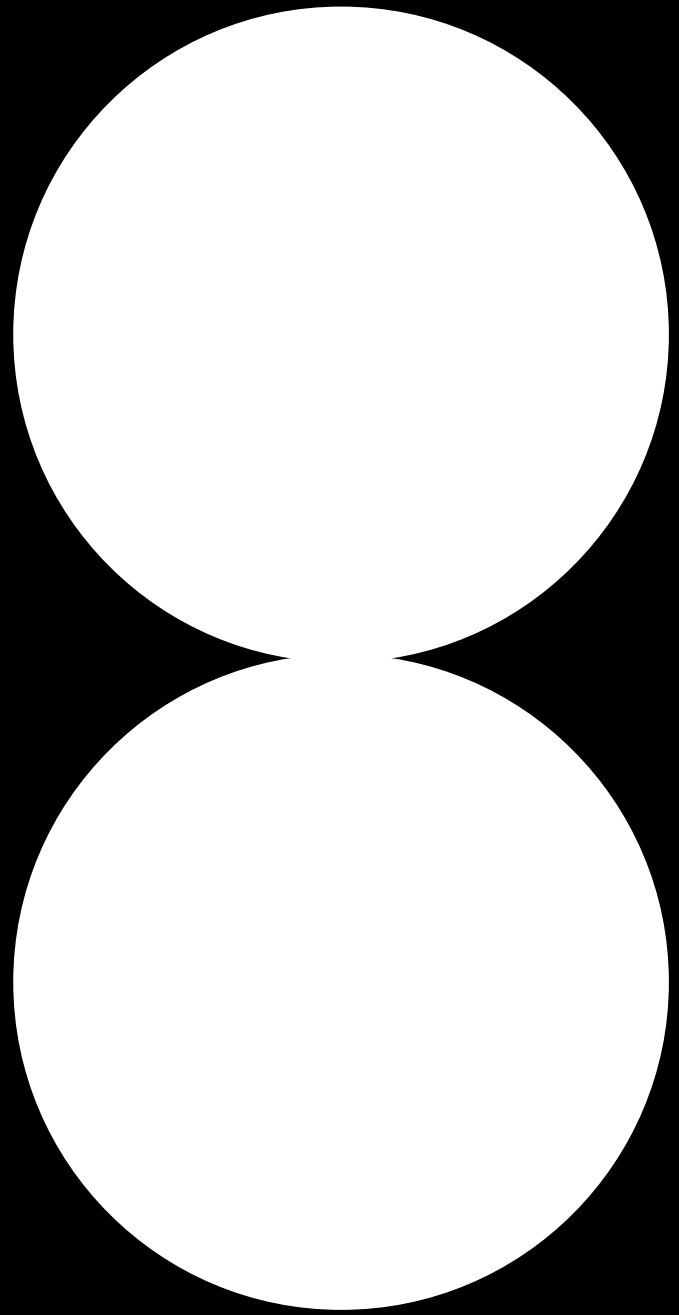


Mixture models

Notebook 3

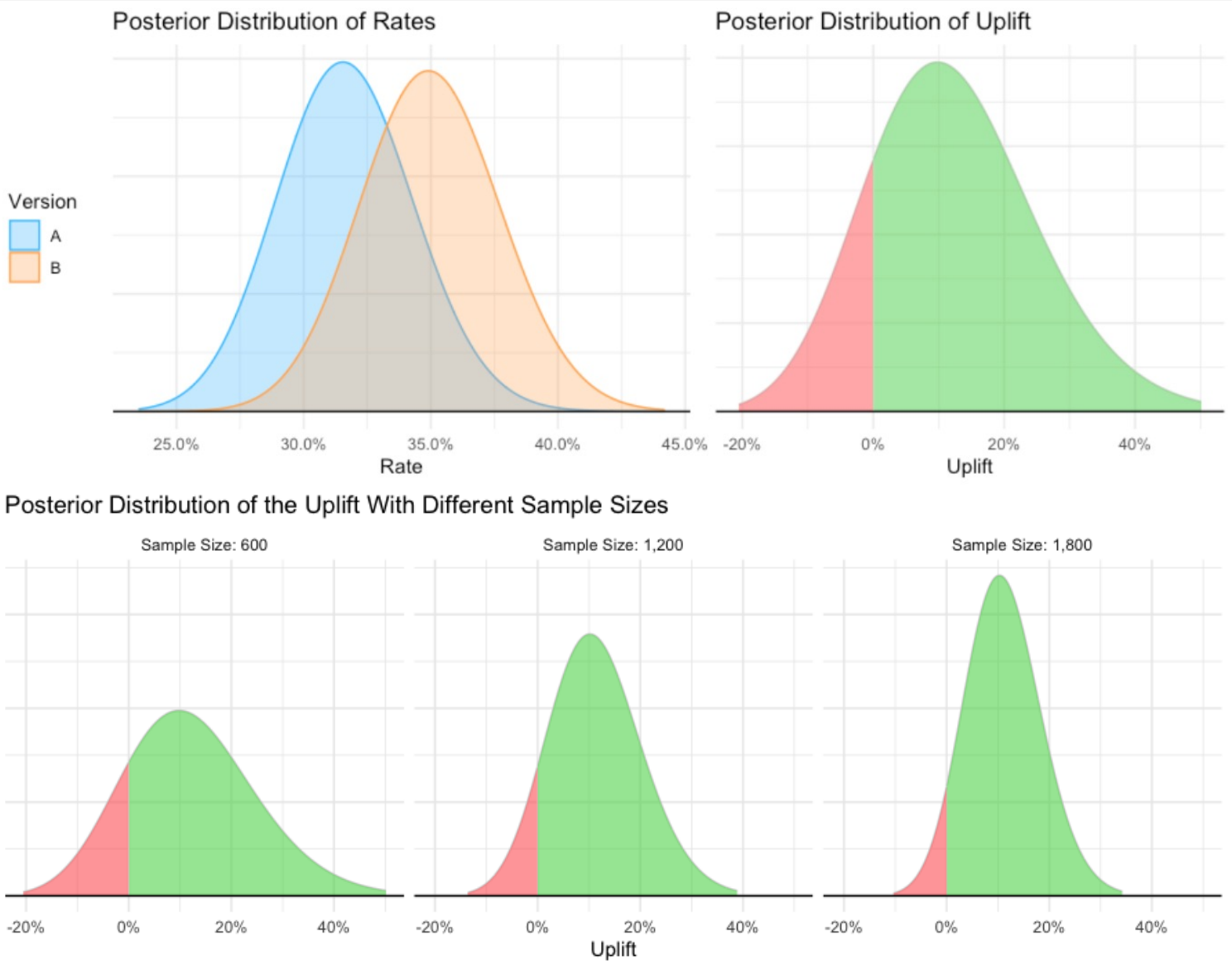
Unit8™

Industry applications

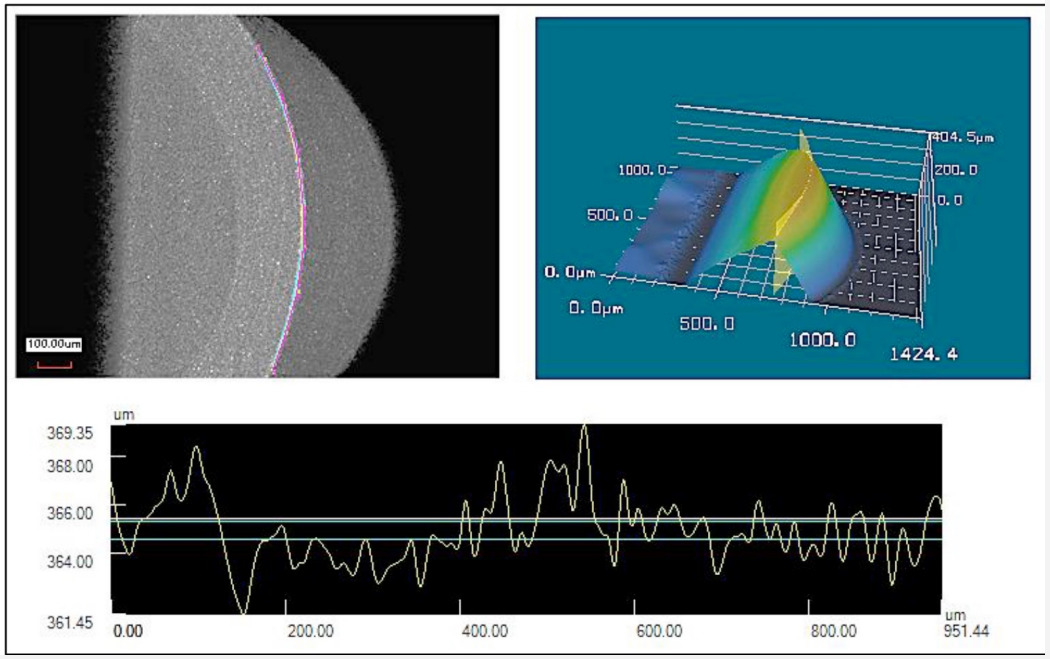


Bayesian A/B testing

	Hypothesis Testing	Bayesian A/B Testing
Knowledge of Baseline Performance	Required	Not Required
Intuitiveness	Less, as p-value is a convoluted term	More, as we directly calculate the probability of A being better than B
Sample size	Pre-defined	No need to pre-define
Peeking at the data while the test runs	Not allowed	Allowed (with caution)
Quick to make decisions	Less, as it has more restrictive assumptions on distributions	More, as it has less restrictive assumptions
Representing uncertainty	Confidence Interval (again, a convoluted interpretation which is often misunderstood)	Highest Posterior Density Region – highly intuitive interpretation
Declaring a winner	When sample size is reached and p-value is below a certain threshold	When either “probability to be best” goes above a threshold or the expected loss is below a threshold (in which case a “tie” can be declared between multiple variations)



Parameters in physical models



(3.13) and the information of the literatures. Consequently, the prior mean and one standard deviation of the parameters are given as follows:

1. $\phi_c = 17 \pm 4\ deg$
2. $\beta_a = 30 \pm 5\ deg$
3. $\tau_s = 550 \pm 80\ MPa$

Bayesian Updating for Sequential Cutting Force Prediction in Orthogonal Turning Process

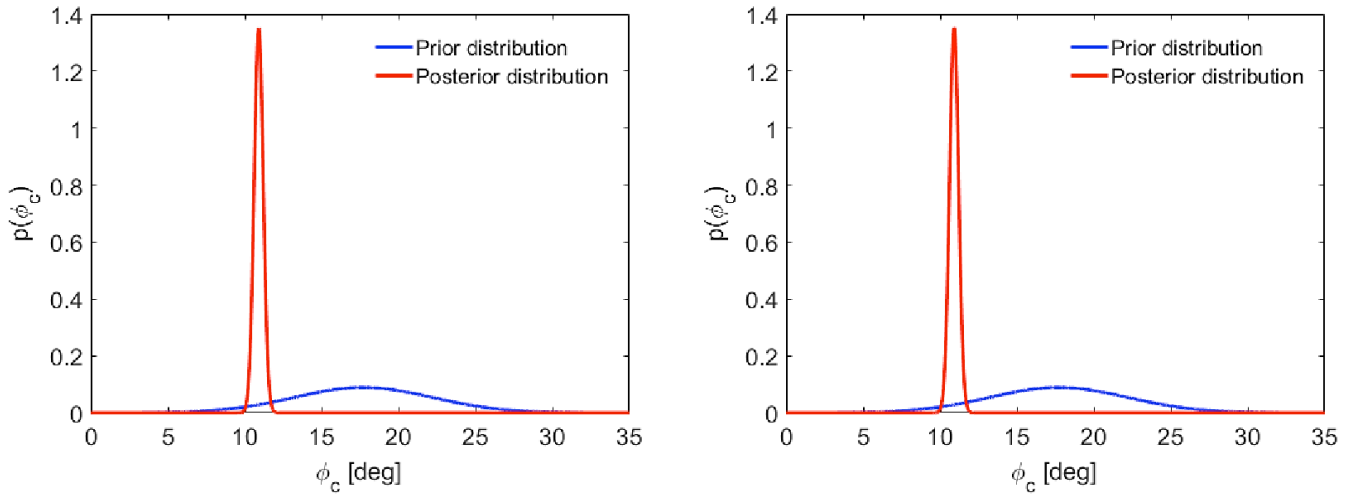


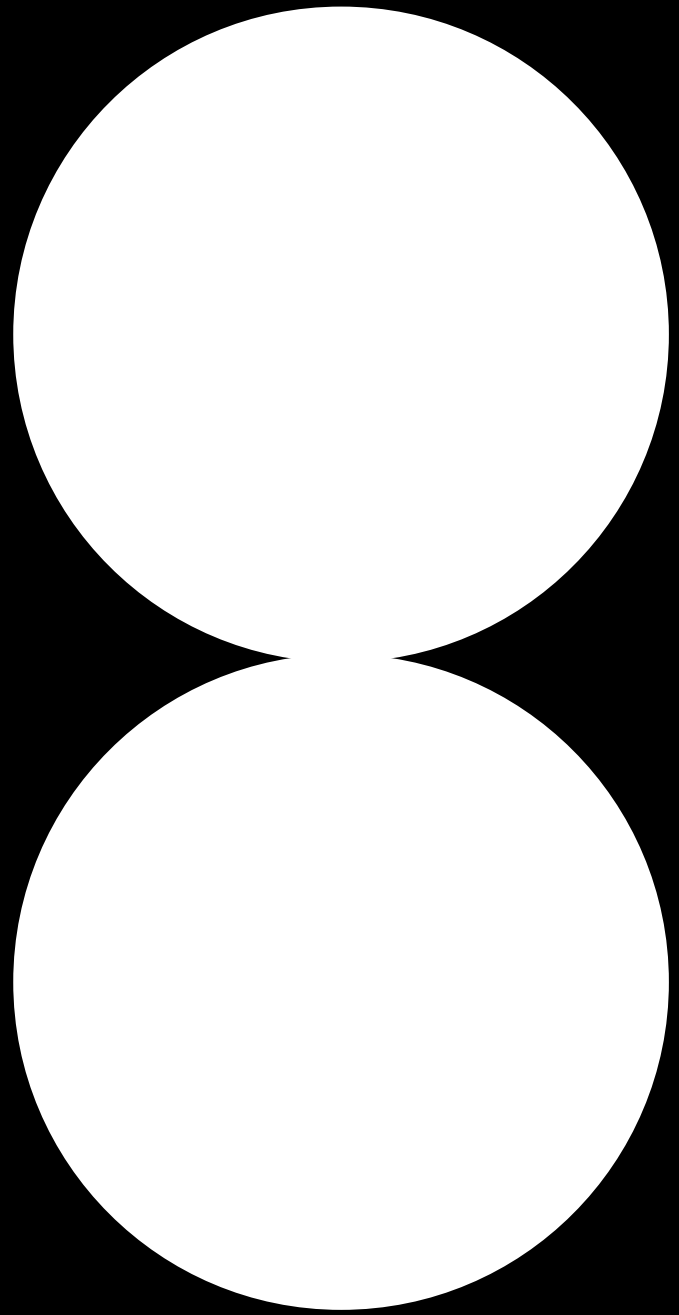
Figure 3.14: Comparison of prior and posterior distributions of ϕ_c after three updates, in tangential (left) and feed (right) directions

Obstacles to adoption

- Bayesian statistics unintuitive at first- especially priors
- Describes the variability of a parameter, not the variability of an outcome for a given hypothesis
- Fewer cookie cutter tools
- Industry inertia

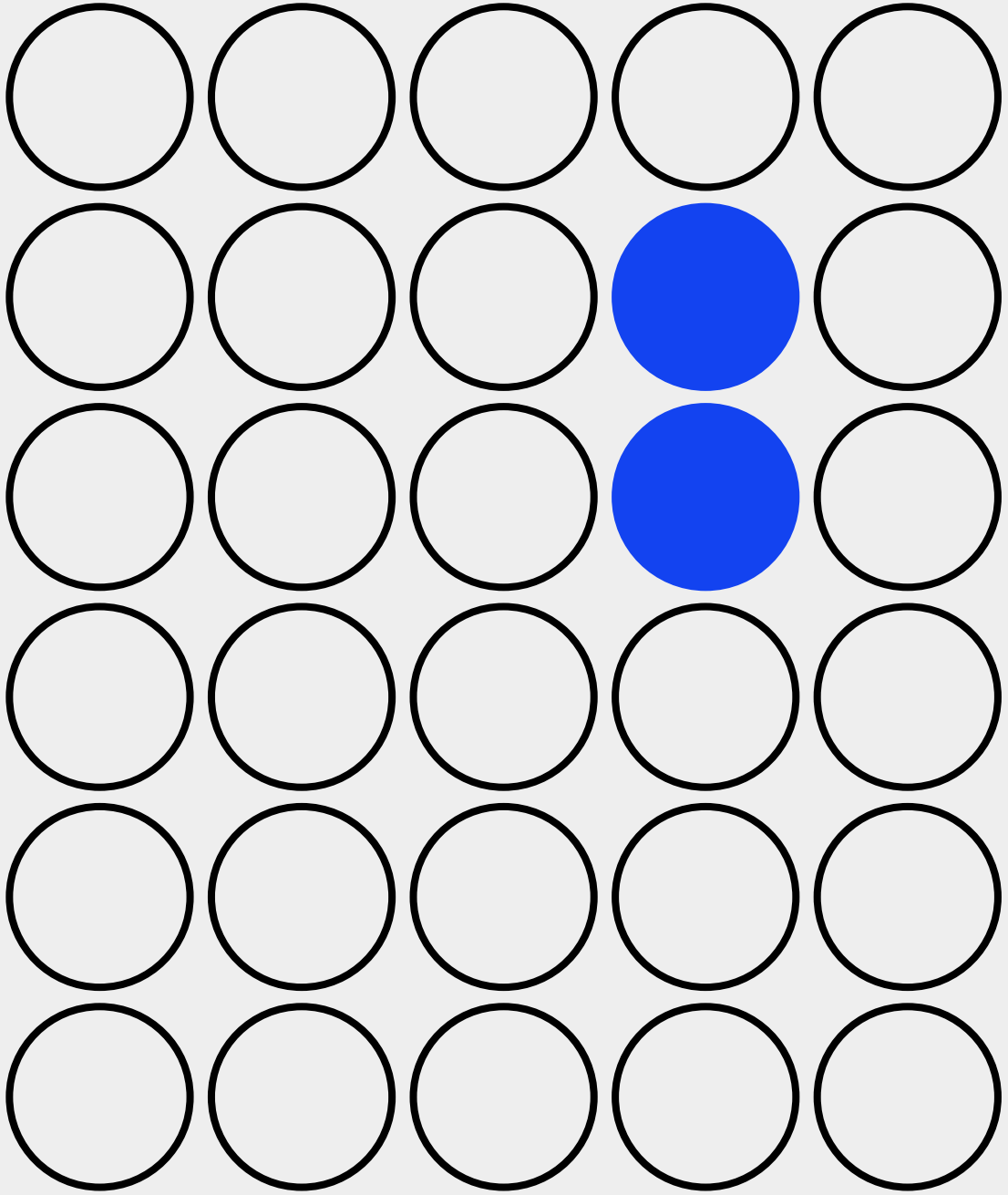
Unit8™

Questions



Unit8

unit8.com



Now Hiring!

Graduate and full time roles available at

https://unit8.com/career/

thank you