MT2002: Statistical Modeling

Structure of a PyMC Model

Muhammad Almas Khan

FAST-NUCES

January 12, 2025

Inference Process

For performing inference, the following steps are needed:

1. **Data:** Raw data observed from the real-world experiment.
2. **Statistical Model:** A probabilistic model describing the relationship between data and parameters.
3. **Analysis/ Decision:** Use plots, metrics (e.g., ROPE) to understand the posterior distribution and make inferences.

Summary of Data Types and Distributions

Heres a summary of common data types, their corresponding distributions, and examples:

Type of Data	Distribution	Example
Binary	Bernoulli	Single toss (head or tail)
	Binomial	10 tosses (number of heads)
Categorical	Categorical	Single animal (cat, dog, bird)
Multiclass	Multinomial	100 animals (cats, dogs, birds)
Continuous	Normal	Height of students
	Exponential	Time between arrivals at a service center

Parts of a PyMC Model

A PyMC model consists of three core components:

1. **Priors**:
   - Capture prior knowledge or belief about model parameters.
   - Example: $\theta \sim \text{Beta}(1, 1)$ (uniform prior, no preference for heads or tails).
   - Choosing Priors:
     - Use domain knowledge if available.
     - If uncertain, use non-informative or weakly informative priors.
2. **Likelihood**:
   - Models how data is generated given the parameters.
   - Example: $y \sim \text{Binomial}(n, \theta)$ for coin tosses.
   - Choosing Likelihood:
     - Use the data-generating process (e.g., Binomial for binary data).
3. **Inference Engine (Backend)**:
   - Uses algorithms (e.g., MCMC) to sample from the posterior distribution.
   - Handles the computation behind the scenes.
   - PyMC automatically selects the method or allows you to customize it.

Example: Coin Toss Inference with PyMC

Task: Estimate the probability of heads (θ) from 7 heads out of 10 tosses.

Model: *Notations book p-32*

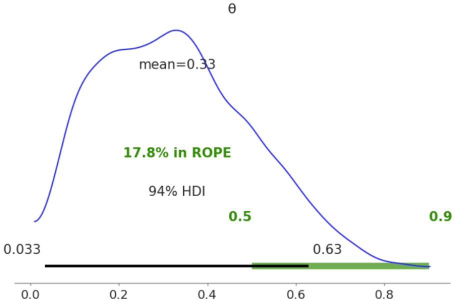
- **Prior:** $\theta \sim \text{Beta}(1, 1)$ (uniform distribution).
- **Likelihood:** $y \sim \text{Binomial}(n = 10, p = \theta)$ (observed data: 7 heads).
- **Inference:** Sample from posterior $P(\theta \mid \text{data})$ using MCMC.

Python Code:

```
#Observed data:  7 heads in 10 tosses
data = [1, 1, 1, 1, 1, 1, 1, 0, 0, 0]
with pm.Model() as coin_toss_model:
theta = pm.Beta("theta", alpha=1, beta=1)
y = pm.Binomial("y", n=1, p=theta, observed=data)
# Perform Inference
trace = pm.sample()
#Visualize posterior distribution
pm.plot_posterior(trace, var_names=["theta"])
```

ROPE Plot

- **ROPE (Region of Practical Equivalence)** helps in determining if the effect size is practically zero.
- It represents the region where parameter values are considered practically equivalent to no effect.



Interpretation: If the posterior distribution falls within the ROPE, we conclude that the effect is negligible.

Priors (*How to decide about prior info in model*)

- Beta distribution with different parameters (weak Vs strong priors).
- Recall the previous slide $\theta \sim \text{Beta}(1, 1)$
- Weak or strong?

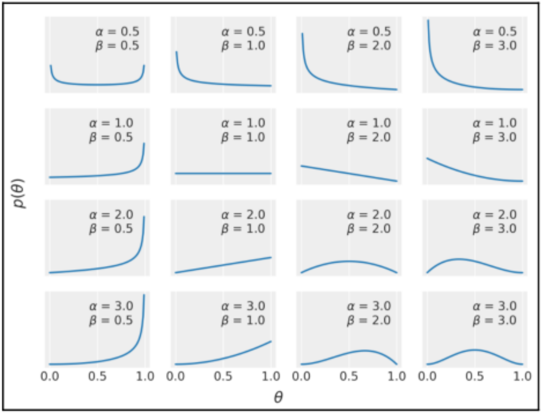
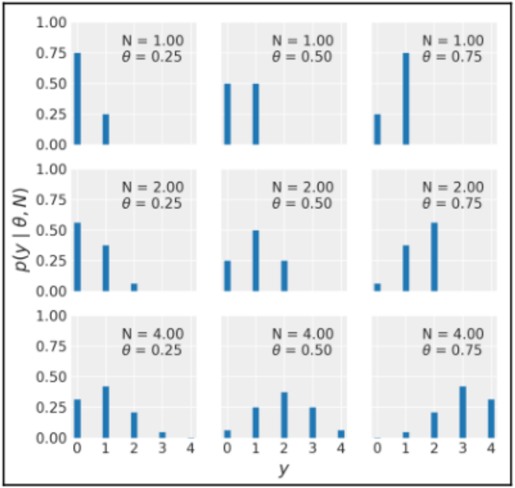


Figure 1.4

Likelihood?

- Likelihood (*How to choose?*)



Computing and plotting the posterior

- Posterior analysis

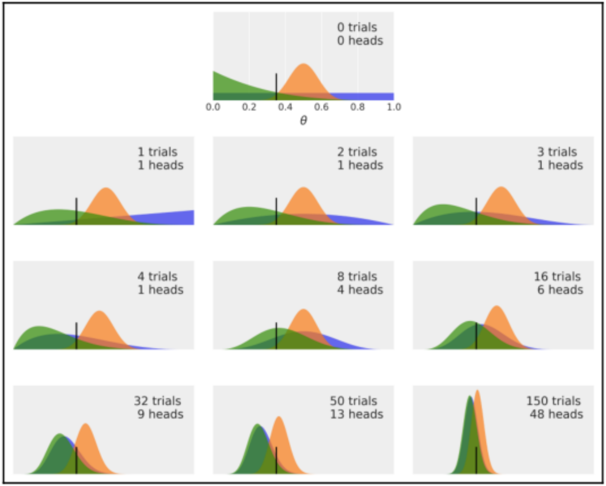


Figure 1.5

Summary

What We Explored:

- **Structure of a PyMC Model:**
  - **Priors:** Initial beliefs about parameters.
  - **Likelihood:** Relates observed data to parameters.
  - **Inference:** Sampling to compute the posterior distribution.
- **What the Model Does:**
  - Combines priors and data via Bayes' theorem.
  - Provides probabilistic estimates of unknown parameters.
- **What the Model Takes as Input:**
  - Observed data (e.g., coin toss results).
  - Prior distributions for unknown parameters.
- **What the Model Produces as Output:**
  - Posterior distribution of parameters.
  - Visualizations (e.g., posterior plots).
- **How It Produces the Output:**
  - Uses an inference engine (e.g., MCMC sampling) to sample the posterior.
  - Requires sufficient data to infer parameter values effectively.

Let's create a basic pymc model!

Ready to explore PyMC in-depth?