The Bias Coin Reloaded

Assignment- 4

Suprio Dubey

2013036

09/12/2022

Question

Suppose that we have a coin, and we would like to figure out what the probability is that it will flip up heads with frequency f.
How should we estimate the bias f?
Q1) Be an orthodox Sampling theorist i.e. a Frequentist] The Binomial distribution is a suitable likelihood function for this problem. Derive an estimator of the bias maximizing the likelihood function (Maximum Likelihood method).
a) Generate N=100 coin flips with an input bias f of your choice. What is the estimated value of f and error bars?
b) Given N=5 toss and 5 heads as an outcome, what is the estimated value of f and error bars?
c) Given the condition in a and then b, try hypothesis testing with H0 - the coin is not biased vs H1 - the coin is biased
Q2) Be a Sampling theorist that makes use of the Bayes rule to generate an estimator] The Binomial distribution is a suitable likelihood function for this problem. Use as "conjugate prior" the Beta Distribution. Derive an estimator of the bias maximizing the posterior function (Maximum A Posteriori method).
a) Generate N=100 coin flips with an input bias f of your choice. What is the estimated value of f and error bars????
b) Given N=5 toss and 5 heads as the outcome what is the estimated value of f and error bars?
c) Model comparison is given conditions a and b.
d) What happens to the MAP estimator when N -¿ infinity
Q3) N data points $\{x_n\}$ are drawn from N distributions, all of which are Gaussian with a common mean μ but with different unknown standard deviation σ_n. What are the maximum likelihood parameters μ, $\{\sigma_n\}$ given the data? For example,

Scientist	x_n
A	-27.20
B	3.750
C	8.191
D	9.898
E	9.603
F	9.945
G	10.056

Table 1: Seven measurements x_n of a parameter μ by seven scientists each having his own noise-level σ_n .

seven scientists (A, B, C, D, E, F, G) with wildly-differing experimental skills measure μ. You expect some of them to do accurate work (i.e., to have small σ_n) and some of them to turn in wildly inaccurate answers (i.e., to have enormous σ_n). The table here shows their seven results. What is μ and how reliable is each scientist? I hope you agree that, intuitively, it looks pretty certain that A and B are both inept measures, that D-G is better and that the true value of μ is somewhere close to 10. But what does maximizing the likelihood tell you?

Answer1

The bias is denoted by f.The likelihood function is defined as: $\mathcal{P}(data|f, I)$.If in the conditioning information I, we assume that the flips of the coin were independent events so that the outcome of one did not influence that of another, then the probability of obtaining the data 'R heads in N tosses' is given by the binomial distribution:

$$\mathcal{P}(data|f, I) \propto f^n * (1-f)^{N-n} \tag{1}$$

f is the chance of obtaining a head on any flip, and there were n of them, and $1-f$ is the corresponding probability for a tail, of which there was $N-n$.
Taking the natural logarithm we get

$$L = nln(f) + (N-n)ln(1-f). \tag{2}$$

Now, to estimate f and its error bar. we need both the first and the second derivative of L w.r.t f.

$$\frac{dL}{df} = \frac{n}{f} - \frac{N-n}{1-f} \tag{3}$$

$$\frac{d^2L}{df^2} = -\frac{n}{f^2} - \frac{N-n}{(1-f)^2} \tag{4}$$

For the optimal value for the bias weighting by putting the first derivative to 0, we get:

$$\frac{dL}{df}_{|f_0} = \frac{n}{f_0} - \frac{N-n}{1-f_0} = 0 \tag{5}$$

So, we get:

$$f_0 = n/N \tag{6}$$

The best estimate of the bias weighting is the relative frequency of outcomes of heads.
Now the error is related to the second derivative for a Gaussian distribution as:

$$\sigma = (-\frac{d^2L}{df^2}_{|f_0})^{-\frac{1}{2}} \tag{7}$$

So, the error bar is evaluated at $f = f_0$ on substituting $n = f_0/N$ in eq. (4) the second derivative of the log-likelihood we get:

$$\frac{d^2L}{df^2} = -\frac{N}{f_0(1-f_0)} \tag{8}$$

Now, substituting the second derivative in eq. (7) to find the error in the bias weighting we get

$$\sigma_f = \sqrt{\frac{f_0(1-f_0)}{N}} = \frac{\sigma^2}{N^2} \tag{9}$$

where σ^2 is the variance for the binomial distribution.

Now, lets us solve our given problem.

a. Simulating 100 coin flips

Considering the input bias as 1/3. We see the estimated value of f is 0.350000 and the error is 0.047697 $thecodeofthesimulationismentionedinAppendix$

b. Simulating 5 tosses with 5 heads as an outcome

Considering the input bias as 1 since we are simulating 5 tosses with 5 heads as an outcome. We see the estimated value of f is 1.000000 and the error is extremely high $thecodeofthesimulationismentionedinAppendix$

c. Hypothesis Testing of a. and b.

Hypothesis testing is to compare the two hypotheses:
1) H_0 = Coin is not biased
2) H_1 = The coin is biased
This model comparison can be simply converted into "is $f = 1/2$ or not"? In a frequentist analysis, we need a tool to understand how our repeated measurement results are consistent with one particular model. We need to understand how much the obtained $\hat{f}$ is different from the expected f_i, taking into account the errorbar σ_f, too. We can use for this the following quantity z, given by

$$z = \frac{|f - f_i|}{\sigma_{f_i}} \,. \tag{10}$$

We can see here that z can be useful for our question because we can compare the measured $\hat{f}$ with the expected theoretical f_i, normalizing this by σ_{f_i}. We can approach this problem by simply substituting $f_0 = 1/2$. If $z \leq 2$, the measured f is inside $2\sigma_{1/2}$ from 1/2, so we can assume that H_0 is valid. If instead $z > 2$, H_0 is not a good model, and since H_0 and H_1 are complementary, we can deduce that H_1 is valid. In both cases, we can proceed to show the z value when N increases. We get the following results: $f_i = 0.5$ is 0.510000 and the error is 0.002499
Case1- input bias(0.33)
The estimated value of f is 0.350000.

Case2- input bias(1)
the estimated value of f is 1.000000
Z for the Case 1 = 3.200640192064023
Z for the Case 1 = 9.801960588196067

ANSWER 2

In question 2 the target is to find the posterior probability for the bias f given n heads in N number of flips. T Using the Bayes theorem we get:

$$\mathcal{P}(f|n,N) = \frac{\mathcal{P}(n|f,N)\,\mathcal{P}(f)}{\mathcal{P}(n|N)}\,. \tag{11}$$

Where the likelihood is:

$$\mathcal{P}(n|f,N) \propto f^n(1-f)^{N-n}\,. \tag{12}$$

The prior probability for f considering the Beta Distribution, given by:

$$\mathcal{P}(f|\alpha) = \frac{f^{\alpha-1}\,(1-f)^{\alpha-1}}{B(\alpha)}\,, \tag{13}$$

where $B(\alpha)$ is the Beta Function, a normalization constant given by

$$B(\alpha) = \int_0^1 x^{\alpha-1}(1-x)^{\alpha-1}dx$$

(14)

which can also be written in form of Gamma Function as:

$$B(\alpha) = \frac{\Gamma(\alpha)}{\Gamma(2\alpha)} \tag{15}$$

The shape of this distribution depends on the value of the parameter α, but it is essentially a curve that has its peak at $f = 0.5$: as α increases from 1 to infinity, the peak starts to get narrower. For $\alpha = 1$ the prior will be uniform.
Now. we can write the posterior as:

$$\mathcal{P}(f|n,N) \propto \mathcal{P}(n|f,N)\,\mathcal{P}(f) \propto f^{n+\alpha-1}(1-f)^{N-n+\alpha-1} \tag{16}$$

The posterior is a Binomial distribution with $n' = n + \alpha - 1$ and $N' = N + 2\alpha - 2$. Let us maximize this posterior by putting the first derivative to 0, we get:

$$f_0 = \frac{n'}{N'} = \frac{n + \alpha - 1}{N + 2\alpha - 2}\,. \tag{17}$$

and its error bar will be :

$$\sigma_f = \sqrt{\frac{f_0(1 - f_0)}{N'}} = \sqrt{\frac{f_0(1 - f_0)}{N + 2\alpha - 2}} == \frac{\sigma^2}{N'^2} \tag{18}$$

where σ^2 is the variance for the binomial distribution.

a. Simulating 100 coin flips

The input bias chosen was $f = \frac{1}{3}$
Case1- input bias(0.33)
For the given values of $\alpha = (1, 21, 41, 61, 81)$
the estimated value of f is 0.350000 and the error is 0.047697
the estimated value of f is 0.392857 and the error is 0.041276
the estimated value of f is 0.416667 and the error is 0.036747
the estimated value of f is 0.431818 and the error is 0.033395
the estimated value of f is 0.442308 and the error is 0.030802

b. Simulating 5 tosses with 5 heads as an outcome

The input bias chosen was $f = 1$
Case2- input bias(1)
For the given values of $\alpha = (1, 21, 41, 61, 81)$
the estimated value of f is 1.000000 and the error is 0.000000
the estimated value of f is 0.555556 and the error is 0.074074
the estimated value of f is 0.529412 and the error is 0.054139
the estimated value of f is 0.520000 and the error is 0.044686
the estimated value of f is 0.515152 and the error is 0.038907

c. Model comparison is given conditions a and b

We

In this case, we have the posterior of the models. We can use the Bayes factor i.e calculate the relative goodness between the probability of having model H_0

(null hypothesis) and model H_1.

$$B_{01} = \frac{\mathcal{P}(H_0|n, N)}{\mathcal{P}(H_1|n, N)}\,, \tag{19}$$

where the number of heads n represents data. Using for both the numerator and the denominator the Bayes theorem, we have

$$\begin{aligned} B_{01} &= \frac{\mathcal{P}(n|H_0, N)\,\mathcal{P}(H_0|N)}{\mathcal{P}(n|N)} \frac{\mathcal{P}(n|N)}{\mathcal{P}(n|H_1, N)\,\mathcal{P}(H_1|N)} \\ &= \frac{\mathcal{P}(n|H_0, N)}{\mathcal{P}(n|H_1, N)} \frac{\mathcal{P}(H_0|N)}{\mathcal{P}(H_1|N)}\,. \end{aligned} \tag{20}$$

Considering a uniform prior for H_0 and H_1, that means

$$\mathcal{P}(H_0|N) = \mathcal{P}(H_1|N) = \frac{1}{2}\,, \tag{21}$$

we are left with

$$B_{01} = \frac{\mathcal{P}(n|H_0, N)}{\mathcal{P}(n|H_1, N)}\,. \tag{22}$$

For both these likelihoods, we have to marginalize over all possible f values. We can derive a general marginalization for the model H_i :

$$\mathcal{P}(n|H_i, N) = \int_0^1 df\, \mathcal{P}(n|H_i, f, N)\, \mathcal{P}(f|H_i, N)\,, \tag{23}$$

were $\mathcal{P}(n|H_i, f, N)$ is the Binomial likelihood

$$\mathcal{P}(n|H_i, f, N) = \binom{N}{n} f^n (1-f)^{N-n}\,. \tag{24}$$

Analyzing them separately.
CASE-1: H_0,
$f = 0.5$, so we can say for the prior probability that

$$\mathcal{P}(f|H_0, N) = \delta\Big(f - 0.5\Big)\,. \tag{25}$$

Therefore we have

$$\begin{aligned} \mathcal{P}(n|H_0, N) &= \int_0^1 df\, \mathcal{P}(n|H_0, f, N)\, \delta\Big(f - \frac{1}{2}\Big) \\ &= \int_0^1 df\, \binom{N}{n} f^n\, (1-f)^{N-n}\, \delta\Big(f - \frac{1}{2}\Big)\,. \\ &= \binom{N}{n} \frac{1}{2^N} \end{aligned} \tag{26}$$

CASE-2: H_1,
We have to consider the Beta prior chosen

$$\begin{aligned}\mathcal{P}(n|H_1, N) &= \int_0^1 df \binom{N}{n} f^n (1-f)^{N-n} \frac{f^{\alpha-1}(1-f)^{\alpha-1}}{B(\alpha)} \\ &= \binom{N}{n} \frac{1}{B(\alpha)} \int_0^1 df\, f^{n+\alpha-1}(1-f)^{N-n+\alpha-1} \\ &= \binom{N}{n} \frac{1}{B(\alpha)} \frac{(n+\alpha-1)!\,(N-n+\alpha-1)!}{(N+2\alpha-1)!}\,.\end{aligned} \tag{27}$$

Substituting in B_{01}, we have

$$\begin{aligned}B_{01} &= \frac{(N+2\alpha-1)!}{(n+\alpha-1)!\,(N-n+\alpha-1)!} \frac{B(\alpha)}{2^N} \\ &= \binom{N+2\alpha-2}{n+\alpha-1} \frac{B(\alpha)}{(N+2\alpha-2)\,2^N} \\ &= \binom{N'}{n'} \frac{B(\alpha)}{N'\,2^N}\end{aligned}\,. \tag{28}$$

Note that this depends only on N and on α, where for an uninformative prior ($\alpha = 1$) we have

$$B_{01,\,\alpha=1} = \binom{N}{n} \frac{1}{N\,2^N}\,. \tag{29}$$

On simulating we get $B_{01,\,\alpha=1} = 8.6385566574165\,25^6$ for N=100 and bias input $= 1/3$ $B_{01,\,\alpha=1} = 0.00625$ for N=5 and bias input = 1

d. What happens to the MAP estimator when $N \to +\infty$

We notice that the best bias estimator is:

$$f_0 = \frac{n'}{N'} = \frac{n+\alpha-1}{N+2\alpha-2} \tag{30}$$

so, if $N \to +\infty$ the value of n increases as well so the above equation:
$f_0 = \frac{n'}{N'} = \frac{n+\alpha-1}{N+2\alpha-2} \to \frac{n}{N}$

ANSWER 3

In question 3, we have a superposition of different Gaussian distributions with the same μ, but with different σ_n. We are looking for two estimators for both μ and

σ_n. The likelihood for our problem is a product of individual Gaussians:

$$\begin{aligned}\mathcal{P}(\{x_n\}|\mu,\{\sigma_n\}) = \prod_{n=1}^{N}\mathcal{P}(x_n|\mu,\sigma_n) \\ = \prod_{n=1}^{N}\frac{1}{\sqrt{2\pi\,\sigma_n^2}}\,e^{-(x_n-\mu)^2/2\sigma_n^2}\end{aligned}, \tag{31}$$

Considering that every Gaussian has the same prior probability. The log-likelihood L that we want to maximize is given by

$$\begin{aligned}L = \sum_{n=1}^{N}\ln\left[\frac{1}{\sqrt{2\pi\,\sigma_n^2}}\,e^{-(x_n-\mu)^2/2\sigma_n^2}\right] \\ = \sum_{n=1}^{N}\left[-\frac{1}{2}\,\ln 2\pi - \ln\sigma_n - \frac{(x_n-\mu)^2}{2\sigma_n^2}\right]\end{aligned}. \tag{32}$$

The estimator for σ_n(for n^{th} measure) is given by

$$\frac{\partial}{\partial\sigma_n}\left[-\frac{1}{2}\,\ln 2\pi - \ln\sigma_n - \frac{(x_n-\mu)^2}{2\sigma_n^2}\right] = -\frac{1}{\sigma_n} + \frac{(x-\mu)^2}{\sigma_n^3}\,. \tag{33}$$

Equating it to zero to find the maximum, we get:

$$\frac{\partial}{\partial\sigma_n}\left[-\frac{1}{2}\,\ln 2\pi - \ln\sigma_n - \frac{(x_n-\mu)^2}{2\sigma_n^2}\right] = 0 \implies \hat{\sigma}_n = |x_n-\mu|\,. \tag{34}$$

Now we have an estimator for σ_n depending on the corresponding n-result x_n. We have still to search for the μ value that maximises μ:

$$\frac{\partial Ł}{\partial\mu} = \sum_{n=1}^{N}\frac{\partial}{\partial\mu}\left[-\frac{1}{2}\,\ln 2\pi - \ln\sigma_n - \frac{(x_n-\mu)^2}{2\sigma_n^2}\right] = \sum_{n=1}^{N}\frac{x_n-\mu}{\sigma_n^2}\,. \tag{35}$$

Considering that for a fixed n we have just estimated the value for $\hat{\sigma}_n$ in eq (33), we can substitute and take the derivative equal to 0:

$$\frac{\partial Ł}{\partial\mu} = \sum_{n=1}^{N}\frac{1}{x_n-\mu} = 0\,. \tag{36}$$

This must be solved via the Newton-Raphson method, which is implemented below in the code of Answer 3. We can see that for $\mu = 3.57$ results are better, considering standard deviation, too.

Appendix

Answer 1a, 1b ,1c, 2a, 2b, 2c,3 Python Code for the simulation of the problem

```python
#!/usr/bin/python
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as st
import scipy.special as sp
from numpy import random as rand

def coin_toss(N,f):
    np.random.seed(1)
    toss = rand.uniform(low=0.0, high=1.0, size=N)
    head_frequency = np.sum(toss < f)

    return head_frequency

def likelihood(N,f):
    n =coin_toss(N,f)/N

    probability=st.binom.stats(N,n)

    return probability,n

def r(n,N,f):
    rm=st.binom.pmf(n,N,f)

    return rm
####Solution for answer 1.a#####

np.random.seed(1)
f=1/3.
N = 100
mean,var = likelihood(N,f)[0]
Mean= mean/N
var/=N**2
var=np.sqrt(var)
print('the estimated value of f is %f and the error is  %f'%(Mean,
    var))

x=np.arange(0,1,0.01)
plt.plot(x,r(mean,float(N),x))
plt.show()

#####Solution for answer 1.b#####

```

```
f =1.
N = 5
# $f$ will be 1 since we get 5 heads
mean,var = likelihood(N,f)[0]
Mean= mean/N
var/=N**2
print('the estimated value of f is %f and the error is  %f'%(Mean,
    var))
x=np.arange(0,1,0.01)
plt.plot(x,r(mean,float(N),x))
plt.show()


###1.c Now we will see the Hypothesis Testing####
f=0.5
N = 100
mean,var = likelihood(N,f)[0]
Mean_0= mean/N
var/=N**2
error_nonbias=np.sqrt(var)
print('the estimated value of f when input bias is 0.5 is %f and
    the error is  %f'%(Mean_0,var))


f=1/3.
N = 100
mean,var = likelihood(N,f)[0]
Mean_f1= mean/N
var/=N**2
var=np.sqrt(var)
print('the estimated value of f is %f and the error is  %f'%(
    Mean_f1,var))

f=1.
N = 5
mean,var = likelihood(N,f)[0]
Mean_f2= mean/N
var/=N**2
var=np.sqrt(var)
print('the estimated value of f is %f and the error is  %f'%(
    Mean_f2,var))

z1 = np.linalg.norm(Mean_f1-Mean_0)/error_nonbias
z2 = np.linalg.norm(Mean_f2-Mean_0)/error_nonbias
print(z1)
print(z2)


##### Answer 2
```

```

## In the second answer we saw that the posterior is a Binomial distribution with $n'=n+\alpha-1$ and$N'=N+2\alpha-2$, where $\alpha$ is a parameter of the Beta - distribution.###
def posterior(N,f,i):
    n =coin_toss(N,f)
    n = n+i-1
    N_new = N+2*(i-1)
    n_new = n/N_new
    p1,p2=st.binom.stats(N_new,n_new)

    return p1,p2,N_new
#### 2.a #########
for i in range(1,100,20):
    N=100
    f = 1/3.
    mean,var = posterior(N,f,i)[:2]
    N_new = (posterior(N,f,i)[2])
    Mean= mean/N_new
    var/=N_new**2
    var=np.sqrt(var)
    print('the estimated value of f is %f and the error is  %f'%(Mean,var))
    x=np.arange(0,1,0.01)
    plt.plot(x,r(mean,float(N_new),x))

plt.title('Posterior')
plt.xlabel('f')
plt.ylabel('PDF')
plt.show()

##### 2.b #########
for i in range(1,100,20):
    N=100
    f = 1/3.
    mean,var = posterior(N,f,i)[:2]
    N_new = (posterior(N,f,i)[2])
    Mean= mean/N_new
    var/=N_new**2
    var=np.sqrt(var)
    print('the estimated value of f is %f and the error is  %f'%(Mean,var))
    x=np.arange(0,1,0.01)
    plt.plot(x,r(mean,float(N_new),x))

plt.title('Posterior')
plt.xlabel('f')
plt.ylabel('PDF')
plt.show()
```

```

### model comparison ####
def Bayes_factor(N,f,i):
    n = coin_toss(N,f)
    n = n+i-1
    N_new = N+2*(i-1)
    b_factor = sp.comb(N_new,n)*sp.beta(i,i)/(N_new*2**N)
    return b_factor
print(Bayes_factor(100,1/3,1))
print(Bayes_factor(5,1,1))


########### Answer 3 ####################
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as st

def seven(x,mu,sigma):
  P=st.norm.pdf(x,loc=mu,scale=sigma)
  return P

def f(x,mu):
  r=0.
  for i in range(len(x)):
    r+=1./(mu-x[i])
  return r

def der(x,mu):
  r=0.
  for i in range(len(x)):
    r+=1./(x[i]-mu)**2.
  return r

eps=1e-6
x=np.array([-27.020,3.570,8.191,9.898,9.603,9.945,10.056])
mu=np.array([-25.,4.,8.50,9.7,9.92,10.])
for j in range(len(mu)):
  mu_old=30.
  while np.linalg.norm(mu[j]-mu_old)>=eps:
    mu_old=np.copy(mu[j])
    mu[j]-=f(x,mu[j])/der(x,mu[j])
print('mu =',mu)

sigma=np.zeros([len(mu),len(x)])
for i in range(len(mu)):
  for j in range(len(x)):
    sigma[i,j]=np.linalg.norm(x[j]-mu[i])

```

```
sigma2=np.copy(sigma)
sigma=pd.DataFrame(sigma, columns=[f'x = {x[0]}', f'x = {x[1]}', f'x = {x[2]}', f'x = {x[3]}', f'x = {x[4]}', f'x = {x[5]}', f'x = {x[6]}'], index=[f'mu = {mu[0]}', f'mu = {mu[1]}', f'mu = {mu[2]}', f'mu = {mu[3]}', f'mu = {mu[4]}', f'mu = {mu[5]}'])
xplot=np.arange(-30.,12.,0.1)

print('standard deviation:')
print(sigma)

Answer 3
mu = [-27.02     3.57     8.191    9.898    9.898    9.945]
standard deviation:
                              x = -27.02       x = 3.57   x = 8.191   x = 9.898  \
mu = -27.01999999999995  4.973799e-14  3.059000e+01     35.211   3.691800e+01
mu = 3.57000000000005    3.059000e+01  5.018208e-14      4.621   6.328000e+00
mu = 8.191               3.521100e+01  4.621000e+00      0.000   1.707000e+00
mu = 9.898000000000012   3.691800e+01  6.328000e+00      1.707   1.243450e-14
mu = 9.898000000003284   3.691800e+01  6.328000e+00      1.707   3.284484e-12
mu = 9.945               3.696500e+01  6.375000e+00      1.754   4.700000e-02

                            x = 9.603   x = 9.945   x = 10.056
mu = -27.01999999999995       36.623      36.965       37.076
mu = 3.57000000000005          6.033       6.375        6.486
mu = 8.191                     1.412       1.754        1.865
mu = 9.898000000000012         0.295       0.047        0.158
mu = 9.898000000003284         0.295       0.047        0.158
mu = 9.945                     0.342       0.000        0.111

for j in range(len(x)):
    plt.plot(xplot, fun(xplot,mu[1],sigma[f'x = {x[j]}'][f'mu = {mu[1]}']),label=f'$\sigma_{j}$')
plt.legend()
plt.scatter(x, np.zeros(len(x),float),label='data points')
plt.title(f'$\mu$ = {mu[1]}')
plt.xlabel('x')
plt.ylabel('PDF')
plt.show()
```