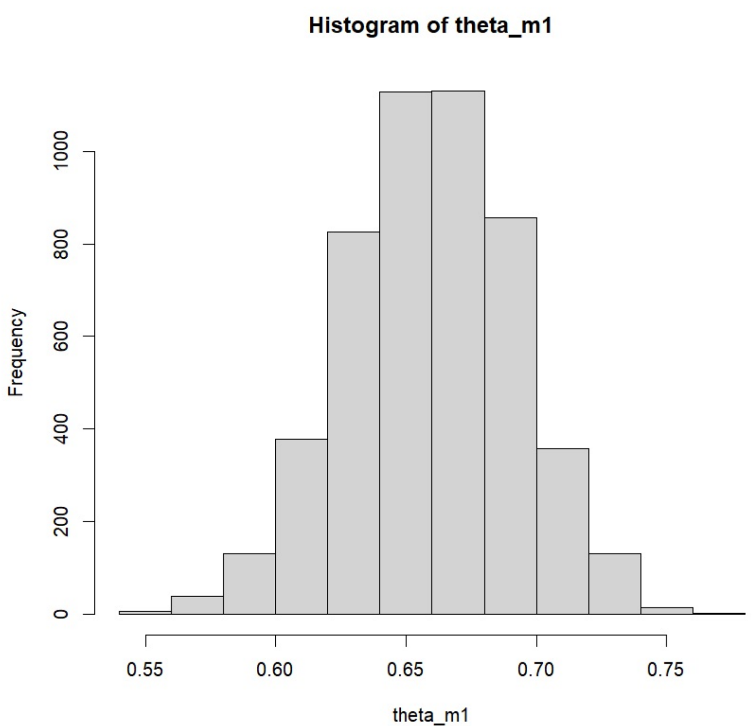
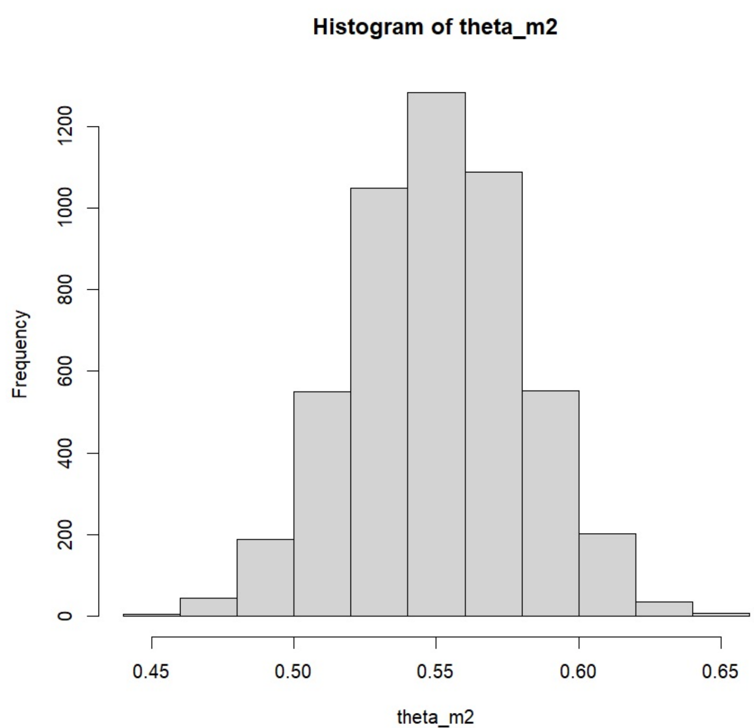
* Assignment - 5 *

Part 1: Information-theoretic measures and cross-validation

1.1
```
#Observed data
y <- c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
N_obs<-10
# Model1: Binom(20,theta), theta ~ beta(6,6)
theta_m1 <- rbeta(5000,6+sum(y),6+N_obs*20-sum(y))
hist(theta_m1)

# Model2: Binom(20,theta), theta ~ beta(20,60)
theta_m2 <- rbeta(5000,20+sum(y),60+N_obs*20-sum(y))
hist(theta_m2)
```





1.2

```r
# Observed data
y <- c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
N_obs <- 10

# Model 1
lppd_m1 <- 0
for(i in 1:N_obs){
  sample_theta <- rbeta(1000, 6 + sum(y), 6 + N_obs * 20 - sum(y))
  lpd_i <- log(mean(dbinom(y[i], 20, sample_theta)))
  lppd_m1 <- lppd_m1 + lpd_i
}

# Model 2
lppd_m2 <- 0
for(i in 1:N_obs){
  sample_theta <- rbeta(1000, 20 + sum(y), 60 + N_obs * 20 - sum(y))
  lpd_i <- log(mean(dbinom(y[i], 20, sample_theta)))
  lppd_m2 <- lppd_m2 + lpd_i
}

# Print the results
print( lppd_m1)
print( lppd_m2)
```

```
> # Print the results
> print( lppd_m1)
[1] -20.40644
> print( lppd_m2)
[1] -25.8608
```

1.3

```r
# Observed data
y <- c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
N_obs <- 10

# Model 1
lppd_m1 <- 0
for(i in 1:N_obs){
  sample_theta <- rbeta(1000, 6 + sum(y), 6 + N_obs * 20 - sum(y))
  lpd_i <- log(mean(dbinom(y[i], 20, sample_theta)))
  lppd_m1 <- lppd_m1 + lpd_i
}

# Model 2
lppd_m2 <- 0
for(i in 1:N_obs){
  sample_theta <- rbeta(1000, 20 + sum(y), 60 + N_obs * 20 - sum(y))
  lpd_i <- log(mean(dbinom(y[i], 20, sample_theta)))
  lppd_m2 <- lppd_m2 + lpd_i
}

# Print the results
print(-2 * lppd_m1)
print(-2 * lppd_m2)
```

```
> # Print the results
> print(-2 * lppd_m1)
[1] 40.73972
> print(-2 * lppd_m2)
[1] 51.75126
```

Ans

We are calling this in-sample deviance because it evaluates how well the model fits and predicts the data on which it was trained.

1.4

Based on sample deviance model 1 has less deviance so better predictive accuracy so better fit to data

1.5

```
# New data points
new_data <- c(5, 6, 10, 8, 9)
N_new <- length(new_data)

lppd_m1 <- 0
for (i in 1:N_new) {
  sample_theta <- rbeta(1000, 6 + sum(y), 6 + N_obs * 20 - sum(y))
  lpd_i <- log(mean(dbinom(new_data[i], 20, sample_theta)))
  lppd_m1 <- lppd_m1 + lpd_i
}

lppd_m2 <- 0

for (i in 1:N_new) {
  sample_theta <- rbeta(1000, 20 + sum(y), 60 + N_obs * 20 - sum(y))
  lpd_i <- log(mean(dbinom(new_data[i], 20, sample_theta)))
  lppd_m2 <- lppd_m2 + lpd_i
}

#print results
print(-2 * lppd_m1)
print(-2 * lppd_m2)
```

```
> #print results
> print(-2 * lppd_m1)
[1] 50.46131
> print(-2 * lppd_m2)
[1] 31.62028
```

Since model 2 has less deviance it has better predictive accuracy

1.6

```r
# Observed data
y <- c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
N_obs <- 10

# Leave-one-out cross-validation for Model 1
lppd_m1 <- 0
for(i in 1:N_obs){
  ytrain <- y[-i]
  ytest <- y[i]
  sample_theta <- rbeta(1000, 6 + sum(ytrain), 6 + (N_obs - 1) * 20 - sum(ytrain))
  lpd_i <- log(mean(dbinom(ytest, 20, sample_theta)))
  lppd_m1 <- lppd_m1 + lpd_i
}

# Leave-one-out cross-validation for Model 2
lppd_m2 <- 0
for(i in 1:N_obs){
  ytrain <- y[-i]
  ytest <- y[i]
  sample_theta <- rbeta(1000, 20 + sum(ytrain), 60 + (N_obs - 1) * 20 - sum(ytrain))
  lpd_i <- log(mean(dbinom(ytest, 20, sample_theta)))
  lppd_m2 <- lppd_m2 + lpd_i
}

# Print the results
print(-2 * lppd_m1)
print(-2 * lppd_m2)
```

```
> # Print the results
> print(-2 * lppd_m1)
[1] 42.20326
> print(-2 * lppd_m2)
[1] 54.39086
```

Part 2: Marginal likelihood and prior sensitivity

2.1

```
# Function to calculate marginal likelihood
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))
  return(ML)
}

# Parameters
k <- 2
n <- 10

ML1 <- ML_binomial(2,10,0.1,0.4)
ML2 <- ML_binomial(2,10,1,1)
ML3 <- ML_binomial(2,10,2,6)
ML4 <- ML_binomial(2,10,6,2)
ML5 <- ML_binomial(2,10,20,60)
ML6 <- ML_binomial(2,10,60,20)

#print results
print(ML1)
print(ML2)
print(ML3)
print(ML4)
print(ML5)
print(ML6)
```

```
> #print results
> print(ML1)
[1] 0.4739564
> print(ML2)
[1] 0.09090909
> print(ML3)
[1] 0.004726891
> print(ML4)
[1] 0.0002313863
> print(ML5)
[1] 5.079397e-21
> print(ML6)
[1] 1.50663e-23
```

2.2

```r
# Define the likelihood function
likelihood_binomial <- function(k, n, theta){
  return(dbinom(k, n, theta))
}

# Define the Monte Carlo Integration function
MC_integration <- function(k, n, a, b, num_samples = 10000){
  theta_samples <- rbeta(num_samples, a, b)
  likelihoods <- sapply(theta_samples, function(theta) likelihood_binomial(k, n, theta))
  marginal_likelihood <- mean(likelihoods)
  return(marginal_likelihood)
}
# Parameters
k <- 2
n <- 10
#calculate for priors
MC1 <- MC_integration(2,10,0.1,0.4)
MC2 <- MC_integration(2,10,1,1)
MC3 <- MC_integration(2,10,2,6)
MC4 <- MC_integration(2,10,6,2)
MC5 <- MC_integration(2,10,20,60)
MC6 <- MC_integration(2,10,60,20)

#print results
print(MC1)
print(MC2)
print(MC3)
print(MC4)
print(MC5)
print(MC6)
```

```
> #print results
> print(MC1)
[1] 0.0400829
> print(MC2)
[1] 0.09038893
> print(MC3)
[1] 0.1990826
> print(MC4)
[1] 0.009985119
> print(MC5)
[1] 0.269771
> print(MC6)
[1] 0.0007913656
```