6: Inferring a binomial probability via exact mathematical analysis

The likelihood function: bernoulli distribution

The below equation has two meanings:

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

- The probability distribution (Bernoulli) over the two discrete values of γ for any fixed value of θ.
- The likelihood function of θ; think of γ as fixed and θ as a variable.
  - Specifies a probability at each value of θ but is not a probability distribution, and does not integrate to 1.

Usually thought of as the *Bernoulli likeliood function.*

A description of credibilities: the beta distribution

- Need the prior probability for each value of θ on the interval [0,1]
- In principle could use any probability density function on the interval.
- When applying Bayes rule there are two desiderata for mathematical tractability:
  - convenient if p($\gamma|\theta$)p(θ) has the same form as p(θ)
  - the denominator of Bayes' rule $\int d\theta p(\gamma|\theta)p(\theta)$ is solvable analytically
- If these are satisfied then the prior is a conjugate prior.
- In our example, a function of form $\theta^a(1-\theta)^b$ would satisfy these requirements, this is a beta distribution when it also includes normalizing constant $B(a,b)$.

Can think of a and b in the prior as previously observed flips of a heads and b tails.

Beta distribution properties:

- mode: $\omega = (a-1)/(a+b-2)$
- mean: $\mu = a/(a_b)$
- concentration: $\kappa = a+b$
- can specify the mode and concentration for example and thereby find appropriate a and b values for a prior.
- alternatively could specify the mean and the standard deviation.

The posterior data

Actual mathematical derivation of posterior distribution given likelihood of z heads with N flips and a prior as a function of a and b. The denominator (normalizing factor) sorts itself out because the result *must be* a probability distribution and the numerator ends up being that of a beta distribution so the denominator *must be* that of the corresponding beta distribution.

End result: $beta(\theta|a,b) -> beta(\theta|z+a, N-z+b)$

Very simple and intuitive.

6.3.1 Posterior is a compromise of prior and likelihood

The posterior mean is literally a weighted average of the prior mean and the mean of the data.

6.5 Summary

Two significant limitations:

- Only simple likelihood functions have conjugate priots
- Even if a conjugate prior exists, not all prior knowledge can be expressed in the mathematical form of the conjugate prior.

Therefore in complex applications we abandon exact mathematical solutions in favour of Markov Chain Monte Carlo (MCMC) methods.

Exercise 6.2

```
library("HDInterval")
prior = c(1,1)
data = c(58,42)
x <- seq(0,1,length=100001)
posterior_distribution = rbeta(x,1+58,1+42)
hdi = hdi(posterior_distribution)
hdi
```

```
##     lower     upper
## 0.4824259 0.6724702
## attr(,"credMass")
## [1] 0.95
```

```
posterior_distribution_2 = rbeta(x,1+58+57,1+42+43)
hdi2 = hdi(posterior_distribution_2)
hdi2
```

```
##     lower     upper
## 0.5036302 0.6395210
## attr(,"credMass")
## [1] 0.95
```

Exercise 6.3

```
x <- seq(0,1,length=100001)
posterior_distribution = rbeta(x,1+40,1+10)
hdi = hdi(posterior_distribution)
hdi
```

```
##     lower     upper
## 0.6781140 0.8934192
## attr(,"credMass")
## [1] 0.95
```

```
x <- seq(0,1,length=100001)
posterior_distribution = rbeta(x,1+15,1+35)
hdi = hdi(posterior_distribution)
hdi
```

```
##     lower     upper
## 0.1838215 0.4305466
## attr(,"credMass")
## [1] 0.95
```

Exercise 6.5

$$p(\gamma = 1|\theta) = \theta$$

```
# setwd("~/Data Science/Portfolio/doing_bayesian_data_analysis/DBDA2Eprograms")
# source("DBDA2E-utilities.R") # Load definitions of graphics functions etc.
# source("BernBeta.R") # Load the definition of the BernBeta function
# openGraph()
# post = BernBeta( priorBetaAB=c(1,1)*500 , Data=c(rep(1,9),rep(0,1)) ,
#  showHDI=TRUE , showCentTend="Mean" )

x <- seq(0,1,length=100001)
posterior_distribution = dbeta(x,500+9,500+1)
post = data.frame(x,posterior_distribution)
post$posterior_distribution = post$posterior_distribution / sum(post$posterior_distribution)
sum(post$x*post$posterior_distribution)
```

```
## [1] 0.5039604
```

```
x <- seq(0,1,length=100001)
posterior_distribution = dbeta(x,0.01+9,0.01+1)
post = data.frame(x,posterior_distribution)
post$posterior_distribution = post$posterior_distribution / sum(post$posterior_distribution)
sum(post$x*post$posterior_distribution)
```

```
## [1] 0.8991975
```