BST 140.651 Problem Set 3

Kai Aragaki

10/2/2020

Problem 1

---

a

The likelihood of a parameter requires observations as well as some underlying probability density/mass function, but we do NOT know the parameters that define that function.

If we are given the results of many coin flips, we do not necessarily know if the coin is fair or not - more formally, we do not know the probability parameter of this pmf that describes coin flips. When we are given the results of many coin flips, but *not* given the order in which they occurred (or we really don't care about order), we can model the pmf as a binomial distribution. This distribution is much like a Bernoulli distribution, however it includes a factorial term - known as a binomial coefficient - that accounts for the fact that, for example, getting one head in three flips of a coin could be either "HTT", "THT" or "TTH".

A likelihood is simply the value of the pmf at a given realization of the random variable. However, here we are not given the probability of getting a head (or the 'fairness' of the coin). If we consider all values of θ (where θ is the probability of flipping a head), the maximum likelihood is given by the 'optimal' value of θ.

A simple way to solve for the maximum likelihood is to take the derivative with respect to θ, set it equal to 0, then solve for θ.

For n flips, the likelihood function is

n_H = number of heads

$$\mathcal{L}(\theta|x) = \frac{n!}{n_H!(n-n_H)!}\theta^{n_H}(1-\theta)^{n-n_H}$$

Where $x \in \{0,1\}$ (where $0 = Tails$ and $1 = Heads$), and θ is the probability of getting a head.

So to find the maximum likelihood, we can take the derivative of thus function, set it equal to 0, and solve for θ.

$$\begin{aligned}&\frac{n!}{n_H!(n-n_H)!}\Big(\theta^{n_H}(1-\theta)^{n-n_H}\Big)' \\ =&\frac{n!}{n_H!(n-n_H)!}\Big(n_H\theta^{n_H-1}(1-\theta)^{n-n_H} - (n-n_H)(1-\theta)^{n-n_H-1}\theta^{n_H}\Big)\end{aligned}$$

Setting this to 0 and simplifying yields

$$\begin{aligned}
\frac{n!}{n_H!(n-n_H)!}(n_H\theta^{n_H-1}(1-\theta)^{n-n_H} - (n-n_H)(1-\theta)^{n-n_H-1}\theta^{n_H}) &= 0\\
n_H\theta^{n_H-1}(1-\theta)^{n-n_H} - (n-n_H)(1-\theta)^{n-n_H-1}\theta^{n_H} &= 0\\
n_H\theta^{n_H-1}(1-\theta)^{n-n_H} &= (n-n_H)(1-\theta)^{n-n_H-1}\theta^{n_H}\\
n_H(1-\theta) &= (n-n_H)\theta\\
n_H - n_H\theta &= (n-n_H)\theta\\
n_H &= (n)\theta\\
\theta &= \frac{n_H}{n}
\end{aligned}$$

Since we know for these trials that the number of heads was 7 and the number of trials was 10,

$\theta = 0.7$

so

$$\mathcal{L}(0.7|x) = \frac{10!}{7!3!}0.7^7(0.3)^3$$

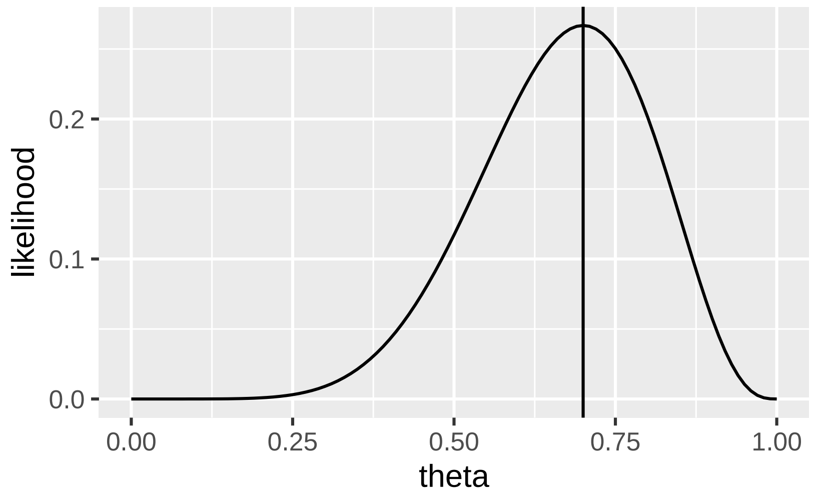
```
(factorial(10)/(factorial(7) * factorial(3))) * 0.7^7 * 0.3^3
```

```
## [1] 0.2668279
```

b

We can show this by plotting:

```
theta <- seq(0, 1, by = 0.01)
likelihood <- factorial(10)/(factorial(7)*factorial(3))*theta^7*(1-theta)^(3)
tibble(theta, likelihood) %>%
        ggplot(aes(theta, likelihood)) +
        geom_line() +
        geom_vline(xintercept = 0.7)
```



To give us a better idea of what this means in more comparable terms, we can renormalize it (ie multiply the function by some value) such that the maximum is equal to 1. To do this, we first find the current maximum likelihood at the given theta:

```
ml <- factorial(10)/(factorial(7) * factorial(3)) * 0.7^7 * (1 - 0.7)^(3)
ml
```

```
## [1] 0.2668279
```

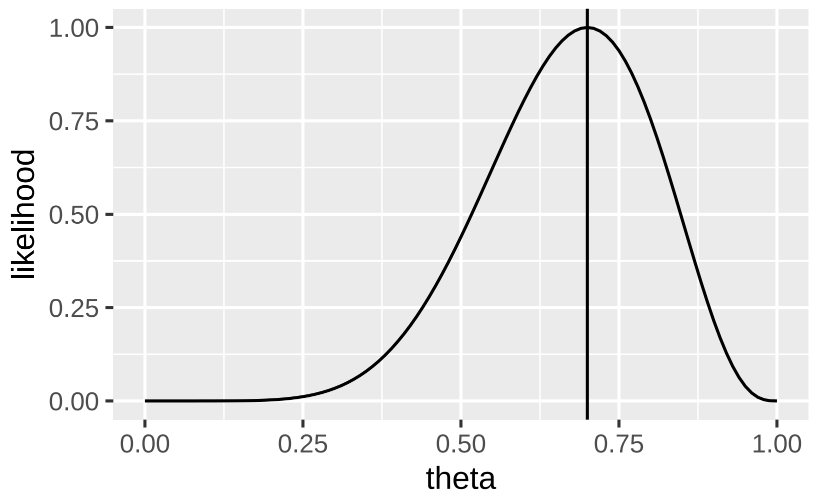
And then determine what number we need to multiply that value by to get 1:

```
scalar <- 1/ml
scalar
```

```
## [1] 3.747734
```

So now we can replot with this scalar out front:

```
theta <- seq(0, 1, by = 0.01)
likelihood <- scalar*factorial(10)/(factorial(7)*factorial(3))*theta^7*(1-theta)^(3)
dat <- tibble(theta, likelihood)
ggplot(dat, aes(theta, likelihood)) +
        geom_line() +
        geom_vline(xintercept = 0.7)
```



We note that the likelihood of $\theta = 0.5$ is roughly around 0.45. This means that it's about half as likely that $\theta = 0.5$ than $\theta = 0.7$. Again, it's not out of the realm of possibility.

c

What is the probability of seeing 7 or more heads if the coin was fair? We can calculate this by adding up the probability of seeing 7, 8, 9, and 10 heads individually:

```
running_sum <- 0
for (i in 7:10) {
    running_sum <- running_sum + factorial(10)/(factorial(i) * factorial(10 - i)) *
        0.5^i * (1 - 0.5)^(10 - i)
}
running_sum
```

```
## [1] 0.171875
```

The probability of getting 7 or more heads if this was a fair coin is ~17% - quite reasonable! This coin may be fair after all.

d

If we assume the trials were performed differently (in this case, flipping until k failures) then it would be more appropriate to model the pmf as the negative binomial distribution.

$$\binom{n-1}{2}(1-\theta)^k\theta^{n-k}$$

Since in this instance we set the number of failures to be 3, and the number of trials (n) is 10:

$$\binom{9}{2}(1-\theta)^3\theta^7 = \frac{9!}{2!7!}(1-\theta)^3\theta^7$$

Here, to calculate the maximum likelihood we take the derivative of the pmf with respect to θ once more, set it to 0, and solve for θ. However, we note that while the constant at the beginning has changed, the rest has not. This will give us the same θ (ie 0.7). We can plug this value in to get the maximum likelihood:

```
ml <- factorial(9)/(factorial(2) * factorial(7)) * 0.3^3 * 0.7^7
ml
```

```
## [1] 0.08004838
```

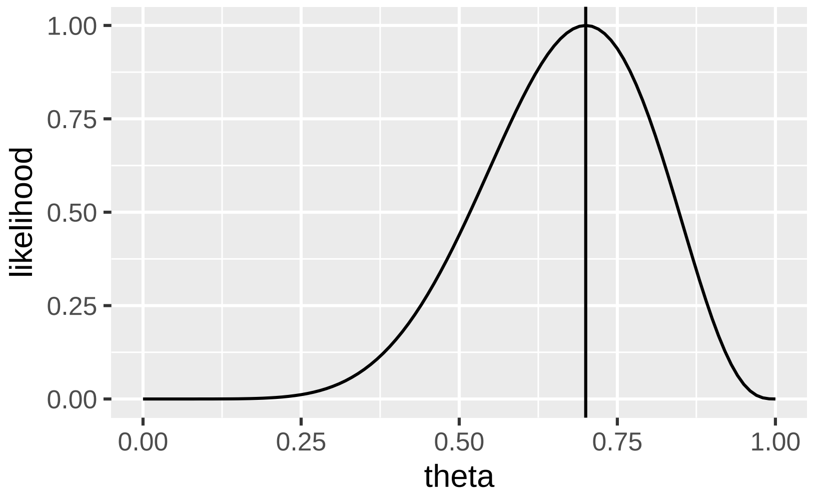
d

To bring the likelihood distribution to a common scale, we renormalize such that the maximum likelihood is 1:

```
scalar <- 1/ml
scalar
```

```
## [1] 12.49245
```

```
theta <- seq(0, 1, by = 0.01)
likelihood <- scalar*factorial(9)/(factorial(2)*factorial(7))*theta^7*(1-theta)^(3)
dat <- tibble(theta, likelihood)
ggplot(dat, aes(theta, likelihood)) +
        geom_line() +
        geom_vline(xintercept = 0.7)
```



We see that this likelihood (and the comparison to the likelihood at $\theta = 0.7$) is identical to that of the previous distribution.

To calculate the probability under the assumption of a new mass function, we again add probabilities - this time for 10 trials, 11 trials, etc...

```
running_sum <- 0
for (i in 10:50){
        running_sum <- running_sum + factorial(i-1)/(factorial(2)*factorial(i-3))*0.5^(i-3)*0.5^3
}
running_sum
```

```
## [1] 0.08984375
```

It becomes increasingly computationally difficult to find the solution at higher trials, but the additional terms become smaller and smaller. We could also simulate this:

```
vec <- vector(length = 100)
for (i in 1:100) {
    vec[i] <- (10000 - sum(table(rnbinom(10000, 3, 0.5))[1:7]))/10000
}

mean(vec)
```

```
## [1] 0.089874
```

Which is very similar to the value that we found.

We note that this probability is lower that in the case where we set out to flip 10 times and then tallied up how many tails we have.