Assignment 5

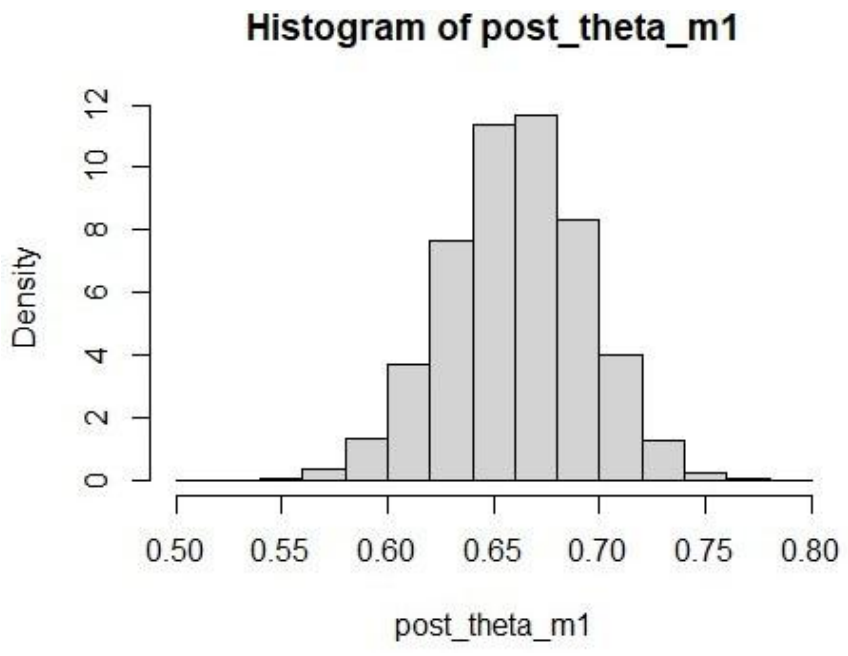
Name:Pritam Priyadarshi
Roll no.: 230793

Part 1: Information-theoretic measures and cross-validation

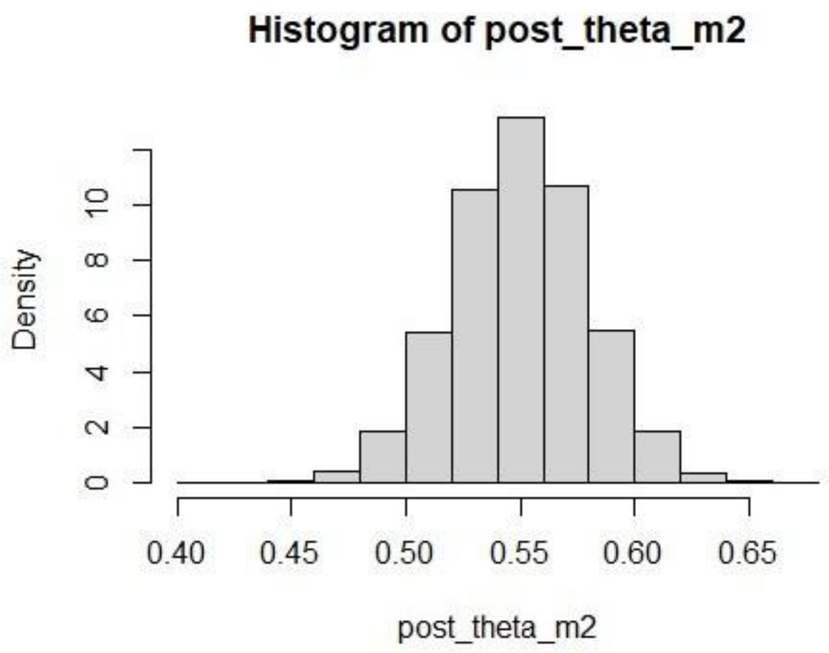
The observed data originate from a binomial distribution, with the parameter θ following a beta distribution. Consequently, using a binomial-beta prior, the posterior distribution of θ given the data is also a beta distribution, specifically θ|y ~ Beta(a + ∑_{i=1}^{N_{obs}} y_i, b + N_{obs} imes n - ∑_{i=1}^{N_{obs}} y_i).

thus directly,

```
y = c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
Nobs = 10
# for each model----- # for model 1
a = 6
b = 6
n = 20
post_theta_m1 = rbeta(100000,a+sum(y),b+(Nobs*n)-sum(y)) hist(post_theta_m1,freq = FALSE)
```



```r
# for model 2
a = 20
b = 60
n = 20
post_theta_m2 = rbeta(100000,a+sum(y),b+(Nobs*n)-sum(y)) hist(post_theta_m2,freq =
FALSE)
```



```r
# Compute log pointwise predictive density (lppd) for the models # for model 1
a = 6
b = 6
n = 20
lppd_m1 = 0
for(i in 1:Nobs)
{
sample_theta =rbeta(1000,a+sum(y),b+(Nobs*n)-sum(y))
lpd_i = log(mean(dbinom(y[i],size = 20,prob = sample_theta))) lppd_m1 = lppd_m1 +
lpd_i
}
lppd_m1
## [1] -20.41417
# for model 2
a = 20
b = 60
n = 20
lppd_m2 = 0
for(i in 1:Nobs)
{
sample_theta =rbeta(1000,a+sum(y),b+(Nobs*n)-sum(y))
lpd_i = log(mean(dbinom(y[i],size = 20,prob = sample_theta))) lppd_m2 = lppd_m2 +
lpd_i
}
lppd_m2
## [1] -25.89447
```

The in-sample deviance is calculated using the formula: -2 times the log posterior predictive density (lppd). A lower deviance indicates a better fit of the model to the data, reflecting improved predictive accuracy.

We refer to it as "*in-sample deviance*" because it measures how well the statistical model fits the observed data points within the sample used for training. Specifically, it assesses the model's goodness of fit based on the data it has been trained on.

```
# in-sample deviance for both the models----- # for model 1 :
deviance_m1 = -2*lppd_m1 deviance_m1
## [1] 40.82833
# for model 2 : deviance_m2 = -2*lppd_m2 deviance_m2
## [1] 51.78895
```

Considering the in-sample deviance, model 1 demonstrates a lower value, indicating a better fit to the data compared to model 2.

```
y_new = c(5, 6, 10, 8, 9) Nobs = length(y_new)
# calculating lppd of the both models using the new data and then compairing. #
for model 1
a = 6
b = 6
n = 20
lppd_m1 = 0
for(i in 1:Nobs)
{
sample_theta =rbeta(1000,a+sum(y_new),b+(Nobs*n)-sum(y_new)) lpd_i =
log(mean(dbinom(y_new[i],size = 20,prob = sample_theta))) lppd_m1 = lppd_m1 + lpd_i
}
lppd_m1
## [1] -10.4823
out_of_sample_deviance_m1 = -2*lppd_m1 out_of_sample_deviance_m1
## [1] 20.96461
# for model 2
a = 20
b = 60
n = 20
lppd_m2 = 0
for(i in 1:Nobs)
{
sample_theta =rbeta(1000,a+sum(y_new),b+(Nobs*n)-sum(y_new)) lpd_i =
log(mean(dbinom(y_new[i],size = 20,prob = sample_theta))) lppd_m2 = lppd_m2 + lpd_i
}
lppd_m2
## [1] -11.03218
out_of_sample_deviance_m2 = -2*lppd_m2 out_of_sample_deviance_m2
## [1] 22.06436
```

Model 1 demonstrates superior performance compared to Model 2, as evidenced by its lower out-of-sample deviance value.

```
y = c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
Nobs = 10

# using LOO-CV for both the models #for model 1 :
a = 6
b = 6
elpd_m1 = 0
for(i in 1:Nobs)
{
y_train = y[-i] y_test = y[i]
sample_theta =rbeta(1000,a+sum(y_train),b+(Nobs*n)-sum(y_train)) lpd_i =
log(mean(dbinom(y_test,size = 20,prob = sample_theta))) elpd_m1 = elpd_m1 + lpd_i
}
elpd_m1
## [1] -22.91949
#for model 1 :
a = 20
b = 60
elpd_m2 = 0
for(i in 1:Nobs)
{
y_train = y[-i] y_test = y[i]
sample_theta =rbeta(1000,a+sum(y_train),b+(Nobs*n)-sum(y_train)) lpd_i =
log(mean(dbinom(y_test,size = 20,prob = sample_theta))) elpd_m2 = elpd_m2 + lpd_i
}
elpd_m2
## [1] -31.52051
```

Using leave-one-out cross-validation, we can conclude that model 1, which has a higher expected log predictive density (elpd), outperforms model 2.

Part 2: Marginal likelihood and prior sensitivity

Consider a binomial model with a sample size of n and a success probability θ. The prior distribution for θ is Beta(a, b). The marginal likelihood of the model, given k successes, is expressed as nCk multiplied by the ratio of factorials: (k + a - 1)! times (n - k + b - 1)! divided by (n + a + b - 1)!.

```r
ML_binomial <- function(k,n,a,b){
ML <- (factorial(n)/(factorial(k)*factorial(n-k)))*( factorial(k+a-1)*factorial(n-k+b-1)/factorial(n+a+b-1)) ML}
```

– Beta(0.1, 0.4)

```r
a = 0.1
b = 0.4
ML_binomial(k=2,n = 10,a,b)
## [1] 0.4739564

```

– Beta(1, 1)

```r
a = 1
b = 1
ML_binomial(k=2,n = 10,a,b)
## [1] 0.09090909

```

– Beta(2, 6)

```r
a = 2
b = 6
ML_binomial(k=2,n = 10,a,b)
## [1] 0.004726891

```

– Beta(6,2)

```r
a = 6
b = 2
ML_binomial(k=2,n = 10,a,b)
## [1] 0.0002313863

```

– Beta(20, 60)

```r
a = 20
b = 60
ML_binomial(k=2,n = 10,a,b)
## [1] 5.079397e-21
```

– Beta(60, 20)

```r
a = 60
b = 20
ML_binomial(k=2,n = 10,a,b)
## [1] 1.50663e-23

```

Lets create a function that will estimate the Marginal Likelihood using the Monte-Carlo Integration Method.

```r
ML_MC = function(k,n,a,b)
{
lkl = c()
for(i in 1:10000)
{
theta_i = rbeta(1,a,b)
likelihood = dbinom(k,size = n,prob = theta_i) lkl = c(lkl,likelihood)
}
mean(lkl)
}
```

Now calculating ML using Monte Carlo integration method for each of them-

– Beta(0.1, 0.4)

```r
a = 0.1
b = 0.4
ML_MC(k=2,n = 10,a,b)
## [1] 0.03908727

```

– Beta(1, 1)

```r
a = 1
b = 1
ML_MC(k=2,n = 10,a,b)
## [1] 0.09003135

```

– Beta(2, 6)

```r
a = 2
b = 6
ML_MC(k=2,n = 10,a,b)
## [1] 0.1991219

```

– *Beta(6, 2)*

```r
a = 6
b = 2
ML_MC(k=2,n = 10,a,b)
## [1] 0.009762455

```

– *Beta(20, 60)*

```r
a = 20
b = 60
ML_MC(k=2,n = 10,a,b)
## [1] 0.2695549

```

– *Beta(60, 20)*

```r
a = 60
b = 20
ML_MC(k=2,n = 10,a,b)
## [1] 0.0007859686

```