Lecture #16: Hierarchical Models and MCMC

aka STAT109A, AC209A, CSCIE-109A

CS109A Introduction to Data Science

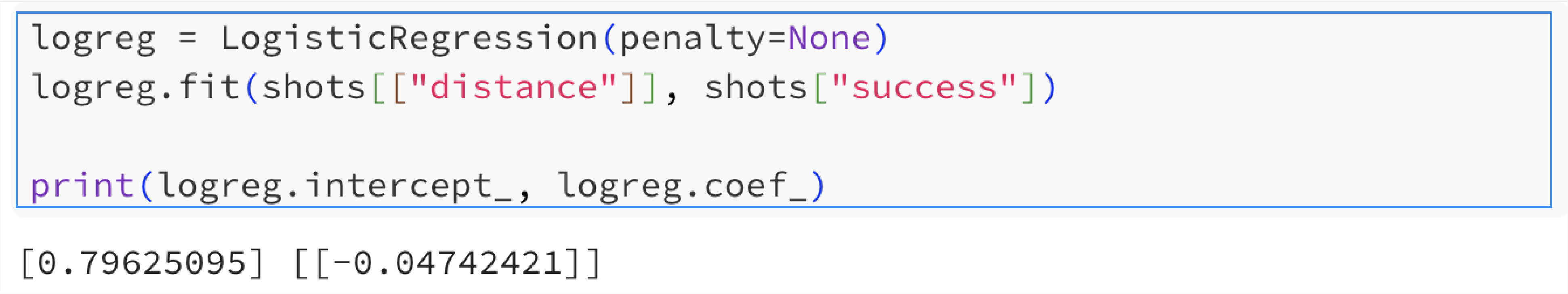
Pavlos Protopapas, Kevin Rader and Chris Gumb

Outline

- Review: logistics and logs
- Beta-Binomial Model
- Hierarchical Modeling
- Posterior Predictive Value
- Catching our Breathe
- Rejection Sampling
- MCMC
- Metropolis-Hastings

Review: Interpreting a logistic regression model

We fit a logistic regression model to predict whether a FG shot in the NBA is successful based on the distance (in feet) from the hoop that the shot was taken:



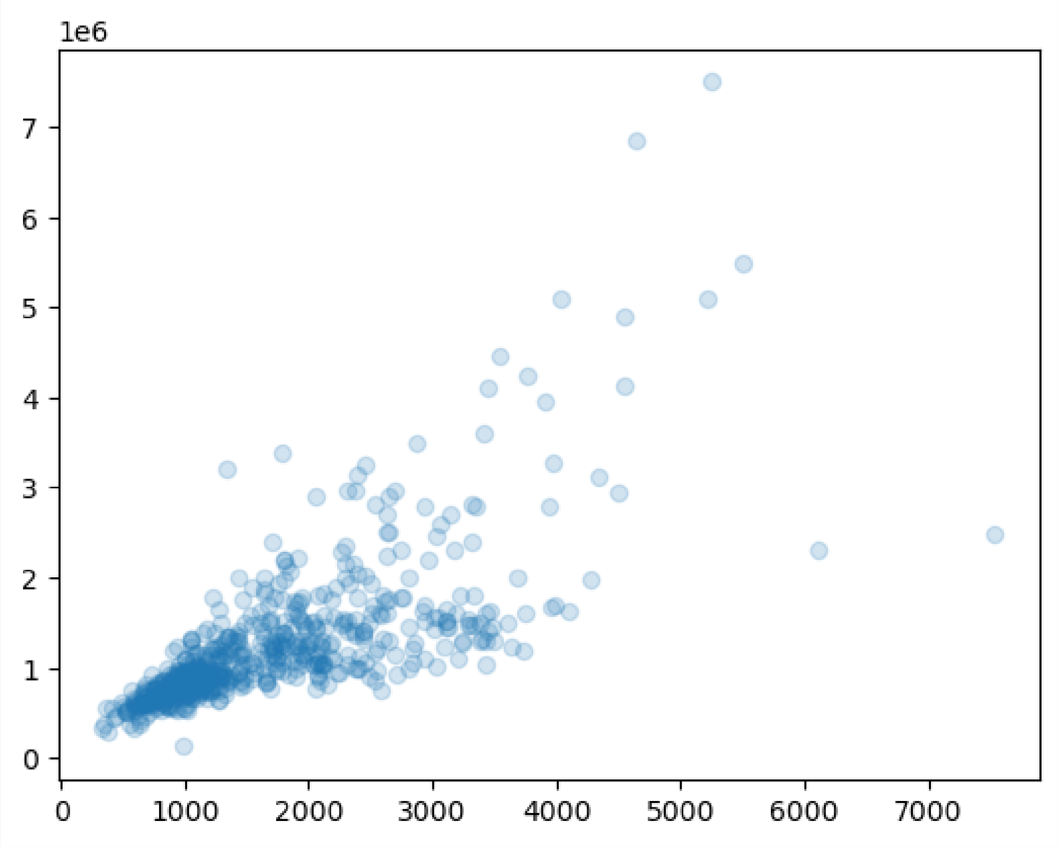
```
logreg = LogisticRegression(penalty=None)
logreg.fit(shots[["distance"]], shots["success"])

print(logreg.intercept_, logreg.coef_)
```

```
[0.79625095] [[-0.04742421]]
```

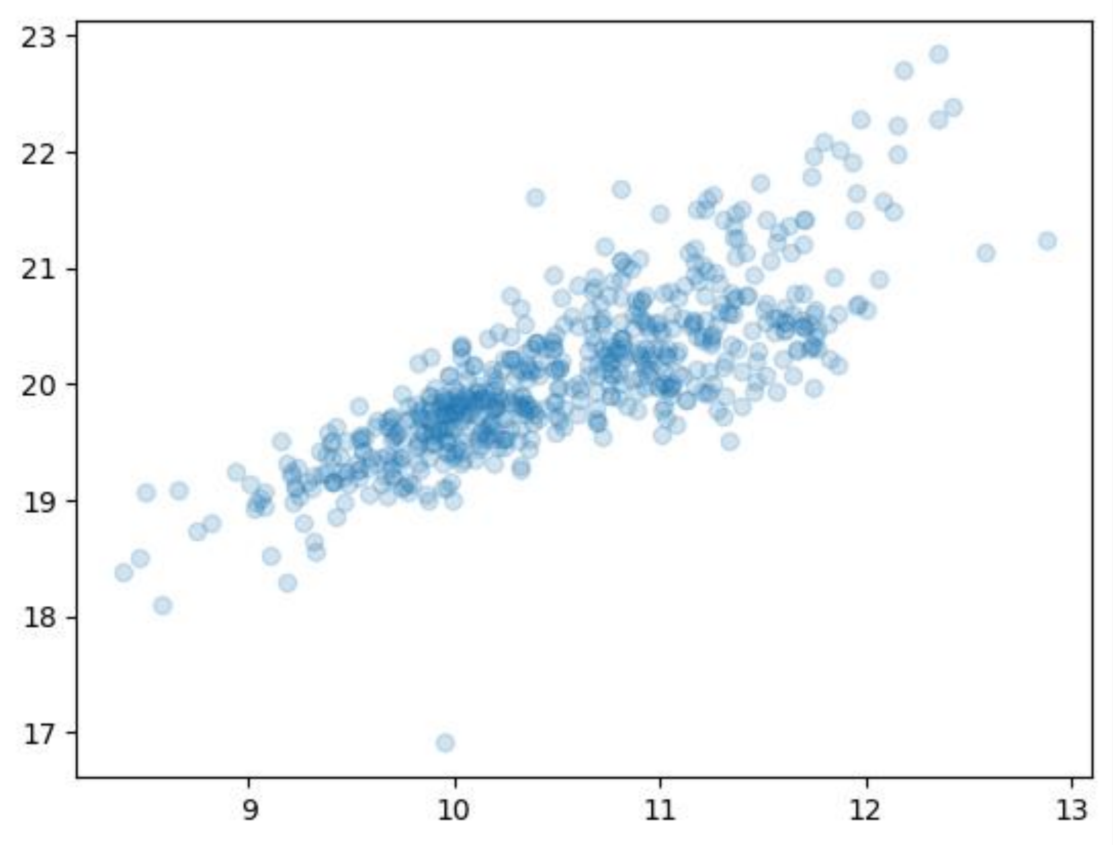
Interpret the results of this model.

Review: Interpreting a log-linear model

price vs. sqft

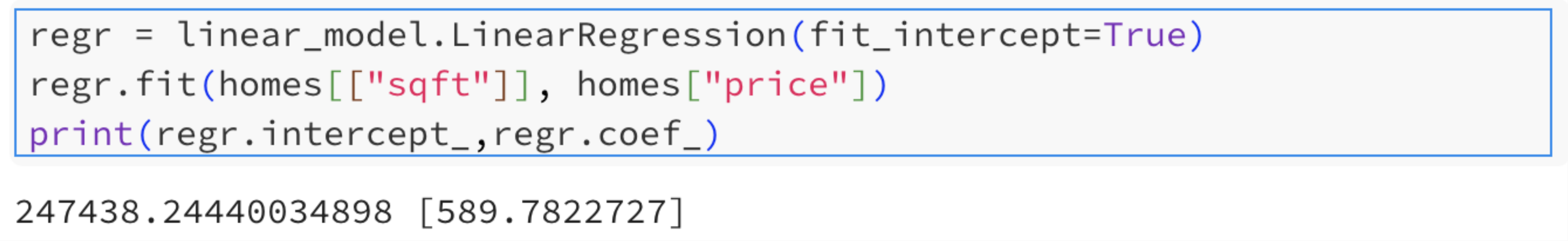


log2(price) vs. log2(sqft)



Review: Interpreting a log-linear model

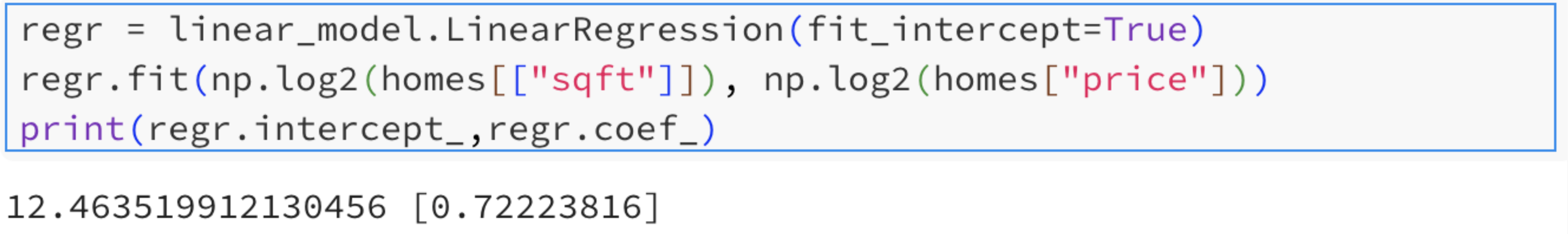
price vs. sqft



```
regr = linear_model.LinearRegression(fit_intercept=True)
regr.fit(homes[["sqft"]], homes["price"])
print(regr.intercept_,regr.coef_)
```

```
247438.24440034898 [589.7822727]
```

log2(price) vs. log2(sqft)



```
regr = linear_model.LinearRegression(fit_intercept=True)
regr.fit(np.log2(homes[["sqft"]]), np.log2(homes["price"]))
print(regr.intercept_,regr.coef_)
```

```
12.463519912130456 [0.72223816]
```

Interpret the results of these model.

Outline

- Review: logistics and logs
- **Beta-Binomial Model**
- Hierarchical Modeling
- Posterior Predictive Value
- Catching our Breathe
- Rejection Sampling
- MCMC
- Metropolis-Hastings

Bayesian Beta-Binomial Model

- Let $X_1, X_2, \ldots, X_n \sim \mathrm{Bern}(p)$.
- Let's put a prior on $p \sim Beta(a_0, b_0)$.
- What are the parameter(s) and the hyperparameters?
- Write down the prior:

$$f(p|a_0, b_0) = \left(\frac{\Gamma(a_o)\Gamma(b_0)}{\Gamma(a_0 + b_0 + 1)} p^{a_0 - 1}(1-p)^{b_0 - 1}\right)$$

- Write down the likelihood:

$$f(X_1, \ldots, X_n|p) = \prod_{i=1}^{n}(p^{x_i}(1-p)^{1-x_i}) = p^{\Sigma x_i}(1-p)^{n-\Sigma x_i}$$

- Let's ignore the normalizing constant and look at the *functional form* of the posterior (what it is proportional to).

Bayesian Beta-Binomial Model

- Thus the posterior distribution is proportional to:

$$
\begin{aligned}
f(p|X) &\propto f(X|p)\cdot f(p) \\
&= \left(\frac{\Gamma(a_o)\Gamma(b_0)}{\Gamma(a_0+b_0+1)}p^{a_0-1}(1-p)^{b_0-1}\right)\cdot\left(p^{\Sigma x_i}(1-p)^{n-\Sigma x_i}\right) \\
&\propto p^{(a_0+\Sigma x_i)-1}(1-p)^{(b_0+n-\Sigma x_i)-1}
\end{aligned}
$$

- Since the posterior's RV is *p*, any multiplicative constant not involving it can be "absorbed" by the normalizing constant. So that's why the gamma terms drop out, and we can just write the posterior as proportional to just the terms involving *p*.
- What distribution (of *p*) does this have the general form of (without the normalizing constant)?

Beta-Binomial Model: Posterior Result

- So the posterior distribution is:

$$(p|Y) \sim Beta(a_0 + \sum y_i, b_0 + (n - \sum y_i))$$

- So what?
- The posterior distribution for p for a Bernoulli distribution, given the data, only depends on the sample data in terms of the number of successes and failures. The posterior of p is Beta dist. (if we start with a prior that is Beta dist.).
- What is the posterior mean estimator (the mean of this distribution)?

$$\hat{p}_{PM} = \frac{a_0 + \sum y_i}{a_0 + b_0 + n}$$

- The posterior mean of p is an a way the weighted average of the prior's mean and the sample proportion $\hat{p}$. So what happens to the effect of the prior on the posterior (and the estimator) as n increases?

Bayesian Logistic

- How do we extend this Beta-Binomial model to Logistic Regression?
- What are the unknown parameters in this model?

- What is the likelihood for this model? How do they link to the response?

- Recall the Logistic Regression likelihood:

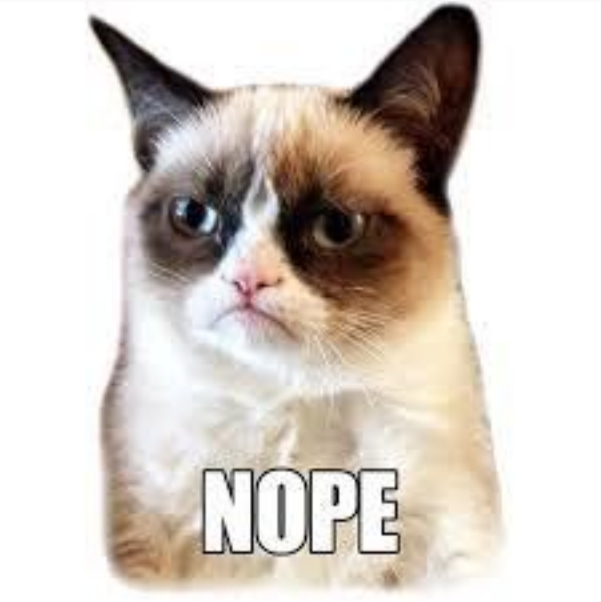
$$\left(Y_i|\vec{\beta},\vec{X}_i\right) \sim Bern(p_i)$$

$$p_i = \frac{e^{\beta_0+\beta_1 x_{1,i}+\cdots+\beta_p x_{p,i}}}{1+e^{\beta_0+\beta_1 x_{1,i}+\cdots+\beta_p x_{p,i}}}$$

- Recall: we often write this as $\ln\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_{1,i} + \cdots + \beta_p x_{p,i}$ for interpretive reasons.

Beta-Binomial Model: Logistic Regression Result

- Can we simply put Beta priors on our unknown parameters?



- Conventionally, we assume $\beta \sim N(\mu_0 = 0, \sigma^2)$ priors. This shrinks the coefficients towards zero (think Ridge).
- We'll come back to this later today when we learn how to **sample** from posterior distributions (think MCMC)!

Outline

- Review: logistics and logs
- Beta-Binomial Model
- **Hierarchical Modeling**
- Posterior Predictive Value
- Catching our Breathe
- Rejection Sampling
- MCMC
- Metropolis-Hastings

Hierarchical Modeling: the idea

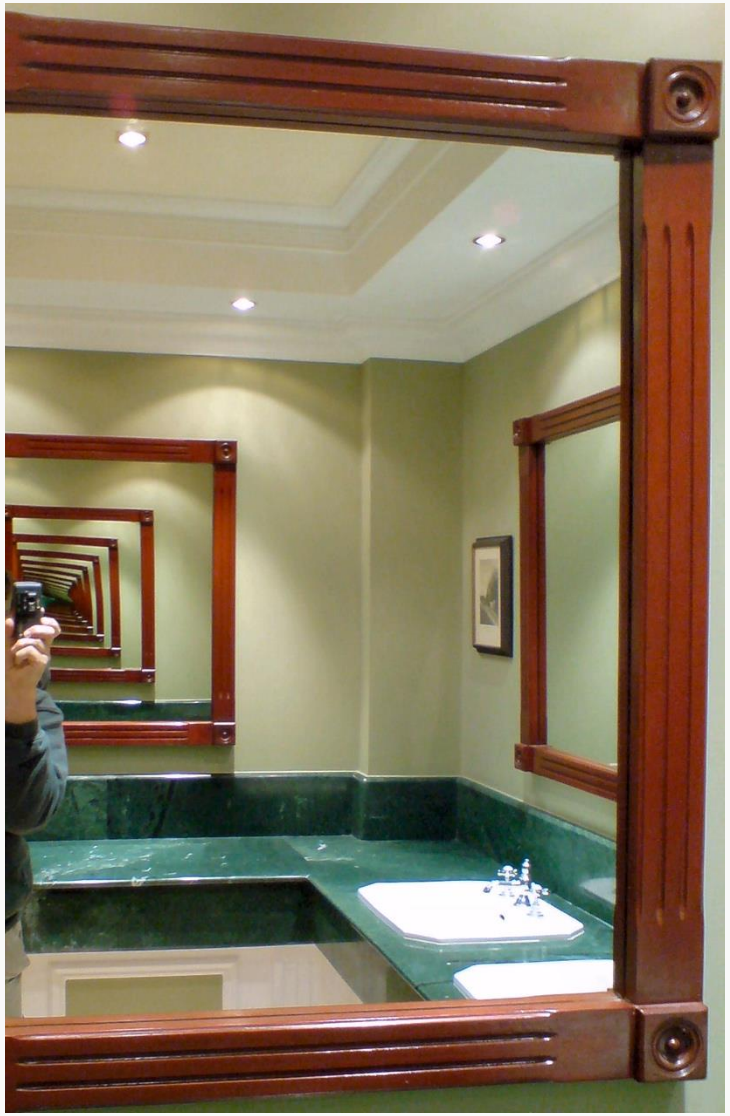
- Recall the Bayesian Beta-Binomial model:

$$(Y_i | p_i) \sim Bern(p_i)$$

$$p_i \sim Beta(a_0, b_0)$$

- What do a_0 and b_0 represent? What do we call them?
- How can we complicate things even more?
  - Let's put priors on the hyperparameters!
- This is called a hyperprior distribution.
  - What would be reasonable hyperprior distributions?
    - Hint: think support.
- What's stopping us from putting priors on our hyperprior's parameters?
  - Shall we call this a hyper-hyperprior distribution?

Hierarchical Modeling: the Bayesian perspective



Hierarchical Modeling: examples

- So why would we bother with putting hyperpriors on our priors?
- 2 main ideas:
  1. We may be uncertain about which hyperparameter(s) to use in our prior.
  2. More commonly: the data's structure. What if we measure data at several *levels*?
- Examples:
  - **Governments:** data could be measured on individuals, that reside within counties (that have their own measurements), which reside within states, that reside withing regions, etc.
  - **Education:** students within schools within districts within states.
  - **Medicine:** patients from doctors within hospitals
  - **Biology:** cells from tissues from organs from individuals
  - And in sports (one last time)…

Hierarchical Modeling: an example

- Who are the best [offensive] players in the NBA?

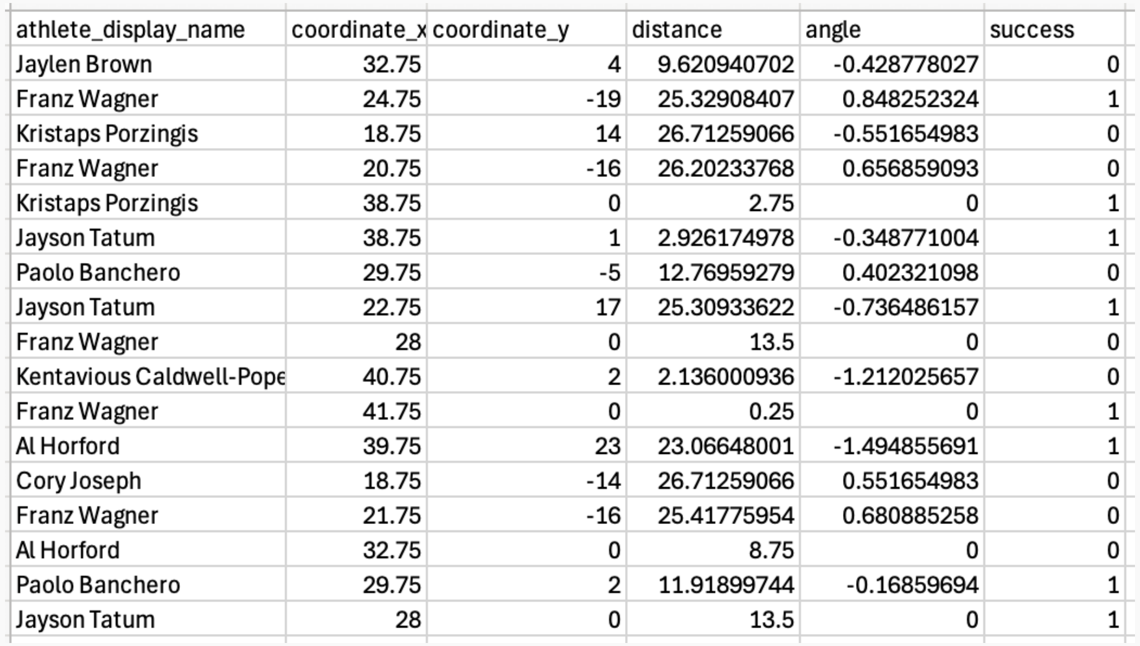
- We’d like to model the chances of a field goal attempt being a success given the location (distance and possibly angle) and the player who is taking the shot.
- What will the data look like? Why is a hierarchical model a reasonable one?

Hierarchical Modeling: an e

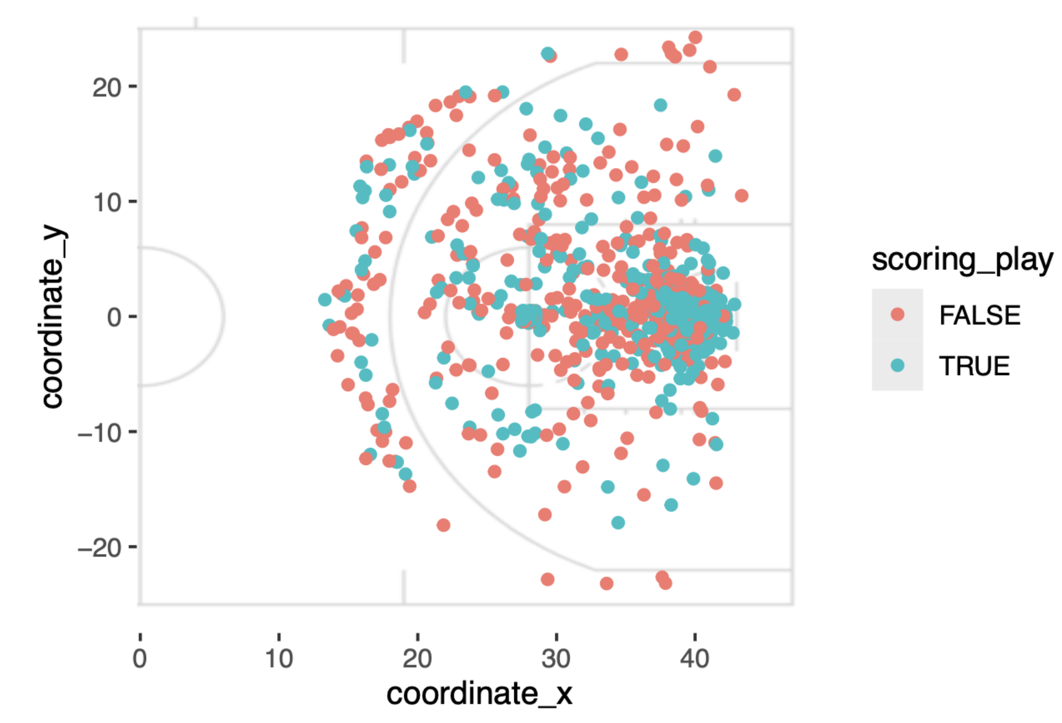
- Who are the best [offens e NBA?

- We’d like to model the ch goal attempt being a success given the locatio d possibly angle) and the player who is takin
- What will t d a reaso

Basketball Shot data



athlete_display_name	coordinate_x	coordinate_y	distance	angle	success
Jaylen Brown	32.75	4	9.620940702	-0.428778027	0
Franz Wagner	24.75	-19	25.32908407	0.848252324	1
Kristaps Porzingis	18.75	14	26.71259066	-0.551654983	0
Franz Wagner	20.75	-16	26.20233768	0.656859093	0
Kristaps Porzingis	38.75	0	2.75	0	1
Jayson Tatum	38.75	1	2.926174978	-0.348771004	1
Paolo Banchero	29.75	-5	12.76959279	0.402321098	0
Jayson Tatum	22.75	17	25.30933622	-0.736486157	1
Franz Wagner	28	0	13.5	0	0
Kentavious Caldwell-Pope	40.75	2	2.136000936	-1.212025657	0
Franz Wagner	41.75	0	0.25	0	1
Al Horford	39.75	23	23.06648001	-1.494855691	1
Cory Joseph	18.75	-14	26.71259066	0.551654983	0
Franz Wagner	21.75	-16	25.41775954	0.680885258	0
Al Horford	32.75	0	8.75	0	0
Paolo Banchero	29.75	2	11.91899744	-0.16859694	1
Jayson Tatum	28	0	13.5	0	1



Predicting shot success

- Let Y_{ij} be an indicator variable for whether the i^{th} shot from the j^{th}player is a success.
- We are going to predict this response based on distance, X_1, and the player taking the shot.
- What would be the standard parametric approach to modeling this situation?
- A logistic regression model of course! With Llotsof predictors:

$$Y_{ij} \sim \text{Bern}\left(\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \beta_0 + \beta_1 X_{1,ij} + \beta_2 \mathbf{1}(j = \text{“}Lebron\text{”}) + \cdots\right)$$

- What are some issues with taking this approach?

Hierarchical Modeling these *clustered* data

- Let Y_{ij} be an indicator variable for whether the i^{th} shot from the j^{th}player is a success.
- We are going to predict this response based on distance, X_1, and the player taking the shot.
- This lends itself naturally to a hierarchical logistic regression model:

$$\left(Y_{ij}|\alpha_j, \beta_1, X_{1,ij}\right) \sim \text{Bern}\left(\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \alpha_j + \beta_1 X_{1,ij}\right)$$

$$\alpha_j \sim N(\alpha_0, \sigma_\alpha^2)$$

- What are the interpretations of these parameters?
- Wait, how is this a Bayesian model?

Hierarchical Modeling: what's the posterior?

$$\alpha_j, \beta_1, \sigma_\alpha^2 \propto \left(Y_{ij} | \alpha_j, \beta_1, X_{1,ij}\right) \sim \text{Bern}\left(\ln\left(\frac{p_{ij}}{1 - p_{ij}}\right) = \alpha_j + \beta_1 X_{1,ij}\right)$$

$$\alpha_j \sim N(\alpha_0, \sigma_\alpha^2)$$

- What would be the functional form of the posterior distribution?
- It's not pretty: no reason to really write it down.
- So how can we estimate the posterior distributions of the parameters we care about?
  - We will have to sample from it!
- Note: we put an **improper prior** on $\alpha_0, \beta_1, \sigma_\alpha^2$ (do not explicitly state what it is). We can call this an **empirical Bayesian** approach (which frequentists can jive with as well!).

Hierarchical Modeling: shooting results

As an OLS (player coefs):

Alondes Williams	10.232546
Jahmir Young	2.467918
Kai Jones	1.616242
N'Faly Dante	1.582438
Daeqwon Plowden	1.547737

Terry Taylor	-9.254500
Zyon Pullin	-9.141658
Isaiah Stevens	-8.755412
Mac McClung	-8.719360
Riley Minix	-8.110082

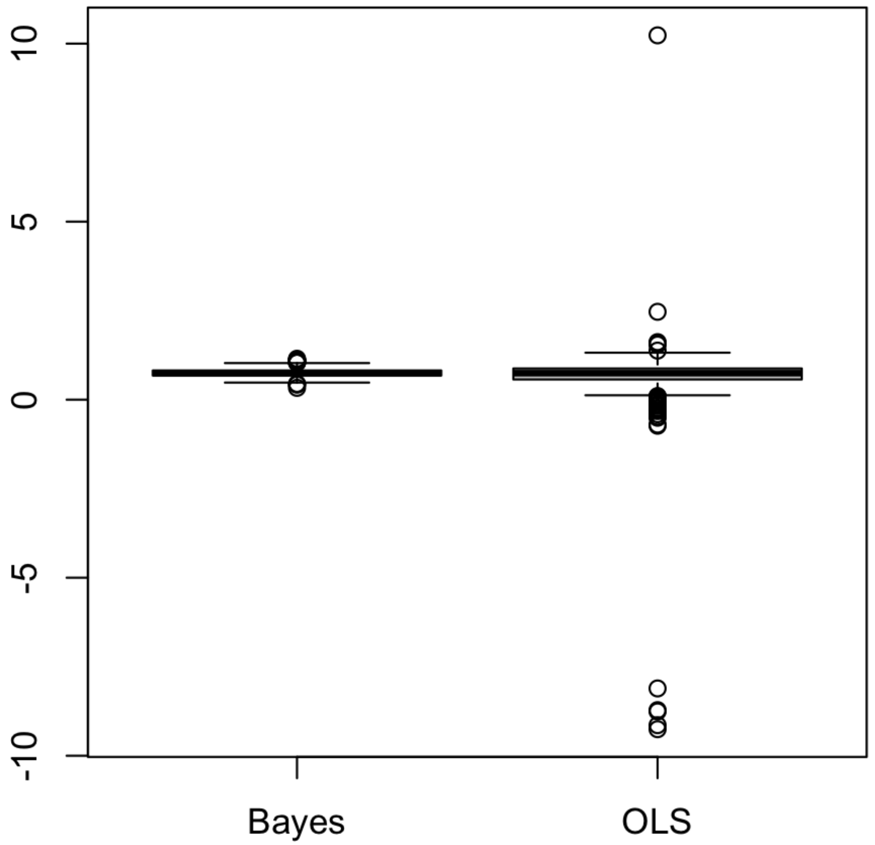
As a Bayes' Hierarchical Model:

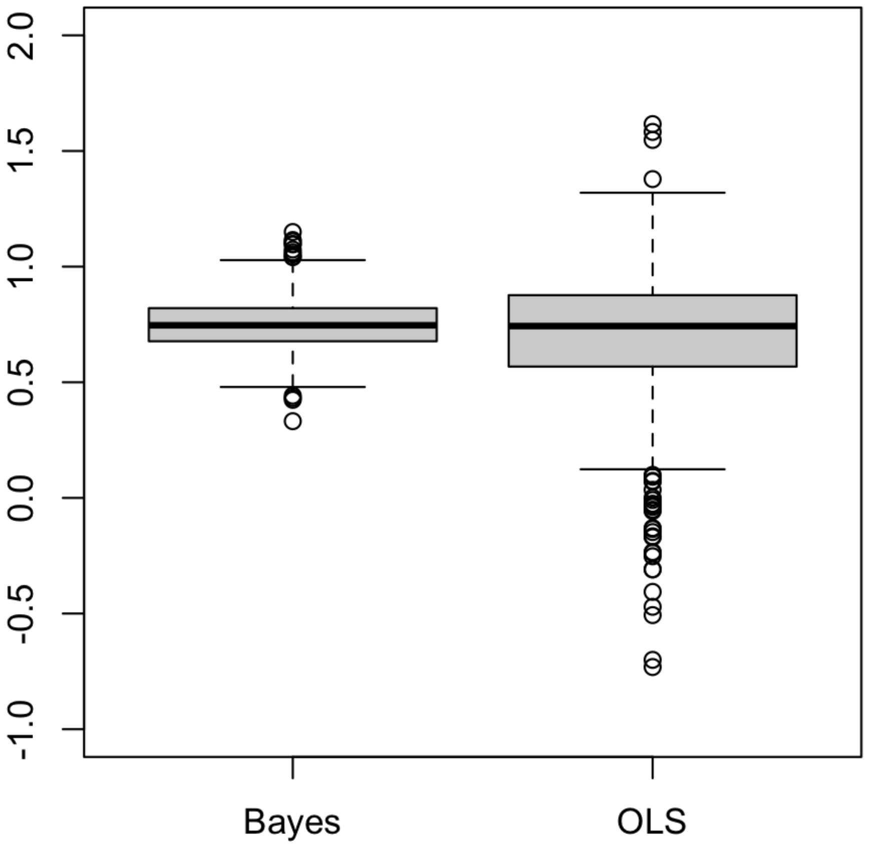
Jarrett Allen	1.149615
Shai Gilgeous-Alexander	1.115016
Damian Lillard	1.109216
Kevin Durant	1.097781
Kai Jones	1.096014

Elfrid Payton	0.3317146
Tristan Thompson	0.4240332
Cody Williams	0.4327059
Kris Murray	0.4419451
KJ Simpson	0.4795819

Mac took 2 shots and missed them both

Hierarchical Modeling: distribution of player coefficients





Hierarchical Modeling: other versions

- Our basketball shooting model was set up as:

$$(Y_{ij}|\alpha_j, \beta_1, X_{1,ij}) \sim \text{Bern}\left(\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \alpha_j + \beta_1 X_{1,ij}\right)$$
$$\alpha_j \sim N(\alpha_0, \sigma_\alpha^2)$$

- By allowing the intercept α_j to vary across players, we are saying each player has an intrinsic shooting ability that applies the same for all distances.
- But what about β_1? We could allow this to vary as well!
- Compared to the average player Some players may shoot well close to the basket (think Giannis) while other shooters may shoot better further from the basket (think Steph Curry).

Hierarchical Modeling: other versions

- Our basketball shooting model could be extended to include **random slopes**:

$$\left(Y_{ij} | \alpha_j, \beta_1, X_{1,ij}\right) \sim \mathrm{Bern}\left(\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \alpha_j + \beta_j X_{1,ij}\right)$$
$$\alpha_j \sim N(\alpha_0, \sigma_\alpha^2)$$
$$\beta_j \sim N(\beta_0, \sigma_\beta^2)$$

Notes:

1. We could model (α_j, β_j) jointly as a Multivariable Normal with correlation ρ
2. We could priors on $\alpha_0, \sigma_\alpha^2, \beta_0, \sigma_\beta^2$ and be fully Bayesian!

Outline

- Review: logistics and logs
- Beta-Binomial Model
- Hierarchical Modeling
- **Posterior Predictive Value**
- Catching our Breathe
- Rejection Sampling
- MCMC
- Metropolis-Hastings

Predicting the future

What if we want to use our Bayes model to predict future observations?

For a given trained model, this is called the **posterior predictive distribution** (of future data).

For example, we may want to predict the probability that a future shot is successful in our basketball shooting model.

What inputs do we need to know in order to do this?

Posterior Predictive Value

- What distribution will a future observation, $\tilde{Y}$, have?
    - Conditional on a known parameter, it is still based on the likelihood function: $p\left(\tilde{Y}\middle|\theta\right)$.
- However, we now need to sample it given 1 or more random variables: the unknown parameters, which follow the now estimated posterior distribution: $p(\theta|Y)$!
- What is a data scientist to do?
    - Consider (aka, Integrate across) all values of the parameter(s)
- Mathematically this means:

$$p\left(\tilde{Y}|Y\right) = \int_{\theta} p\left(\tilde{Y}\middle|\theta\right)p(\theta|Y)d\theta$$

- What do each of these pieces mean? What is this analogous to in the standard parametric model?

Outline

- Review: logistics and logs
- Beta-Binomial Model
- Hierarchical Modeling
- Posterior Predictive Value
- **Catching our Breathe**
- Rejection Sampling
- MCMC
- Metropolis-Hastings

Stepping Back for a Moment

- When deriving the posterior distribution, we can drop scaling constants not including the unknown parameter.
  - These constants make the posterior area *sum* to 1.
- By combining posterior for θ with our data generating model, we can derive a posterior predictive distribution of future data.
  - A helpful tool for forecasting!
- For our choices of likelihood and prior, the posterior has been the same type of distribution as the prior so far.
  - Recall: this is called **conjugacy**.
- There are many examples of conjugate distributions (with their associated posterior predictive distributions).
  - See the Wikipedia page on conjugate priors for a thorough list.

Stepping Back for a Moment

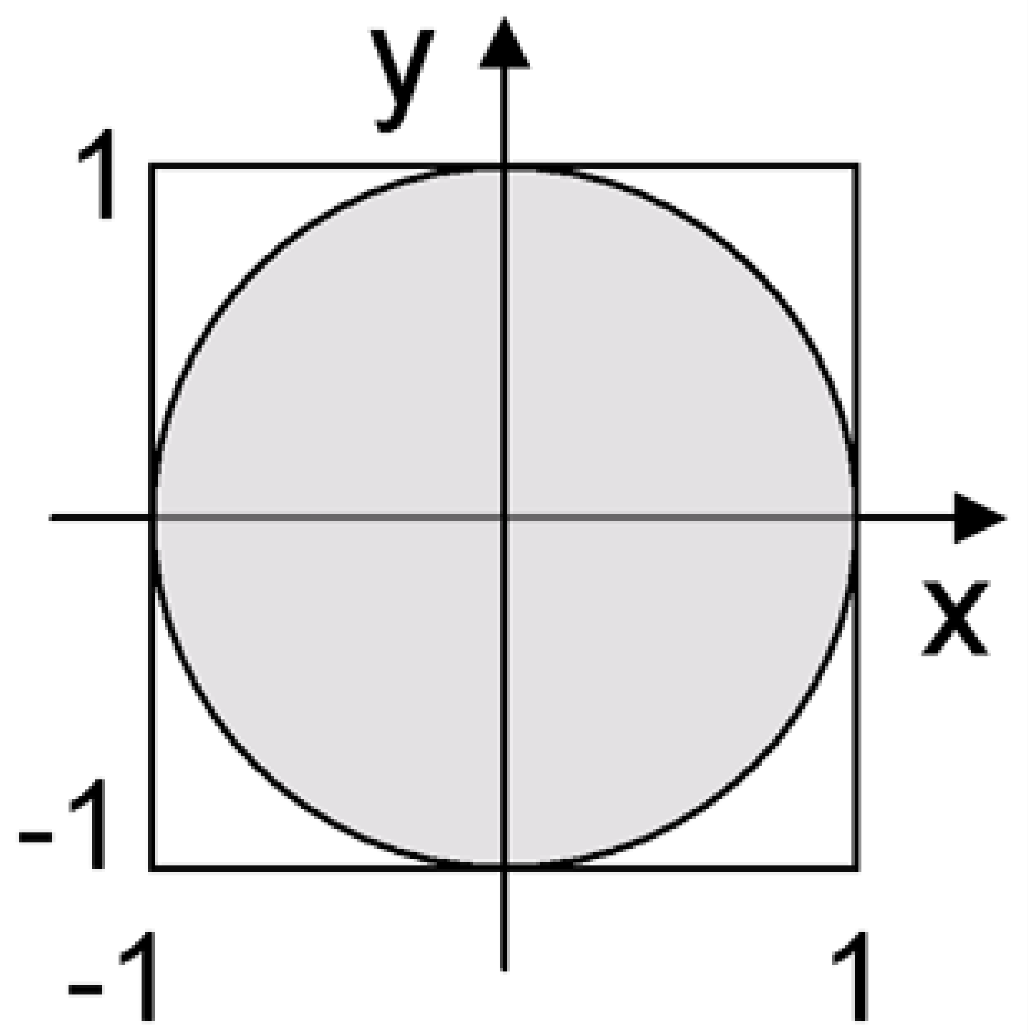
- By combining posterior for θ with our data generating model, we can derive a posterior predictive distribution of future data.
    - A helpful tool for forecasting
- The conditional structure of Bayesian models will allow us to approximate lengthy integrals with (nested) simulations (Monte Carlo to come).
    - The complexity of what we have to simulate depends on the complexity of the distributions used in our models
- Monte Carlo methods can address difficult or intractable analysis
    - Not a one-size-fit-all solution.
    - Sometimes a little/lot of pen and paper is worth years of compute.

Outline

- Review: logistics and logs
- Beta-Binomial Model
- Hierarchical Modeling
- Posterior Predictive Value
- Catching our Breathe
- **Rejection Sampling**
- MCMC
- Metropolis-Hastings

Rejection Sampling

Let's throw some darts!



Rejection Sampling

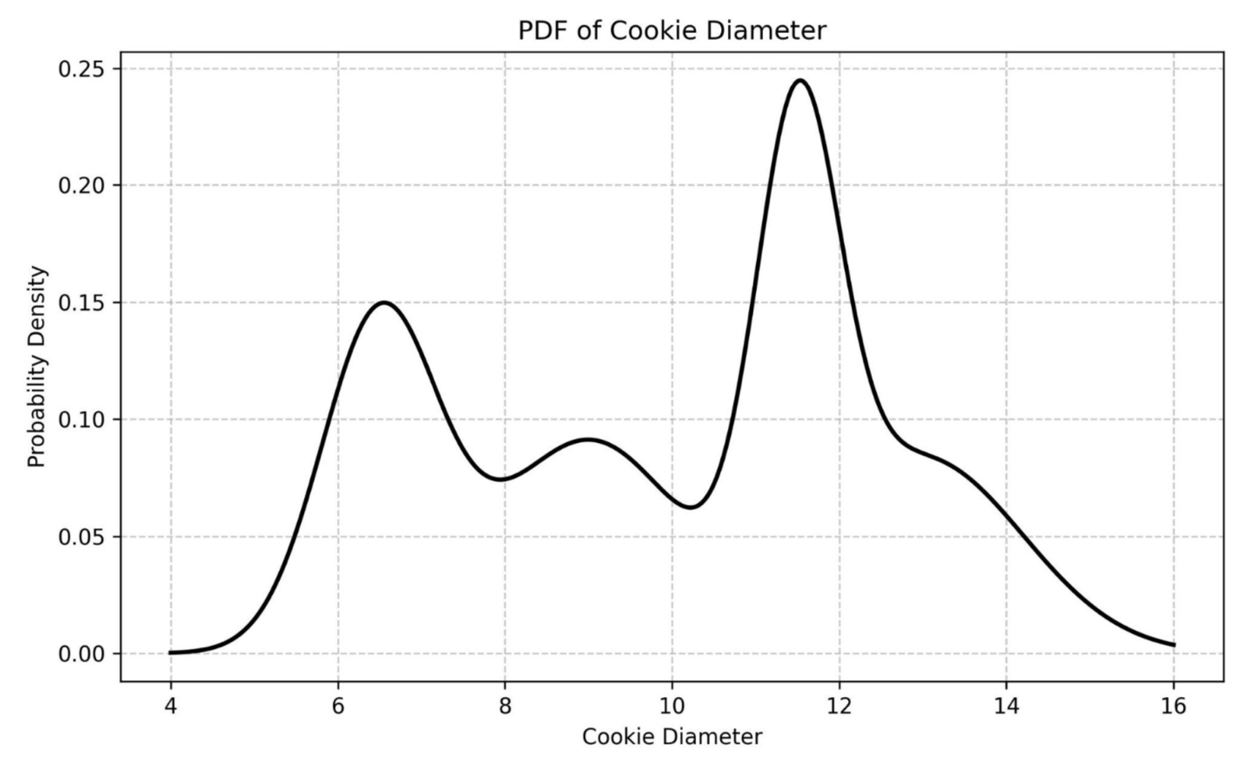
1. **Define a Target Distribution** $f(x)$: Specify the probability distribution you want to sample from.
2. **Choose a Proposal Distribution** $g(x)$: Select a simpler distribution from which you can easily sample, ensuring $f(x) \leq Mg(x)$ for all x, where $M > 1$ is a scaling factor.
3. **Draw a Candidate Sample:** Generate a random sample x from the proposal distribution $g(x)$.
4. **Generate a Uniform Random Number:** Draw $u \sim Uniform(0,1)$.
5. **Acceptance Condition:**
   - Compute the acceptance ratio $r = f(x)/Mg(x)$.
   - Accept the candidate sample x if $u \leq r$; otherwise, reject it.
6. **Repeat Until Desired Sample Size:** Repeat steps 3–5 until you collect enough accepted samples.

Cookie Monster Needs Our Help!

Cookie Monster stumbled upon a famous bakery that **sells four varieties of cookies: chocolate chip, oatmeal raisin, peanut butter, and sugar cookies.**

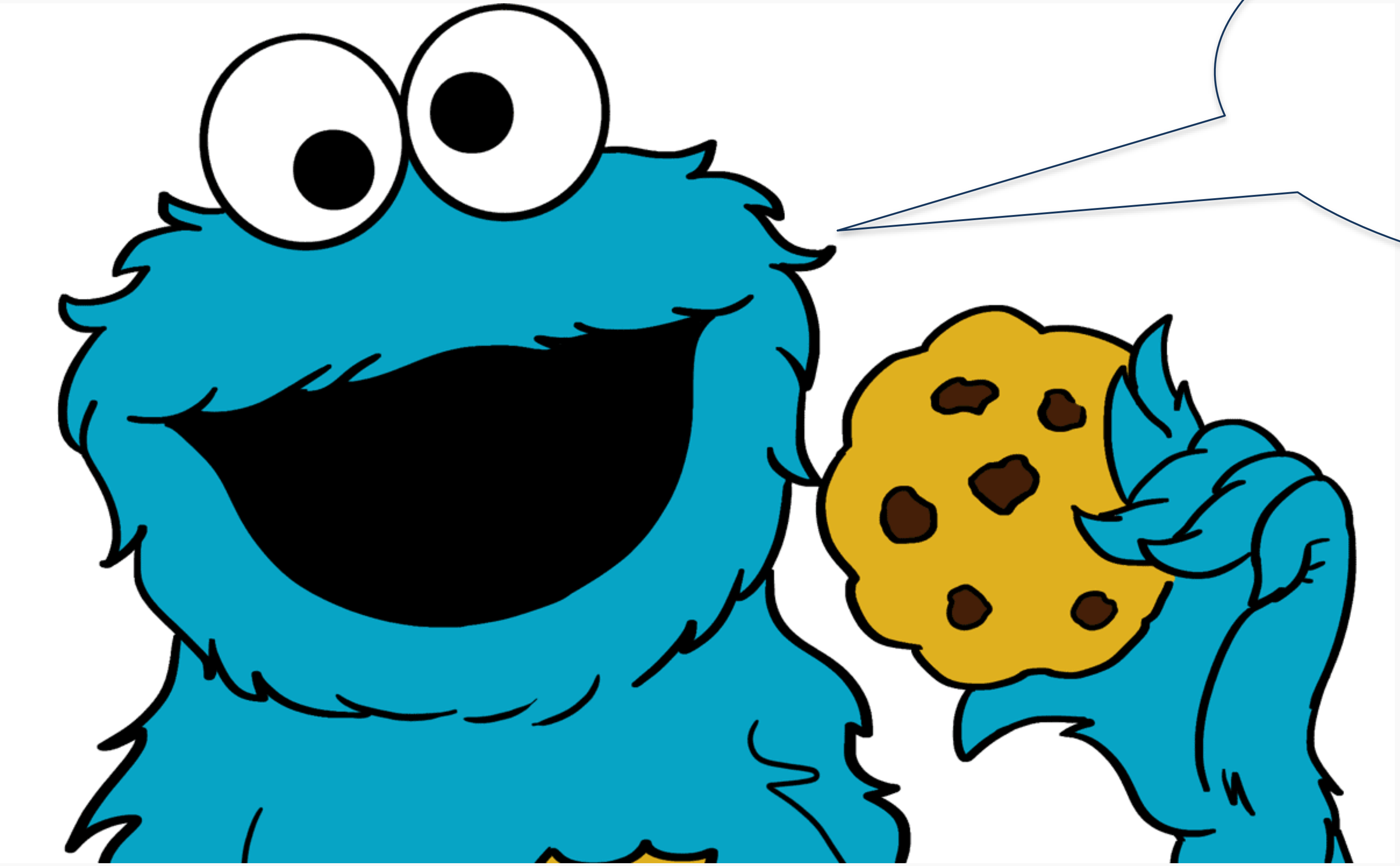
- Each variety has a diameter that varies slightly due to the baking process and the ingredients.
- The Cookie Monster is interested in sampling from the PDF of cookie diameters, but how?

Only Information Available: Cookie Diameter PDF



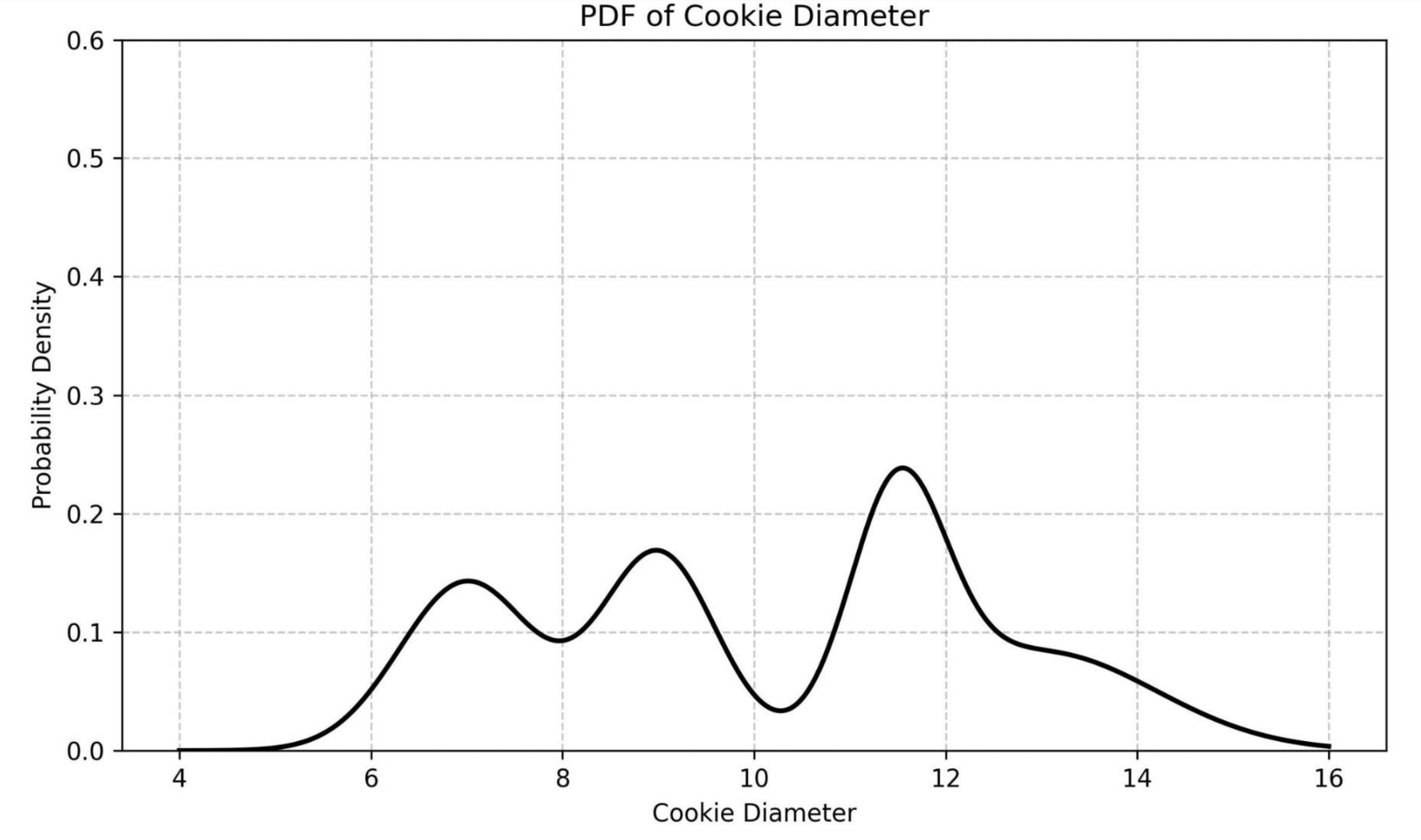
How would you describe this distribution? What is this suggestive of?

Cookie Monster is Smart!

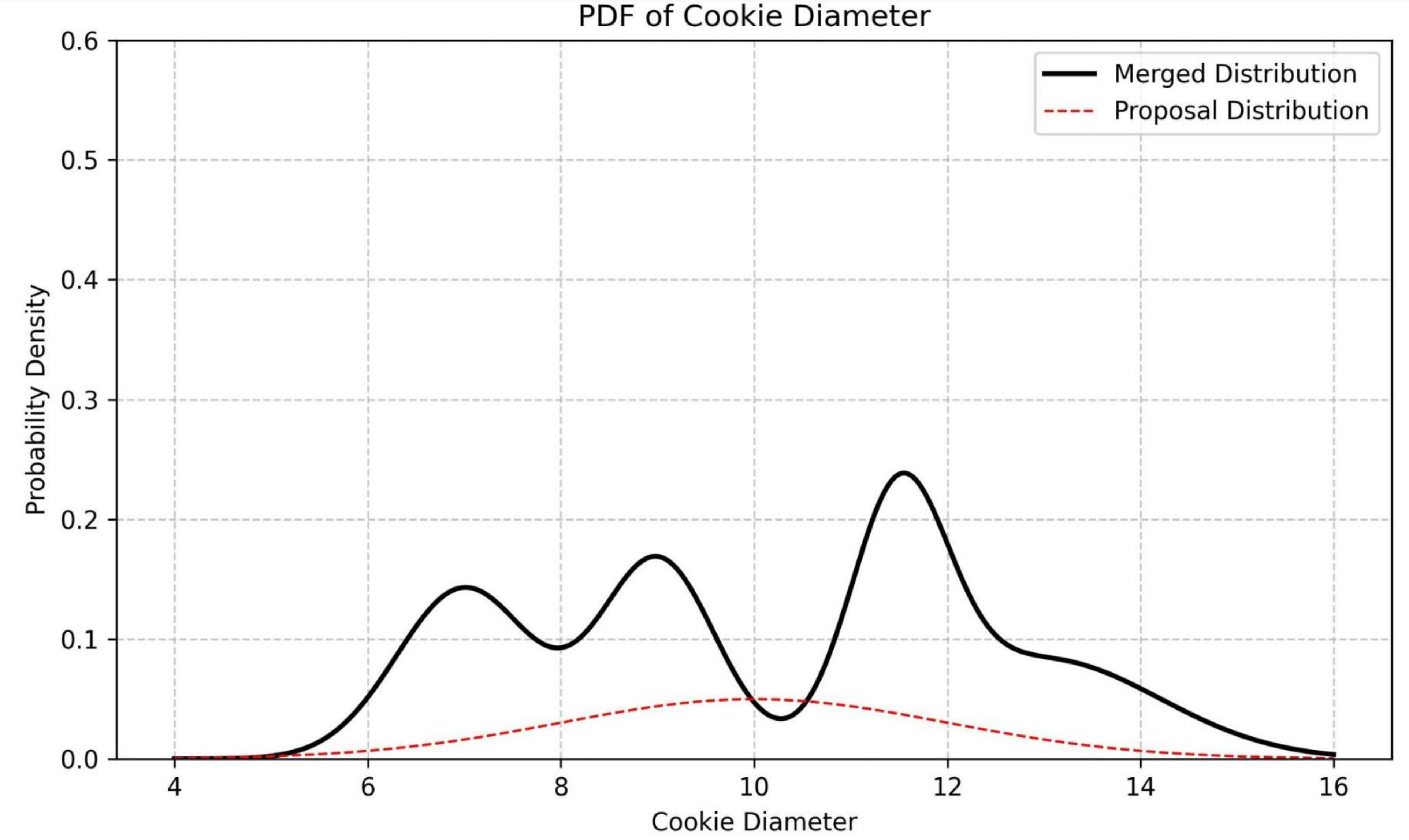


Can we use rejection sampling to generate a sample from this distribution?

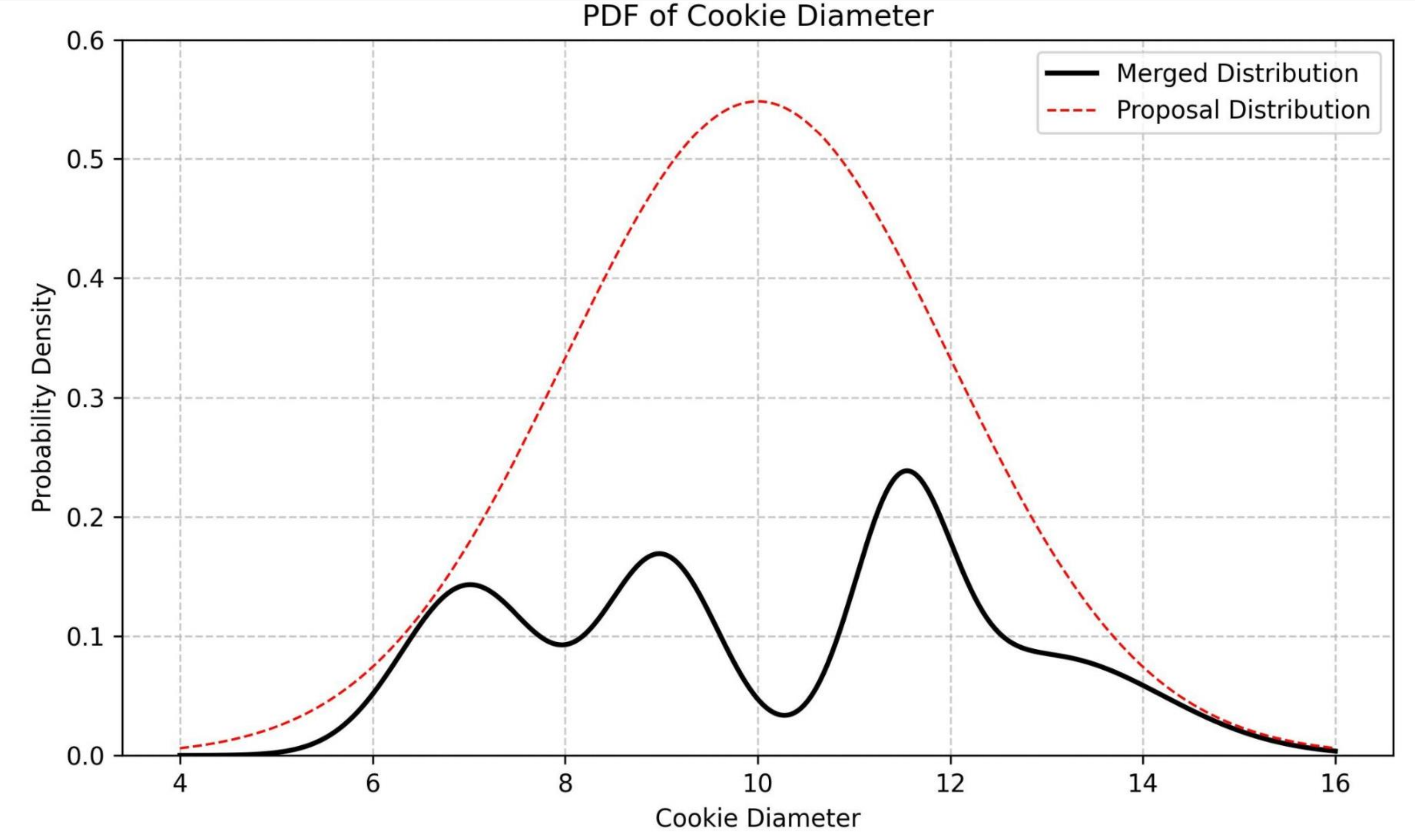
Step 1: Know your Target Distribution



Step 2A: Proposal Distribution



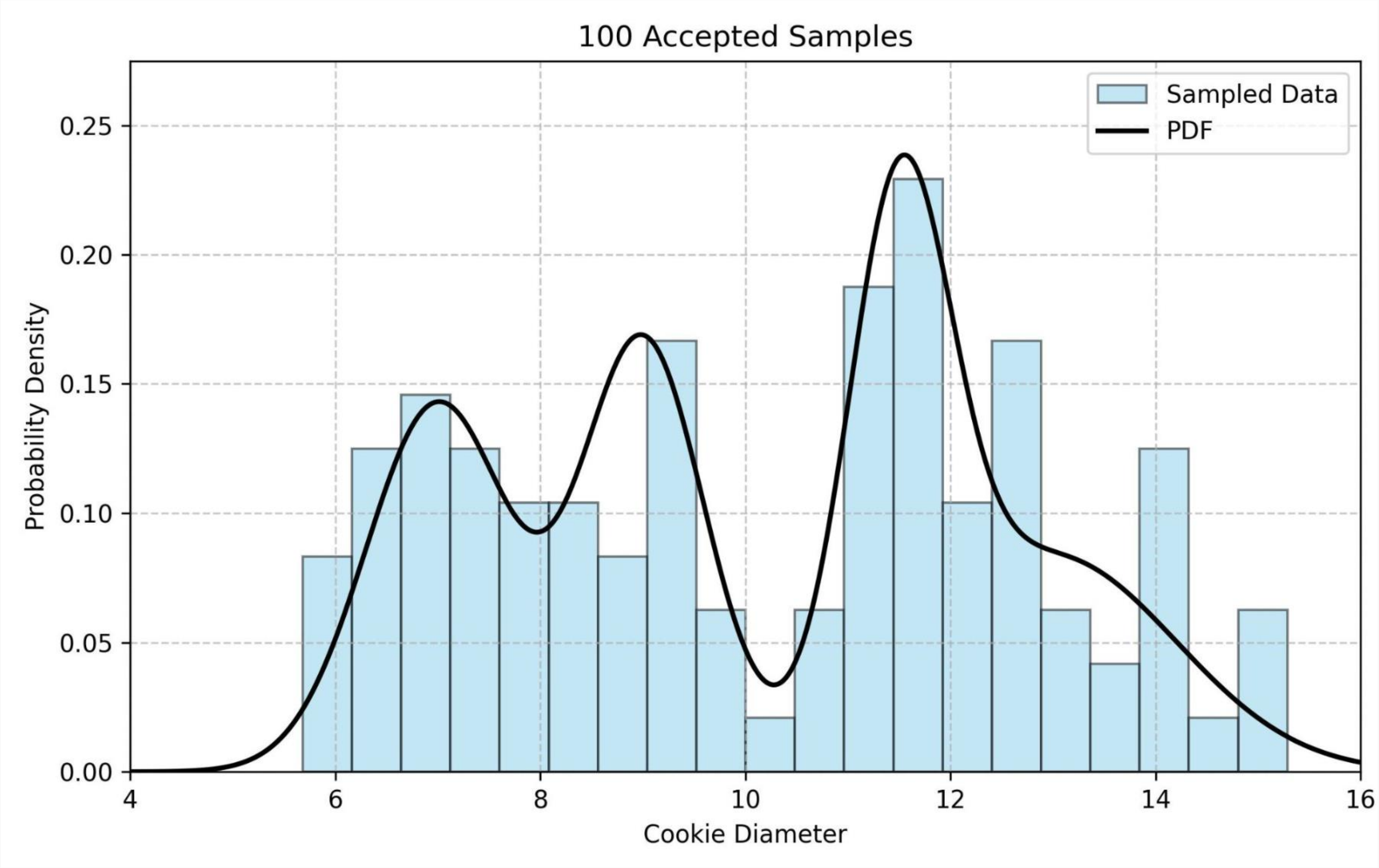
Step 2B: Set Proposal Distribution ($M = 11$)



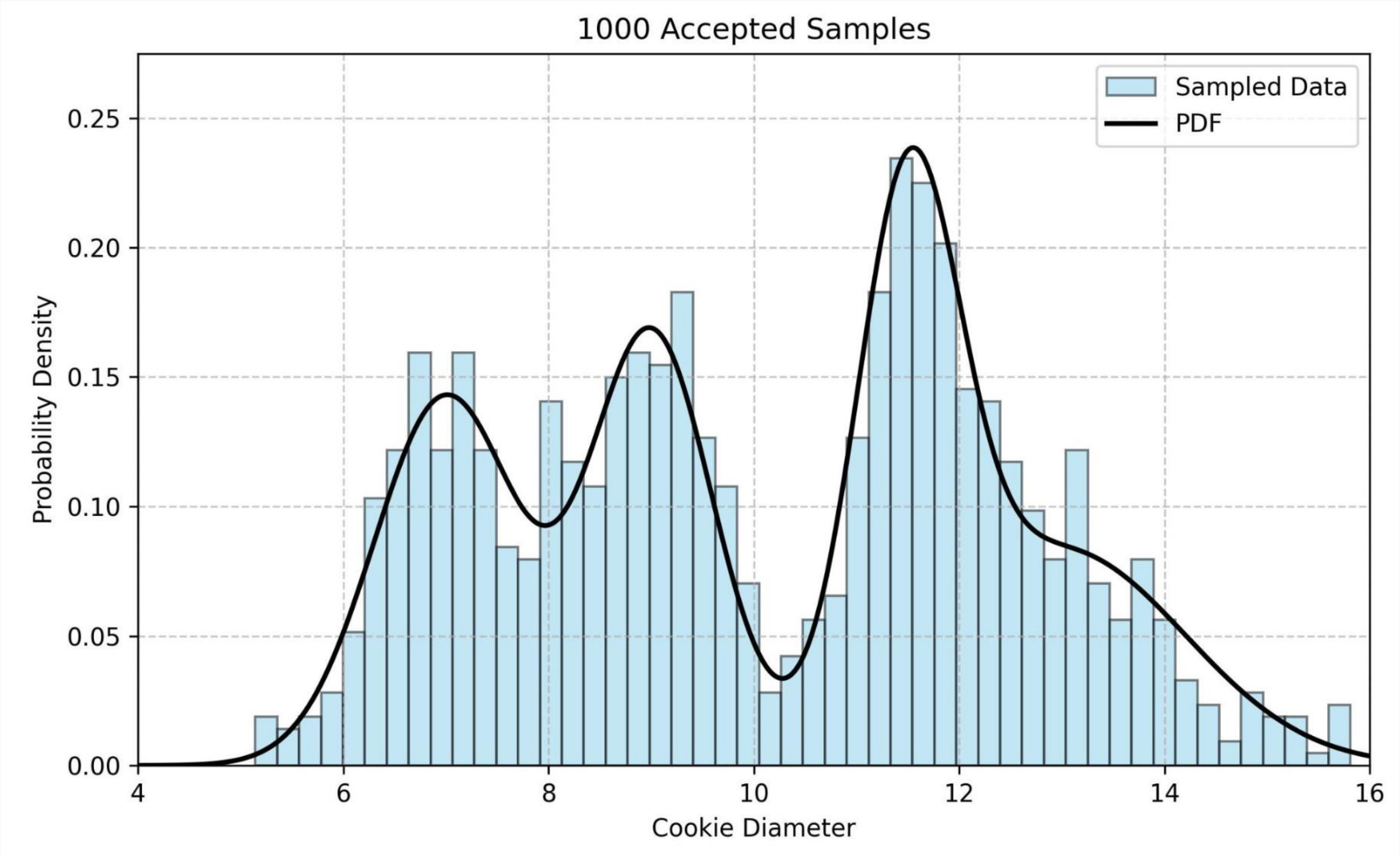
Rejection Sampling

- **Now, we repeat the rest of the steps until we get a sample of some size:**
- **Draw a Candidate Sample:** Generate a random sample x from the proposal distribution $g(x)$.
- **Generate a Uniform Random Number:** Draw $u \sim Uniform(0,1)$.
- **Acceptance Condition:**
- Compute the acceptance ratio $r = f(x)/Mg(x)$.
  Accept the candidate sample x if $u \leq r$; otherwise, reject it.

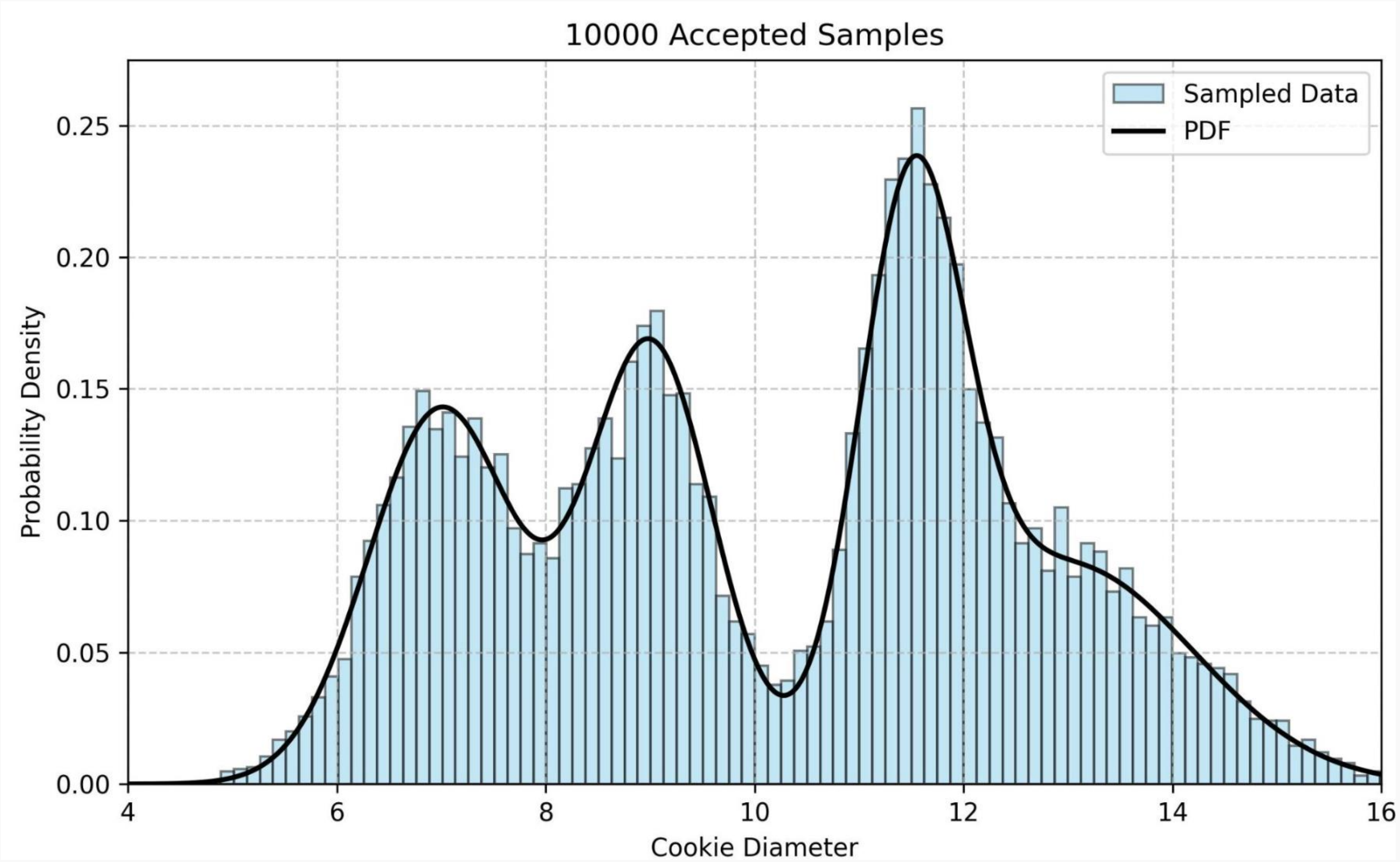
After 100 Accepted Samples…



After 1,000 Accepted Samples…



After 10,000 Accepted Samples…



Cookie Diameter: 1,000 Sample Results

True Expected Value:	$\mu = 10.125$
Estimated Expected Value:	$\hat{\mu} = 10.226$
True Variance:	$\sigma^2 = 5.932$
Estimated Variance:	$\hat{\sigma}^2 = 5.943$

Cookie Diameter: 10,000 Sample Results

True Expected Value:	$\mu = 10.125$
Estimated Expected Value:	$\hat{\mu} = 10.167$
True Variance:	$\sigma^2 = 5.932$
Estimated Variance:	$\hat{\sigma}^2 = 6.004$

Cookie Diameter: 100,000 Sample Results [6 Second Runtime]

True Expected Value:	$\mu = 10.125$
Estimated Expected Value:	$\hat{\mu} = 10.130$
True Variance:	$\sigma^2 = 5.932$
Estimated Variance:	$\hat{\sigma}^2 = 5.944$

Bayesian Rejection Sampling

Now, we repeat the rest of the steps until we get a sample of some size:

Draw a Candidate Sample: Generate a random sample x from the proposal distribution $g(x)$.

Generate a Uniform Random Number: Draw $u \sim Uniform(0,1)$.

Acceptance Condition:

Compute the acceptance ratio $r = f(x)/Mg(x)$.
Accept the candidate sample x if $u \leq r$; otherwise, reject it.

Importance Sampling: an aside

However, the previous algorithm is sometimes **inefficient**; instead of sampling from the distribution, we can use **importance sampling**:

Choose a proposal distribution from which it is easy to sample.

Draw samples from *q(x)*.

Assign a weight to each sample based on the ratio of the target distribution *p(x)* to the proposal distribution *q(x)*:

4. Use the weighted samples to compute the desired expectation or integral.

Outline

- Review: logistics and logs
- Beta-Binomial Model
- Hierarchical Modeling
- Posterior Predictive Value
- Catching our Breathe
- Rejection Sampling
- **MCMC**
- Metropolis-Hastings

Limitation of Rejection (& Importance) Sampling

- The main problem with rejection sampling is **computational efficiency**:

- It could take ages in rejection sampling before a sample is accepted.
- It could take many many samples in importance sampling before reweighted samples are sufficiently reliable to calculate posterior statistics.
- With both methods, the choice of proposal distribution makes a massive difference.

An Alternative Approach: MCMC!

- **Markov Chain Monte Carlo** is another method we can use.
  - **Markov Chain**: a memoryless random process.
  - **Monte Carlo**: estimates based on randomly generated samples.
    - We’ve been doing this already in rejection sampling and bootstrapping!

- **Main Idea**: Construct a random process which, as it evolves, has outputs that start resembling samples from the posterior distribution.

What is a Markov Chain?

- A Markov Chain is an algorithm that evolves in discrete times steps.
- $\theta^{(t)} \subset \Omega$ denotes state of algorithm at step t.
  - State space Ω is a set of all possible states for the algorithm.
- We denote the evolution of the algorithm using $\theta^{(0)}, \theta^{(1)}, \dots, \theta^{(t)}$

- To denote the probability density of $\theta^{(t)}$ given the history of the process, we use:

$$p\left(\theta^{(t)} | \theta^{(t-1)}, \dots, \theta^{(0)}\right)$$

The Markov-ian Property

- A central property of Markov Chains is that updates are random but memoryless:

$$p\left(\theta^{(t)}|\theta^{(t-1)}, \dots, \theta^{(0)}\right) = p\left(\theta^{(t)}|\theta^{(t-1)}\right)$$

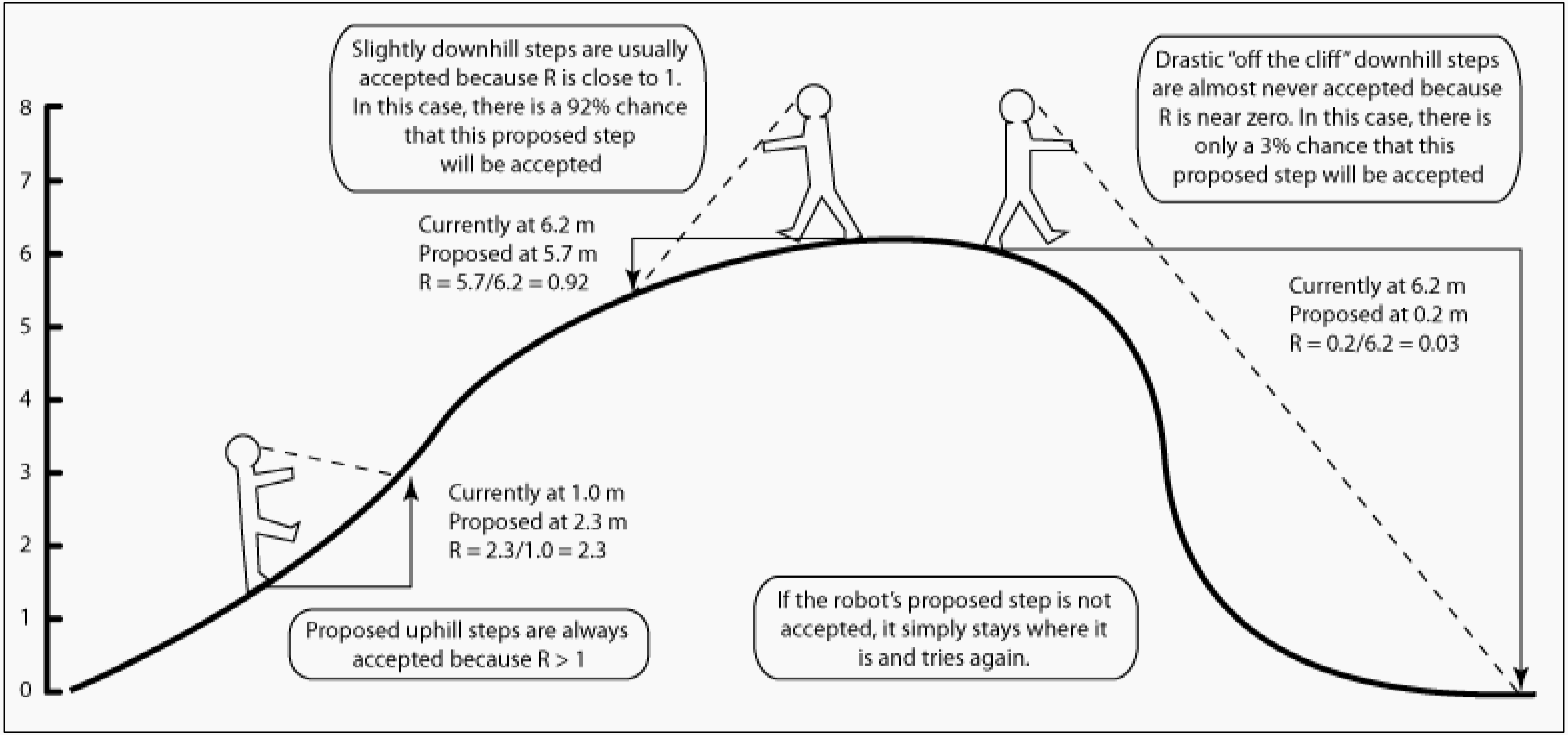
  - It doesn’t matter where I’ve been.
  - It only matters where I am currently at.
- When does this property hold in real life in a ***random process***?

- **So how do we use this idea to sample from a posterior distribution?**

Outline

- Review: logistics and logs
- Beta-Binomial Model
- Hierarchical Modeling
- Posterior Predictive Value
- Catching our Breathe
- Rejection Sampling
- MCMC
- **Metropolis-Hastings**

Metropolis-Hastings: a random walk through the distribution


Slightly downhill steps are usually accepted because R is close to 1. In this case, there is a 92% chance that this proposed step will be accepted
Drastic "off the cliff" downhill steps are almost never accepted because R is near zero. In this case, there is only a 3% chance that this proposed step will be accepted
Currently at 6.2 m
Proposed at 5.7 m
R = 5.7/6.2 = 0.92
Currently at 6.2 m
Proposed at 0.2 m
R = 0.2/6.2 = 0.03
Currently at 1.0 m
Proposed at 2.3 m
R = 2.3/1.0 = 2.3
Proposed uphill steps are always accepted because R > 1
If the robot's proposed step is not accepted, it simply stays where it is and tries again.
8
7
6
5
4
3
2
1
0

Metropolis-Hastings Algorithm

- Let $q(y|x)$ denote a chosen proposal distribution (we pick this).
    - $q(y|x)$ gives density for moving to y from x
- Let $p(\theta|data)$ be our target limiting distribution.

Algorithm (One Step): Currently at $\theta^{(t)}$

- Generate proposal: $\theta^* \sim q(y|\theta^{(t)})$
- Compute acceptance probability:

$$a = \min\left\{1, \frac{f(\theta^*)q(\theta^{(t)}|\theta^*)}{f(\theta^{(T)})q(\theta^*|\theta^{(t)})}\right\}$$

- With probability a set $\theta^{(t+1)} = \theta^*$. Otherwise, set $\theta^{(t+1)} = \theta^{(t)}$

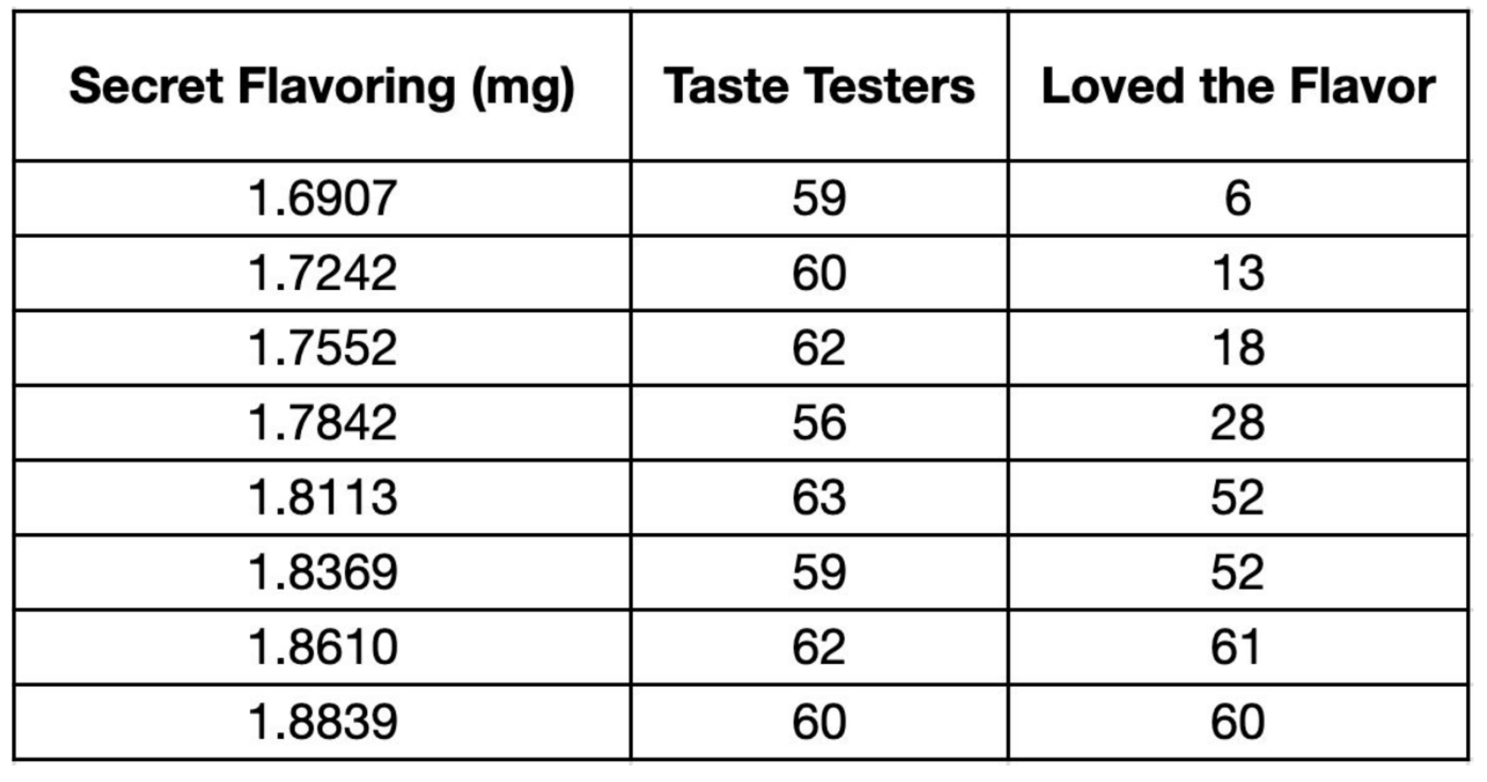
Interpreting Metropolis Hastings

$$a = \min\left\{1, \frac{f(\theta^{*})q(\theta^{(t)}|\theta^{*})}{f(\theta^{(T)})q(\theta^{*}|\theta^{(t)})}\right\}$$

- Uphill proposals (ones that take the Markov Chain to a local maximum) are always accepted.
- Downhill proposals (ones that move away from a local maximum) are accepted with probability equal to the relative heights of the posterior density at the proposed and current values.

Putting Things Into Practice: Skittles

The company behind Skittles is contemplating a new flavor: mango. They need to figure out how much of their secret flavoring to add to each skittle to maximize the number of people who enjoyed the new flavor. The company shared the following taste test data:



Secret Flavoring (mg)	Taste Testers	Loved the Flavor
1.6907	59	6
1.7242	60	13
1.7552	62	18
1.7842	56	28
1.8113	63	52
1.8369	59	52
1.8610	62	61
1.8839	60	60

Skittles Modeling

- What is the response variable? What is the predictor? What sort of model should we use?
- For each of the 8 observations: $Y_i \sim \mathrm{Binom}(n_i, p_i)$
- What is the PMF (the likelihood)?

$$P(Y_i = y_i) = \binom{n_i}{y_i} p_i^{y_i} (1 - p_i)^{n_i - y_i}$$

- We will use a logistic model: $\ln\left(\frac{p_i}{1-p_i}\right) = \alpha + \beta x_i$
  - Recall: this is equivalent to $p_i = \frac{e^{\alpha+\beta x_i}}{1+e^{\alpha+\beta x_i}}$
- The goal is to perform inference for the two parameters, α and β.
- Let’s take a Bayesian approach!

Priors become Posteriors

- We choose proper priors for both parameters of interest:

$$\alpha \sim N(0{,}100^2)$$
$$\beta \sim N(0{,}100^2)$$

- This essentially acts like a uniform distribution for each parameter because the variances are so large.

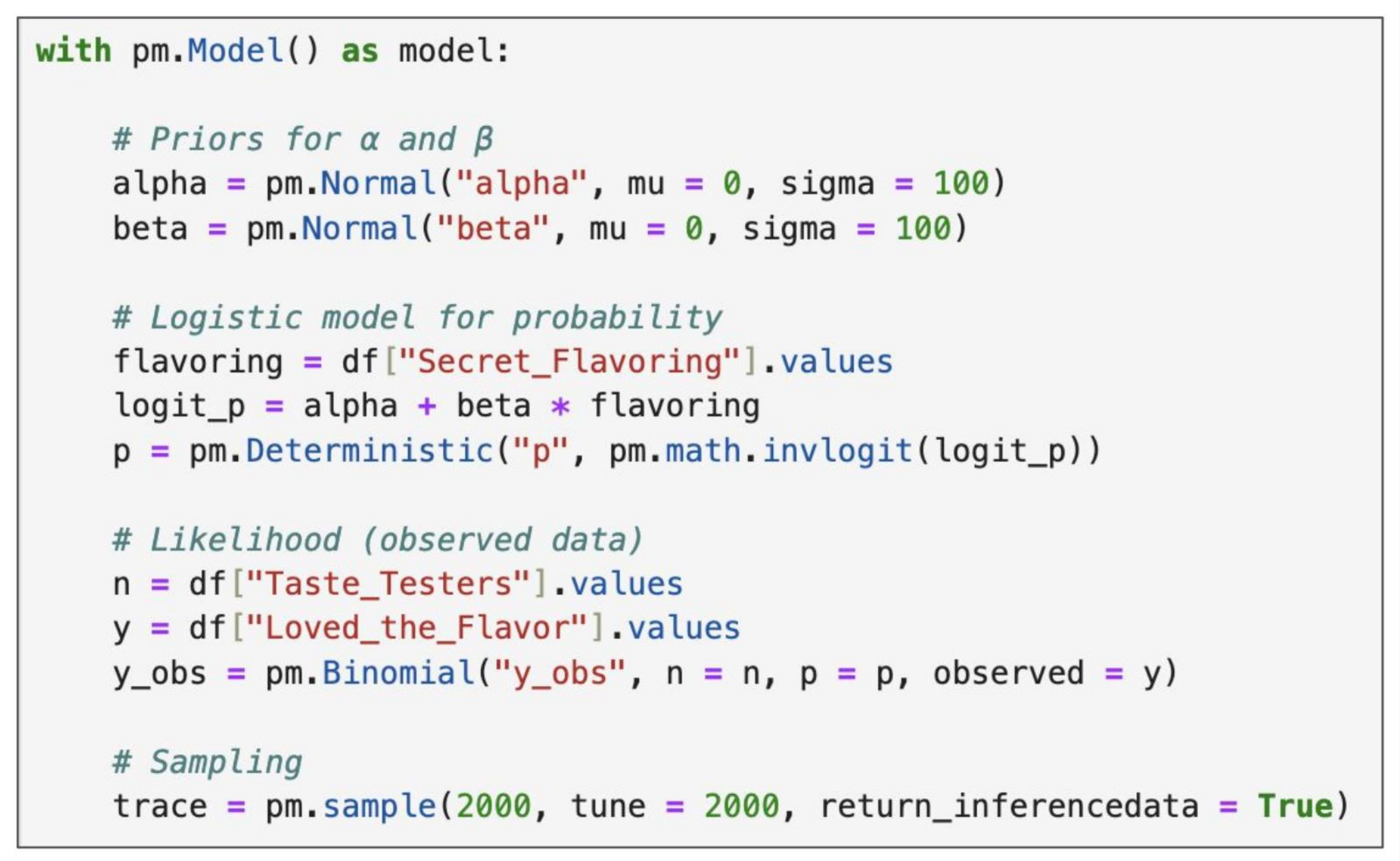
- By Bayes rule, the posterior distribution can be written as:

$$p(\alpha,\beta|y) \propto e^{-\left(\frac{\alpha^2}{2\cdot 100^2}\right)} \cdot e^{-\left(\frac{\beta^2}{2\cdot 100^2}\right)} \cdot \prod_{i=1}^{8} p_i^{y_i}(1-p_i)^{n_i-y_i}$$

Using pymc for Modeling

- The pymc library allows us to abstract away from the details of implementing MCMC and the Metropolis-Hastings algorithm by hand:
- It runs several parallel MCMC samplers with different starting values.
- It simulates values from the Markov chains for a “burn-in” period (before the Markov chains have converged to the stationary distribution), and discard the burn-in simulations.
- It saves simulated values after the burn-in period. These will be the simulated values on which we can perform inferential summaries.

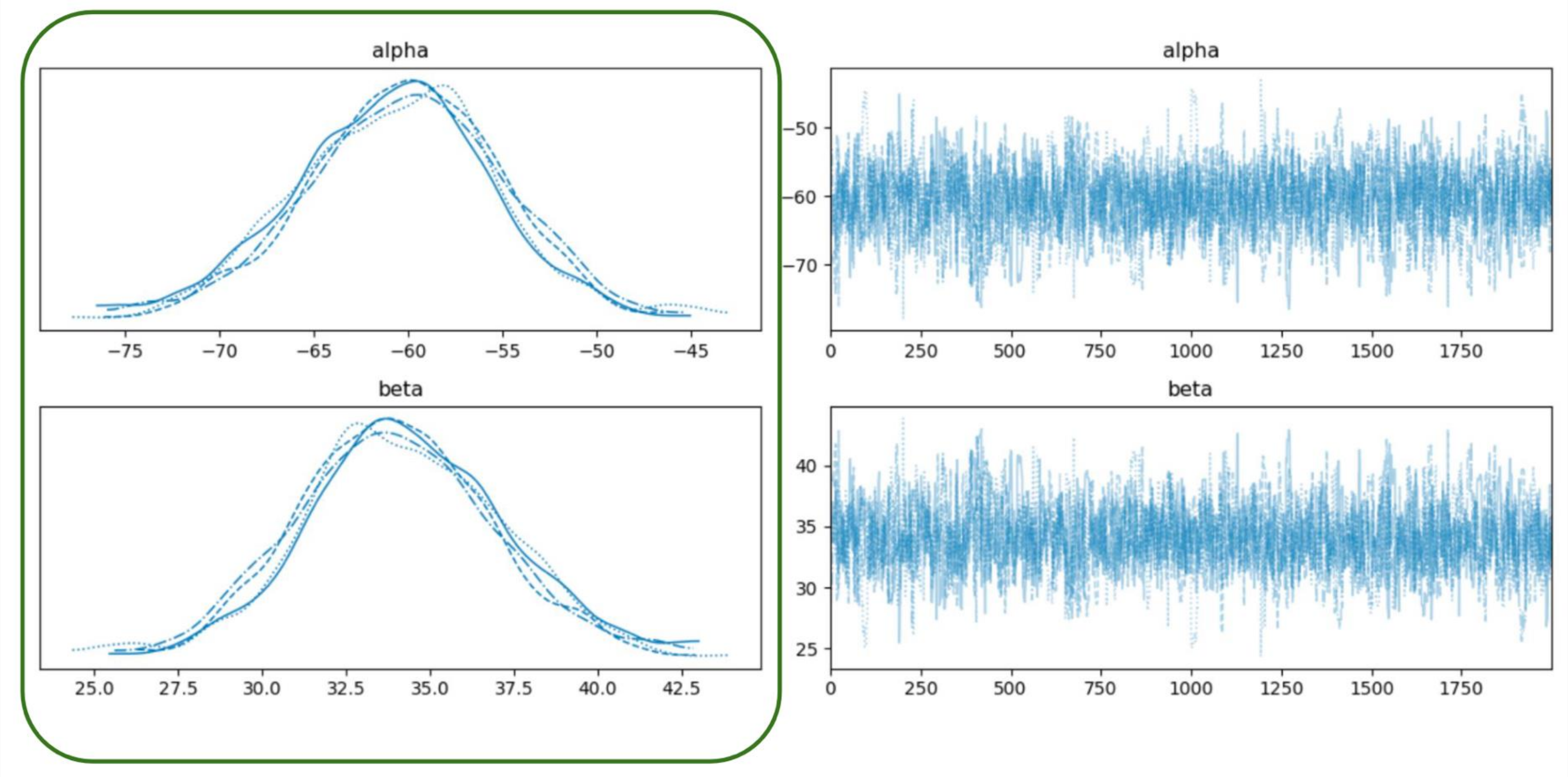
Using pymc for Modeling (cont.)



```
with pm.Model() as model:

    # Priors for α and β
    alpha = pm.Normal("alpha", mu = 0, sigma = 100)
    beta = pm.Normal("beta", mu = 0, sigma = 100)

    # Logistic model for probability
    flavoring = df["Secret_Flavoring"].values
    logit_p = alpha + beta * flavoring
    p = pm.Deterministic("p", pm.math.invlogit(logit_p))

    # Likelihood (observed data)
    n = df["Taste_Testers"].values
    y = df["Loved_the_Flavor"].values
    y_obs = pm.Binomial("y_obs", n = n, p = p, observed = y)

    # Sampling
    trace = pm.sample(2000, tune = 2000, return_inferencedata = True)
```

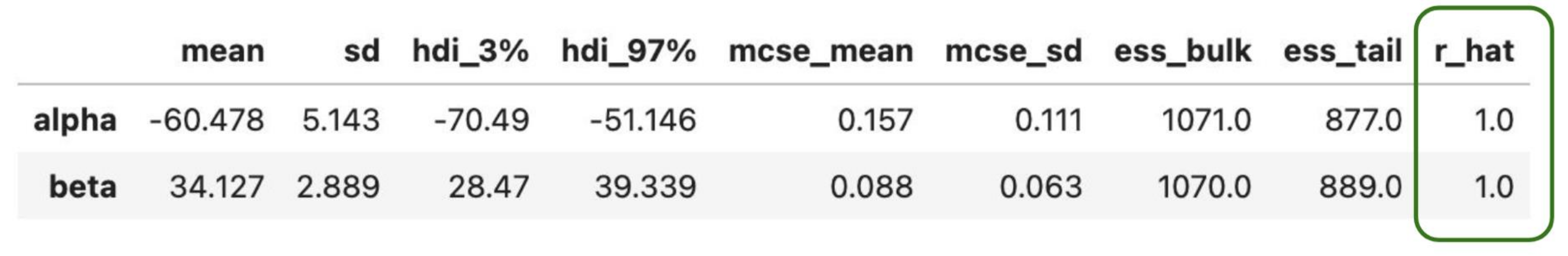
Interpreting pymc Results

The pymc library gives us sampled posterior distributions for the parameters:



Interpreting pymc Results

- We are also provided with some summary information on the stability of the sampling:



	mean	sd	hdi_3%	hdi_97%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
alpha	-60.478	5.143	-70.49	-51.146	0.157	0.111	1071.0	877.0	1.0
beta	34.127	2.889	28.47	39.339	0.088	0.063	1070.0	889.0	1.0

- The r_hat statistic is a numerical measure that indicates whether the Markov chain was run long enough to reach convergence.
  - Values near 1.0 indicate convergence.
  - Large values (say 1.3 or greater) indicate either that the procedure has not been run long enough, or that the parameter itself may be difficult to obtain reasonable samples given strong autocorrelation in the sampler.
  - The statistic essentially computes a ratio of between-chain variance and within-chain variance.

Taste the Rainbow!