Bayesian-Beta-Bernoulli-Conjugate

September 13, 2021

1 DATA 5600: Introduction to Regression and Machine Learning for Analytics

1.1 Notes on the Bayesian Beta-Bernoulli Conjugate Model

Author: Tyler J. Brough Last Update: September 13, 2021

[41]:
```
import numpy as np
from scipy import stats
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_style('darkgrid')
plt.rcParams['figure.figsize'] = [10, 5]
```

1.2 Introduction

I would like to take a step back and consider the process of Bayesian inference in a simpler model. We will first consider the simple case of flipping a coin and estimating the proportion of heads. Let $Y \in \{0, 1\}$ be our random variable for the outcome of a coin flip where $y = 1$ stands for heads and $y = 0$ stands for tails.

We will first consider the case where we have a fair coin. That is, where $\theta = 0.5$, and θ is the proportion of heads in the total number of flips. By definition, each flip is independent of the last. We usually model coin flipping with the *Binomial distribution*. So we will first take some time to understand the basic details of that particular distribution.

Note that we are no so interested in the coin flipping problem per se, but rather are interested in it because it offers a simple setting in which to come to grips with the process of Bayesian statistical inference. Although, the binomial distribution does play an important role in finance, and coin flipping can be replaced in a more general setting for any binary outcome. Therefore, the process of statistical inference that we derive for coin flipping can apply much more broadly to more interesting questions. Here are some examples of binary random variables that we might be interested in:

- The existence of God
- In a two-party election, what is the probability that a person will vote for each candidate?
- If you are an airline analyst, what is the probability that a passenger will show up for his flight? Should you overbook? By how much?

- For a given job training program, what is the probability that we will see at least a 10% improvement?
- For a given survey question, what is the probability that a respondent will agree or disagree?
- In online advertising what is the probability of a click-through?

1.3 The Bernoulli and Binomial Distributions

Please see the following Wikipedia artciles on the *Bernoulli* and *Binomial* distributions:

- The Bernoulli distribution: https://en.wikipedia.org/wiki/Bernoulli_distribution
- The Binomial distribution: https://en.wikipedia.org/wiki/Binomial_distribution

1.3.1 The Bernoulli Distribution

Consider flipping a coin a single time and recording the outcome. Let the random variable Y represent the outcome of the single flip. Also let θ be the probability of the coin landing on heads (if it is a fair coin, then $\theta = 0.5$). Then we can say the following:

- $p(y = 1|\theta) = \theta$
- $p(y = 0|\theta) = 1 - \theta$

We can combine these equations as follows:

$$p(y|\theta) = \theta^y(1 - \theta)^{(1-y)}$$

This is the *Bernoulli distribution* for $Y \in \{0, 1\}$ and $0 \leq \theta \leq 1$.

We can show that:

- $E(X) = \theta$
- $Var(X) = \theta(1 - \theta)$

Note that when $y = 1$ the expression simplifies to θ, and when $y = 1$ the expression becomes $(1-\theta)$

1.3.2 The Binomial Distribution

Now consider the case when we flip a coin some number of times, denote this by N. We might then ask how many times a heads came up in the total number of flips. In this case, θ is still the probability of heads for a single flip, but we can also interpret it as the proportion of heads in N flips. Each flip is called a Bernoulli trial. So the Binomial distribution is a generalization of the Bernoulli distribution. In fact, when $N = 1$, the Binomial distribution collapses to the Bernoulli distribution.

Let $X \in \{1, \ldots, N\}$ represent the number of heads in N trials. In other words, let $X = \sum_{i=1}^{N} Y_i$, where each Y_i is the outcome of a single trial. Then X has a *Binomial* distribution (or X is distributed according to the Binomial distribution):

$$p(X|N) = \binom{N}{X}\theta^X(1-\theta)^(N-X)$$

where

$$\binom{N}{X} = \frac{N!}{(N-X)!X!}$$

is the number of ways to choose X items from N.

Say we have the data set $D = \{y_1, \ldots, y_N\}$ where each y_i is the observed outcome of a single flip (i.e. a single Bernoulli trial). Notice that the likelihood function becomes:

$$p(D|\theta) = \prod_{i=1}^{N} p(y_i|\theta) = \prod_{i=1}^{N} \theta^{y_i}(1-\theta)^{(1-y_i)} = \theta^{N_1}(1-\theta)^{N_0}$$

where $N_1 = \sum_i y_i$ is the number of heads and $N_0 = \sum_i (1-y_i)$ is the number of tails. Obviously, $N = N_1 + N_0$.

This the Binomial likelihood function for our data.

1.4 Parameter Estimation

Say we have a coin with probability of heads θ. How do we estimate θ from a sequence of coin tosses $D = \{Y_1, \ldots, Y_N\}$, where $Y_i \in \{0, 1\}$?

1.4.1 Maximum Likelihood

One approach is to find a maximum likelihood estimate (**NB:** a frequentist approach):

$$\hat{\theta}_{ML} = \arg\max_{\theta} p(D|\theta)$$

Given $D = \{y_1, \ldots, y_N\}$, the likelihood is

$$p(D|\theta) = \theta^{N_1}(1-\theta)^{N_0}$$

as shown above. To find the maximum likelihood it is sometimes easier to work with the log-likelihood function.

The log-likelihood function is

$$L(\theta) = \log p(D|\theta) = N_1 \log \theta + N_0 \log 1 - \theta$$

Solving for $\frac{dL}{d\theta} = 0$ yields

$$\hat{\theta}_{ML} = \frac{N_1}{N_1 + N_0} = \frac{N_1}{N}$$

It can be shown (although we won't here) that this estimator has all the good properties that we desire from a frequentist estimator such as unbiasedness, consistency, and efficiency. It also has the nice feature of simplicity.

Suppose we flip the coin $N = 100$ times and we observe $N_1 = 48$ heads. Then, the maximum likelihood estimator gives us:

$$\hat{\theta}_{ML} = \frac{N_1}{N} = \frac{48}{100} = 0.48$$

We might simulate this in Julia as follows:

[42]:
```
theta = 0.5
N = 10
flips = np.random.binomial(n=1, p=theta, size=N);
flips[:20]
```

[42]:
```
array([0, 0, 0, 0, 1, 0, 1, 0, 0, 1])
```

[43]:
```
N1 = np.sum(flips)
theta_hat_ml = N1 / N
print(f"\nThe Maximum Likelihood Estimator is: \t{theta_hat_ml : 0.6f}\n")
print(f"The Mean Squared Error is: \t\t{np.sqrt((theta - theta_hat_ml)**2.0) :␣
↪0.6f}\n\n")
```

```
The Maximum Likelihood Estimator is:     0.300000

The Mean Squared Error is:               0.200000
```

But what happens if we flip the coin $N = 3$ times and we observe $N_1 = 0$ heads? Then we predict that heads are impossible!

$$\hat{\theta}_{ML} = \frac{N_1}{N} = \frac{0}{3} = 0$$

This is known as a ***sparse data problem***: if we fail to see something in our data sample, then we predict that it will never happen in the future. We will see how the Bayesian approach avoids this problem.

1.4.2 Bayesian Estimation

The Bayesian approach is to treat θ as an uncertain variable and to use the rules of probability to characterize that uncertainty. In addition, we can use Bayes' Rule to update our belief about θ having confronted the evidence $D = \{y_1, \ldots, y_N\}$:

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{\int_{\theta'} p(\theta', D)}$$

Notice that the answer to the Bayesian inference problem is an entire probability distribution that now characterizes our uncertainty regarding θ having confronted the evidence. If we wish to have a point estimate, we can report the mean or mode of the posterior distribution. We can also look at the standard deviation of the posterior distribution to give a sense of our uncertainty.

Below we will detail the process of Bayesian inference for coin flipping in some detail. We will also compare it with the maximum likelihood estimator.

1.5 Bayesian Inference for Coin Flipping

1.5.1 The Likelihood Function

As before, we have the Binomial likelihood function:

$$p(D|\theta) = \theta^{N_1}(1-\theta)^{N_0}$$

Now that we have a good likelihood function, we turn our attention to a prior distribution that might characterize our beliefs about θ before we observe the data. We might like to use a *Natural Conjugate Prior* distribution. For a Binomial likelihood function, one such prior distribution is the *Beta* distribution.

Let's look at the Beta distribution next.

1.5.2 The Beta Distribution

First of all, take a look at the Wikipedia article on the Beta distribution: https://en.wikipedia.org/wiki/Beta_distribution

The Beta distribution is expressed as follows:

$$p(\theta) = Be(\theta|a,b) = \frac{1}{B(a,b)}\theta^{(a-1)}(1-\theta)^{(b-1)}$$

for θ in the interval $[0,1]$ and where $B(a,b)$ is known as the *Beta Function*. See the Wikipedia article here: https://en.wikipedia.org/wiki/Beta_function. This is a special function with the following property:

$$B(a,b) = \int_0^1 \theta^{(a-1)}(1-\theta)^{(b-1)}d\theta.$$

In `Python` the Beta function is `scipy.special.beta(a, b, out=None)`, and the Beta distribution is `scipy.stats.beta = <scipy.stats._continuous_distns.beta_gen object>`.

- ***Note:*** see here for documentation on the beta function: https://docs.scipy.org/doc/scipy/reference/generated/scipy.special.beta.html
- ***Note:*** see here for documentation on the beta distribution: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.beta.html

Note: the random variable in the Beta distribution is θ, and that a and b are its parameters.

It turns out that we can express the Beta function in terms of another special function, the *Gamma Function* (see here: https://en.wikipedia.org/wiki/Gamma_function).

The Gamma function can also be expressed as an integral:

$$\Gamma(a) = \int_0^\infty t^{(a-1)} exp(-t) dt$$

It can be shown that

- $\Gamma(a) = (a-1)!$
- $B(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$

It can also be shown that:

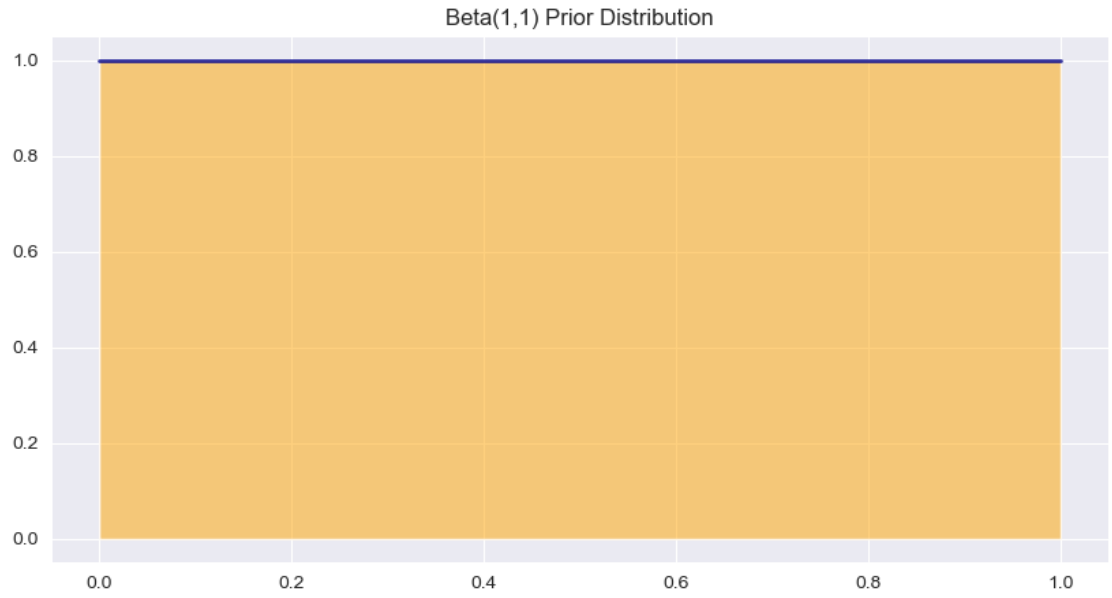
- $E(\theta) = \frac{a}{a+b}$

The Beta distribution can generate many different shapes. Let's look at some below using some `Python` code.

Let's start off with the case when $a = b = 1$

[44]:
```
## Helper function to plot Beta priors
def plot_beta_prior(a=1, b=1):
    x = np.linspace(0, 1, 1000)
    y = stats.beta.pdf(x, a, b)
    plt.plot(x, y, lw = 2.0, color='darkblue', alpha=0.8)
    plt.fill_between(x, y, facecolor='orange', alpha=0.5)
    plt.title(f"Beta({a},{b}) Prior Distribution")
    plt.show()
```

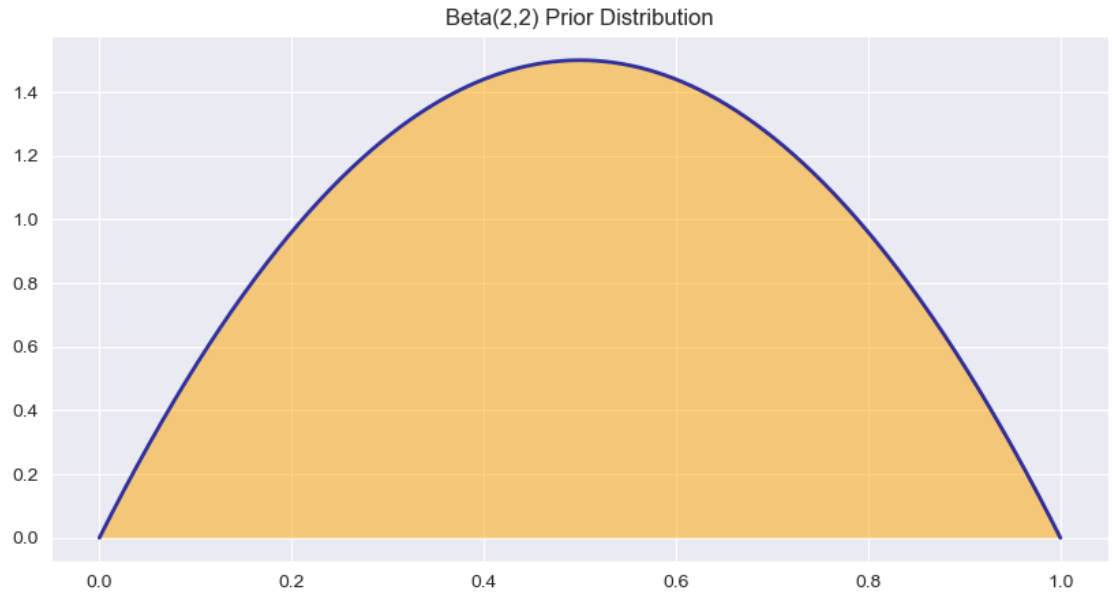
[45]:
```
## Plot a Uniform(0, 1) which is a special case of the Beta
## Uniform(0, 1) = Beta(a=1, b=1)

plot_beta_prior()
```



This turns out to be a special case of the Beta distribution known as the ***uniform distribution*** (see here: https://en.wikipedia.org/wiki/Uniform_distribution_(continuous)). It's called the Uniform distribution because it treats each value in the interval $[0, 1]$ uniformly. That is, each value in $[0, 1]$ is equally likely. Sometimes this form of the Beta prior is used to characterize complete ignorance (e.g. "I don't know, so I'll treat every value as equally likely.").

Next let's look at the case when $a = b = 2$.

[46]:
```
plot_beta_prior(a=2, b=2)
```



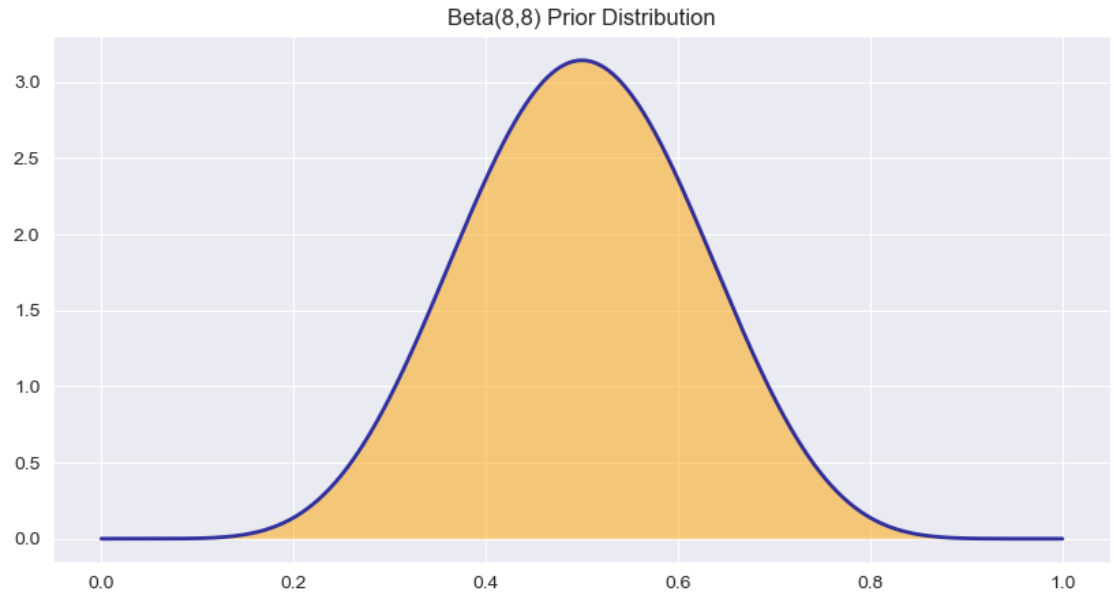
Here the distribution is centered around the value $\theta = 0.5$, but it is quite diffuse - expressing quite a bit of uncertainty.

Let's look at a case when we are just a bit more certain: $a = b = 4$.

[47]:
```
plot_beta_prior(a=4, b=4)
```

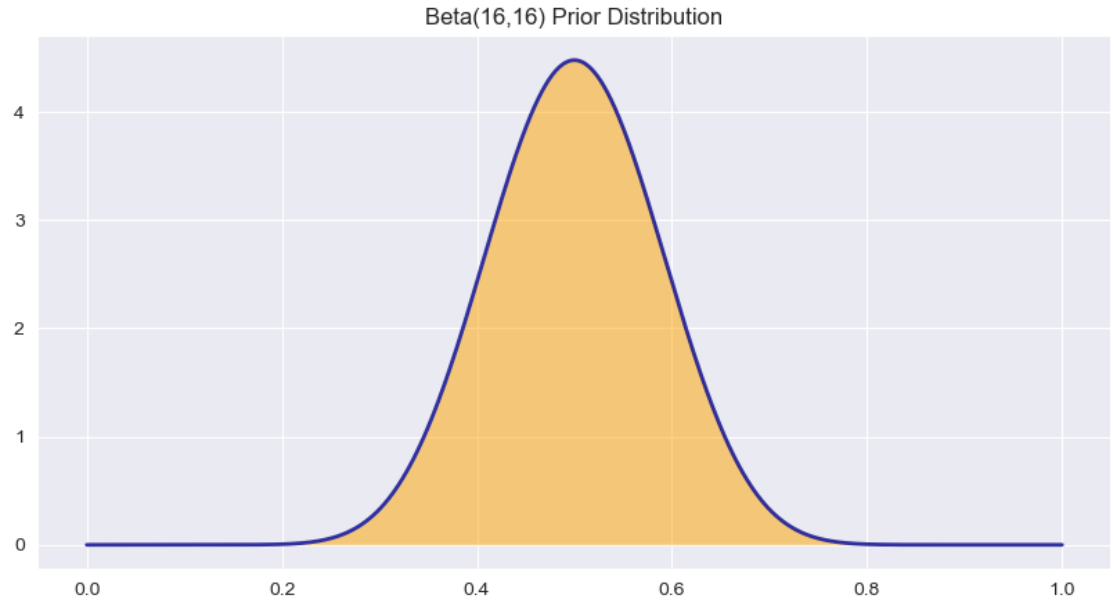
And even a little bit more certain: $a = b = 8$

[48]:
```
plot_beta_prior(a=8, b=8)
```



We can continue this process: $a = b = 16$

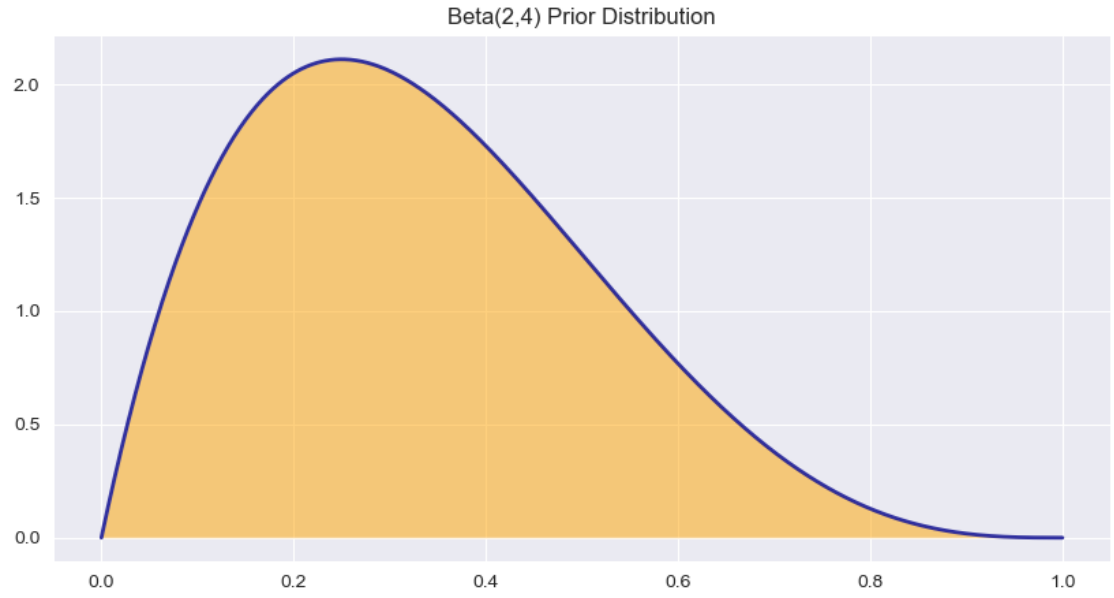
[49]:
```
plot_beta_prior(a=16, b=16)
```



Note: play with these plots and generate your own!

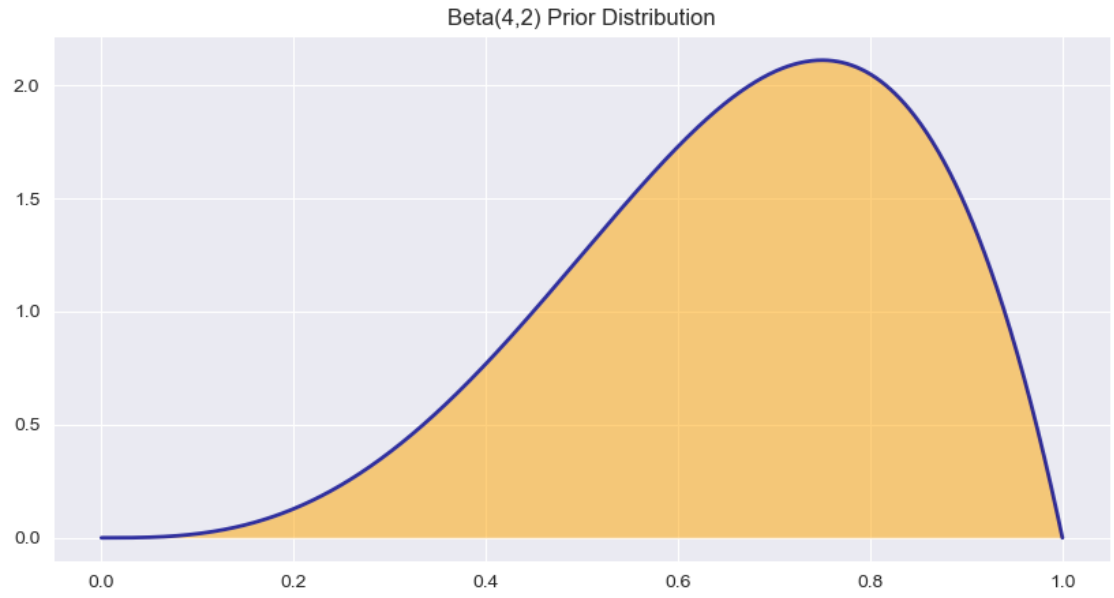
We can also have a situation that is not symetric about center: $a = 2$ and $b = 4$

[50]:
```
plot_beta_prior(a=2, b=4)
```



And the other way too!

[51]:
```
plot_beta_prior(a=4, b=2)
```



We can even get some funkier shapes like this: $a = b = 0.5$

We can see that the Beta distribution is quite flexible and can express quite a number of different prior beliefs about θ depending on the chosen hyperparameters a and b.

One way of eliciting a prior belief about *theta* is think about having witnessed some *virtual* flips of the coin. For example, we might expect that it is a fair coin, so that the distribution ought to be centered around 0.5. We might parameterize this with the variable m, thinking of m as the mean proportion of heads in our virtual experiment:

$$m = \frac{a}{(a+b)}$$

Further, we might think of the variable $n = a+b$ as our virtual number of flips, and a as the virtual number of heads and b as the virtual number of tails. As n gets larger we are expressing greater and greater confidence. However, if n is small, say $n = 4$, then we might not be so confident (i.e. we are imagining only having witnessed 4 flips - it wouldn't take too much data to convince us that we are wrong).

Solving these two equations gives us:

$$\begin{aligned} a &= m \times n \\ b &= (1-m) \times n \end{aligned}$$

1.5.3 The Posterior Distribution

Now we have the likelihood function and a natural conjugate prior. Let's use Bayes' Rule to derive the posterior distribution.

$$\begin{aligned} p(\theta|D) &\propto p(D|\theta) \times p(\theta) \\ &\propto [\theta^{N_1}(1-\theta)^{N_0}][\theta^{(a-b)}(1-\theta)^{(b-1)}] \\ &= \theta^{N_1+a-1}(1-\theta)^{N_0+b-1} \end{aligned}$$

Note: this is a $Beta(\theta|N_1+a, N_0+b)$ distribution, which demonstrates that it is conjugate (i.e. of the same form as the prior).

1.6 Applying the Model to Coin Flipping

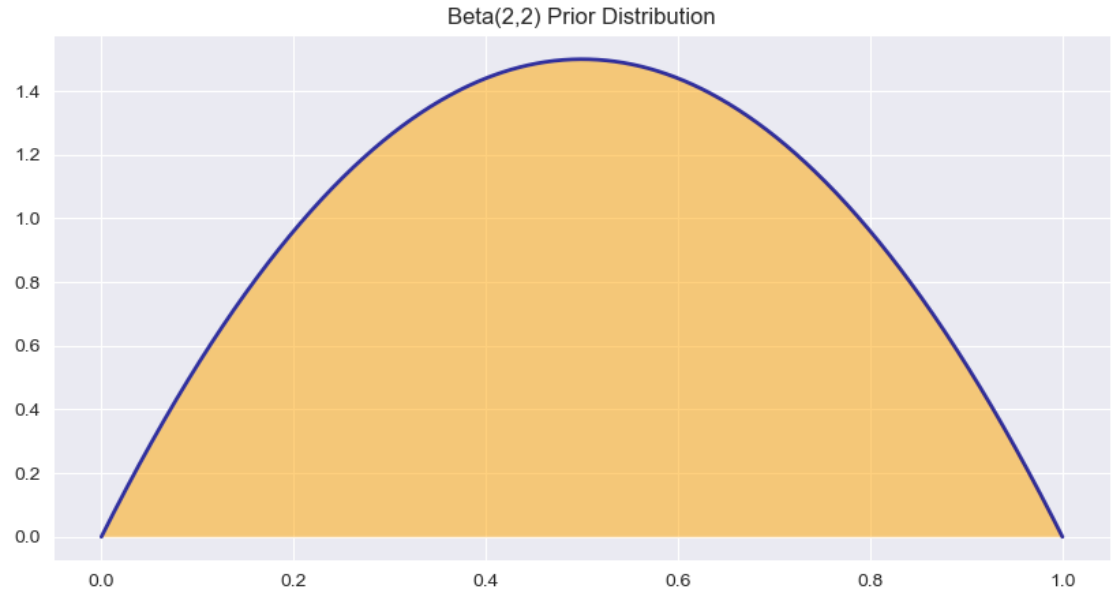
We can now use this model to revise our beliefs as the data arrive (i.e. after each coin flip). This is called sequential analysis, and note that the Bayesian paradigm lends itself quite naturally to this kind of *on-line* analysis due to the process of Bayesian updating. Whereas, in the frequentist case we have to precommit ourselves to conducting a study to flip the coin N times, and then count the number of heads.

Let's look at an example. Suppose you start with the belief that $a = b = 2$. Then you observe the first flip $y = 1$ (a heads) to get $Beta(\theta|3, 2)$, so the mean shifts from $E(\theta) = 2/4$ to $E(\theta|D) = 3/5$. We see that the hyperparameters a and b do in fact act like "pseudo counts" and correspond to "virutal" heads or tails. We might call $n = a+b$ the effective sample size (it plays a role analogous to $N = N_1 + N_0$).

Let's simulate the process and see how our beliefs are updated with each flip.

[52]:
```
a = 2 # virtual heads
b = 2 # virtual tails
```

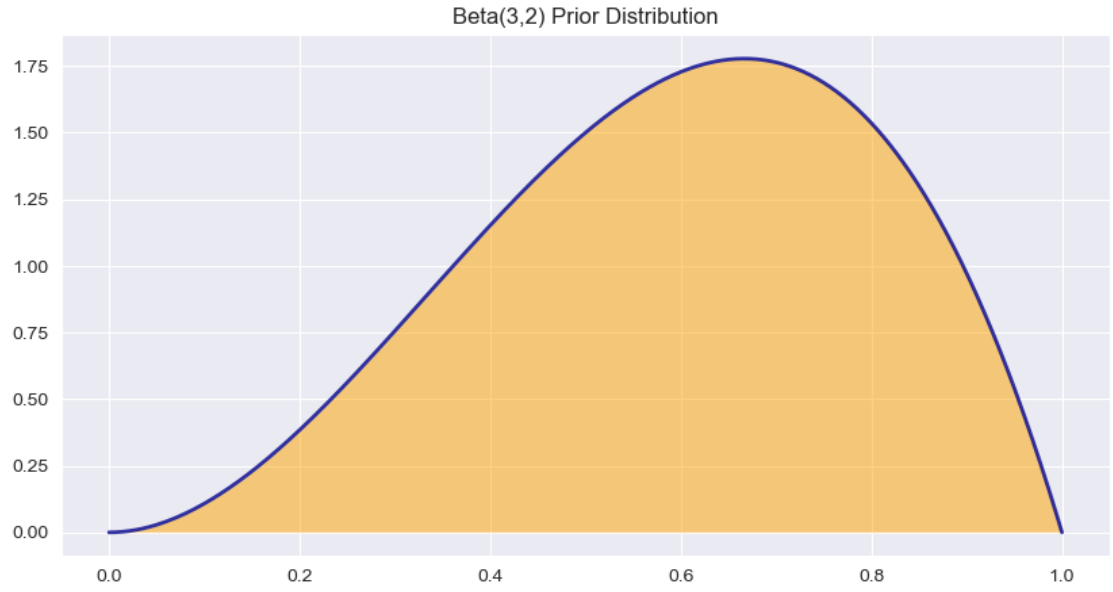
[53]:
```
plot_beta_prior(a=a,b=b)
```



[54]:
```
## Set the "true" probability
theta = 0.5 ## i.e. a fair coin

## Simulate a single flip of the coin
flip = np.random.binomial(n=1,p=theta,size=1)


## Set up counting variables
N1 = 0
N0 = 0
```

[55]:
```
## Update the prior to get the posterior using Bayes' Rule
N1 += flip[0]
N0 += 1 - flip[0]
a_post = N1 + a
b_post = N0 + b
```

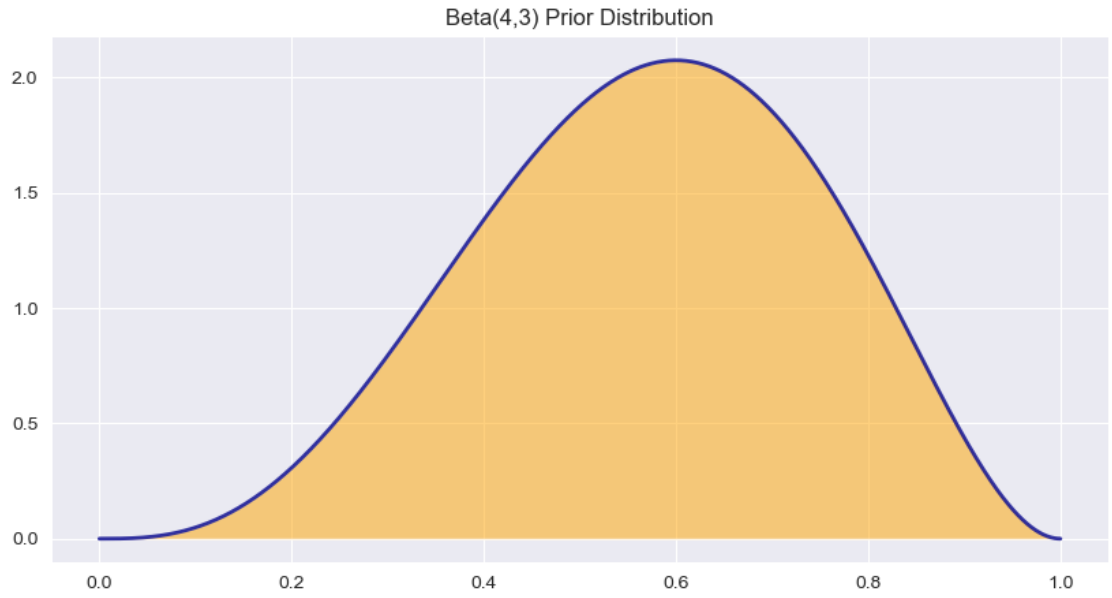
[56]:
```
plot_beta_prior(a=a_post,b=b_post)
```



[57]:
```
## Use the recent posterior as the new prior and repeat the process
a = a_post
b = b_post
flip = np.random.binomial(n=1,p=theta,size=1)
```

[58]:
```
## Update the prior to get the posterior using Bayes' Rule
N1 += flip[0]
N0 += 1 - flip[0]
a_post = N1 + a
b_post = N0 + b
```

[59]:
```
plot_beta_prior(a=a_post,b=b_post)
```



And so on, and so on.

Here is a question to think about: *if we are flipping a biased coin (say $\theta = 0.45$), which analyst will discover it first? The Bayesian, or the Frequentist?*

We will discuss this example again later and show the comparison.

Here is another question to think about: *if two Bayesians start with different prior beliefs (say the first with $B(\theta|1,1)$ and the second with $B(\theta|8,8)$) how long will it take them to converge in their beliefs? Will they for sure converge?*

Again, we can talk about this some more later.

One final note. The Bayesian does not have to proceed sequentially. He can process a batch of data all at once. Let's see this case, again starting with the prior $B(\theta|2,2)$.

[60]:
```
## Set the prior
a, b = 2, 2
```

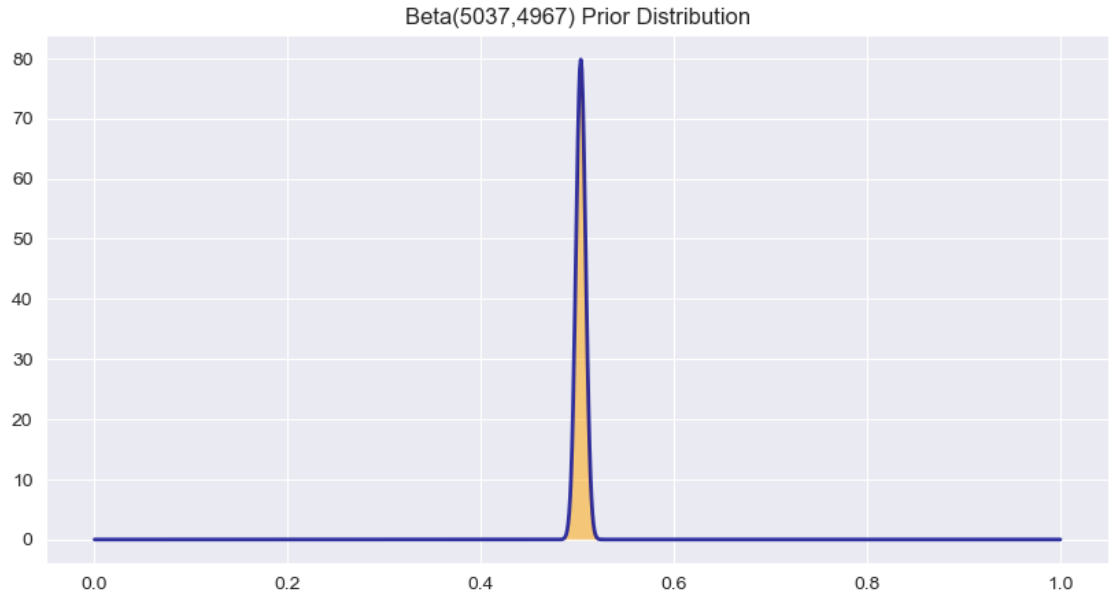
[61]:
```
## Set true theta
theta = 0.5
```

[62]:
```
## The number of coin flips
M = 10000

## Simulate flipping the coin M times
flips = np.random.binomial(n=1, p=theta, size=M)
```

[63]:
```
## The update step
N1 = np.sum(flips)
```

```
N0 = M - N1
a_post = N1 + a
b_post = N0 + b
```

[64]:
```
plot_beta_prior(a=a_post, b=b_post)
```



[65]:
```
posterior_mean = a_post / (a_post + b_post)
print(f"\nThe posterior mean is: {posterior_mean : 0.4f}\n")
```

```
The posterior mean is:  0.5035
```

[66]:
```
np.isclose(posterior_mean, theta)
```

[66]:
```
False
```

[67]:
```
print(f"\nThe MSE: {np.sqrt((theta - posterior_mean)**2.0) : 0.6f}\n")
```

```
The MSE:  0.003499
```

Note: with $N = 10000$ flips we become pretty certain after seeing the data.

Also, if we are dealing with a biased coin we will also learn that. Say the true probability of heads is $\theta = 0.65$ and that we flip the coin $N = 100$ times.

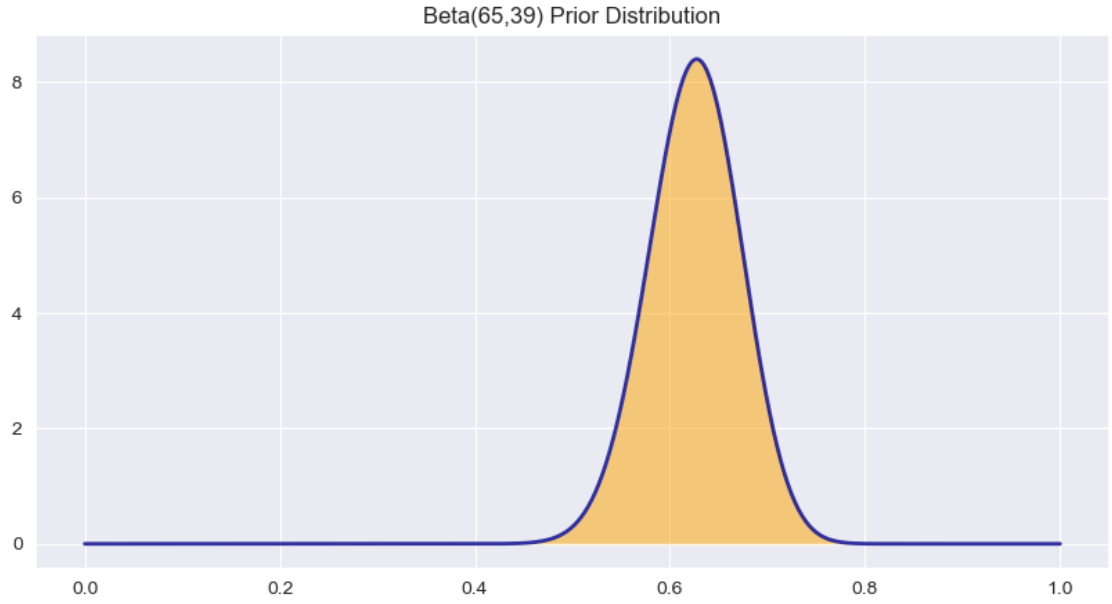
[68]:
```
## Set the prior
a, b = 2, 2
```

[69]:
```
## Set true theta
theta = 0.65
```

[70]:
```
## The number of coin flips
M = 100

## Simulate flipping the coin M times
flips = np.random.binomial(n=1, p=theta, size=M)
```

[71]:
```
## The update step
N1 = np.sum(flips)
N0 = M - N1
a_post = N1 + a
b_post = N0 + b
```

[72]:
```
plot_beta_prior(a=a_post, b=b_post)
```



[73]:
```
posterior_mean = a_post / (a_post + b_post)
print(f"\nThe posterior mean is: {posterior_mean : 0.4f}\n")
```

```
The posterior mean is:  0.6250
```

[74]:
```
np.isclose(posterior_mean, theta)
```

[74]: False

[75]:
```
print(f"\nThe MSE: {np.sqrt((theta - posterior_mean)**2.0) : 0.6f}\n")
```

```
The MSE:  0.025000
```