Prof. Alex Rogers
Department of Computer Science
University of Oxford
24th October 2020

Probability Distributions and PyMC3 Session 3

Contents

Exercises
Probability Distributions
PyMC3 Implementation
Discrete Distributions
Continuous Distributions

Exercises

1. [**Advanced**] In the session notes we showed that the posterior distribution that results from applying Bayes rule with an uninformed prior and a Bernoulli likelihood function is a beta probability density function. Show that this is also the result when the prior is already a beta probability density function and it is updated with a further Bernoulli likelihood function.

2. A coin is flipped 8 times and comes up heads just two times. Use PyMC3 to model this setting and generate a trace representing the posterior probability of the coin coming up heads. Use that trace to calculate:
   (a) The probability of the next coin flip coming up heads.
   (b) The probability of the next two coin flips both coming up heads.
   (c) The probability of any three out of the next five coin flips coming up heads.

3. Consider the case where a coin has been flipped 6 times and has come up heads twice. This can be represented as a prior with a beta probability density function using the built-in beta function as below:

```
p = pm.Beta('p', 3, 5)
```

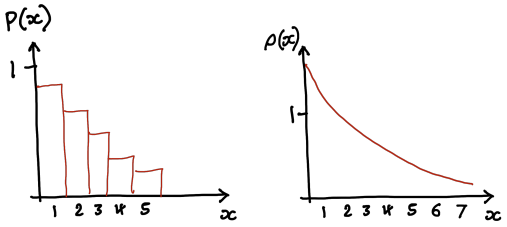
   Build an appropriate model in PyMC3 using this prior and generate a trace representing the posterior probability of the coin coming up heads assuming that two more coin flips are made and both of them come up heads.

4. For what values of a and b does the beta probability density function have a value greater than 0 at $p = 0$ and $p = 1$. What does this mean in terms of the results of the coin flipping experiments discussed in this session.

Probability Distributions

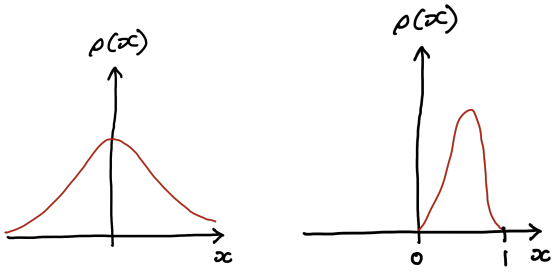
Discrete and Continuous Distributions

A key distinction between the probability distributions that we will consider is whether they are continuous or discrete.



Distributions with Infinite or Bounded Support

We also need to consider the support of a distribution. Continuous distributions may have infinite or bounded support.



Random Variables

When we define variables in PyMC3 we are really defining random variables. The random variable has a distribution from which its actual value will be derived at some future time.

The notation for random variables is often confusing but in general a tilde is used to indicate that the random variable derives its value from the given distribution:

$$X \sim \text{Beta}(3, 5)$$

We may also have multiple such values and use a subscript to indicate this:

$$x_i \sim \text{Beta}(3, 5)$$

random_variables.ipynb

Measurement Noise

We may also make actual observations of realised values of random variables. It is this feature that allows us to describe likelihood functions and to calculate the posterior probability density function of prior distributions:

Assume we make repeated measurements of some physical parameter. Each time we do so, we make small measurement errors. We normally assume that these measurement errors are Gaussian.

$$x_i \sim \text{Normal}\left(\mu, \sigma^2\right)$$

There is some true value of the physical parameter that we are trying to measure, given by μ. Our noisy measurement process is described by the standard deviation σ.

Normal Distribution

The normal distribution is a symmetrical continuous infinite probability density function described by its mean and variance:

$$\rho(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

https://en.wikipedia.org/wiki/Normal_distribution

https://docs.pymc.io/api/distributions/continuous.html

continuous_distributions.ipynb

Prior Distributions

To complete our model we need to describe the prior distributions for the two unknown parameters.

We know that the standard deviation has to be positive. Depending on what we are measuring the expected value may also be strictly positive.

We may know something about the measurement process and have a good intuition for suitable ranges. We should try to incorporate all our domain knowledge into the model and the prior distributions.

We should be careful about forcing low probability outcomes to be zero probability!

Exponential Distribution

The exponential distribution is described by a continuous probability density function with positive support defined by a single parameter.

$$\rho(x) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x \leq 0 \end{cases}$$

https://en.wikipedia.org/wiki/Exponential_distribution

https://docs.pymc.io/api/distributions/continuous.html

Gamma Distribution

The gamma distribution is described by a continuous probability density function with positive support defined by two parameters.

$$\rho(x) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$$

https://en.wikipedia.org/wiki/Gamma_distribution

https://docs.pymc.io/api/distributions/continuous.html

continuous_distributions.ipynb

Measurement Noise

Putting it all together we can declare our model with a normal distribution describing our measurement process and continuous distributions representing our prior beliefs about the parameters of this normal distribution.

Given some observations we can use Markov chain Monte Carlo to infer the posterior probability distribution of these parameters.

measurement_noise.ipynb

Counting Events

We often have settings where we count events and must estimate the underlying rate of some process.

For example, what is the rate that people visit a website if 10 people visit in one hour.

In this case, it is easy to estimate mean rate 10 per hour but how certain are we about this estimate?

Poisson Distribution

The Poisson distribution is a positive-only discrete probability distribution described by a rate parameter:

$$P(n|\mu) = \frac{\mu^n e^{-\mu}}{n!}$$

https://en.wikipedia.org/wiki/Poisson_distribution

https://docs.pymc.io/api/distributions/discrete.html

counting_events.ipynb

Categorical Outcomes

We have already seen examples where there are two outcomes from some trial - a Bernoulli process with a binomial likelihood function.

$$P(n|p,N) = \frac{N!}{n!\,(N-n)!}p^n(1-p)^{N-n}$$

This model can be extended to cases where there are more than two possible outcomes.

Multinomial Distribution

The Multinomial distribution is a discrete probability distribution describing the probability of seeing $n_1, \ldots, n_k$ outcomes in k different classes after N repeated trials:

$$P(n_1, \ldots, n_k|p_1, \ldots, p_k, N) = \frac{N!}{\prod_{i=1}^{k} n_i!}\prod_{i=1}^{k} p_i^{n_i}$$

https://en.wikipedia.org/wiki/Multinomial_distribution

https://docs.pymc.io/api/distributions/multivariate.html

Prior Distribution

For the Bernoulli process with a binomial likelihood function we used a beta probability density function as the prior to represent our belief about a probability that could fall between 0 and 1.

In the multinomial case we use a Dirichlet distribution for the same purpose.

The Dirichlet distribution is a multivariate extension of the beta distribution.

Dirichlet Distribution

The Dirichlet distribution is a continuous multivariate probability density function described a set of parameters:

$$\rho(x_1, \ldots, x_k|a_1, \ldots, a_k) = \frac{\Gamma\left(\sum_{i=1}^{k} a_i\right)}{\prod_{i=1}^{k}\Gamma\left(a_i\right)}\prod_{i=1}^{k} x_i^{a_i-1}$$

https://en.wikipedia.org/wiki/Dirichlet_distribution

https://docs.pymc.io/api/distributions/multivariate.html

rolling_dice.ipynb

Next Time

Building Models in PyMC3