CGS-698C::Assignment 5

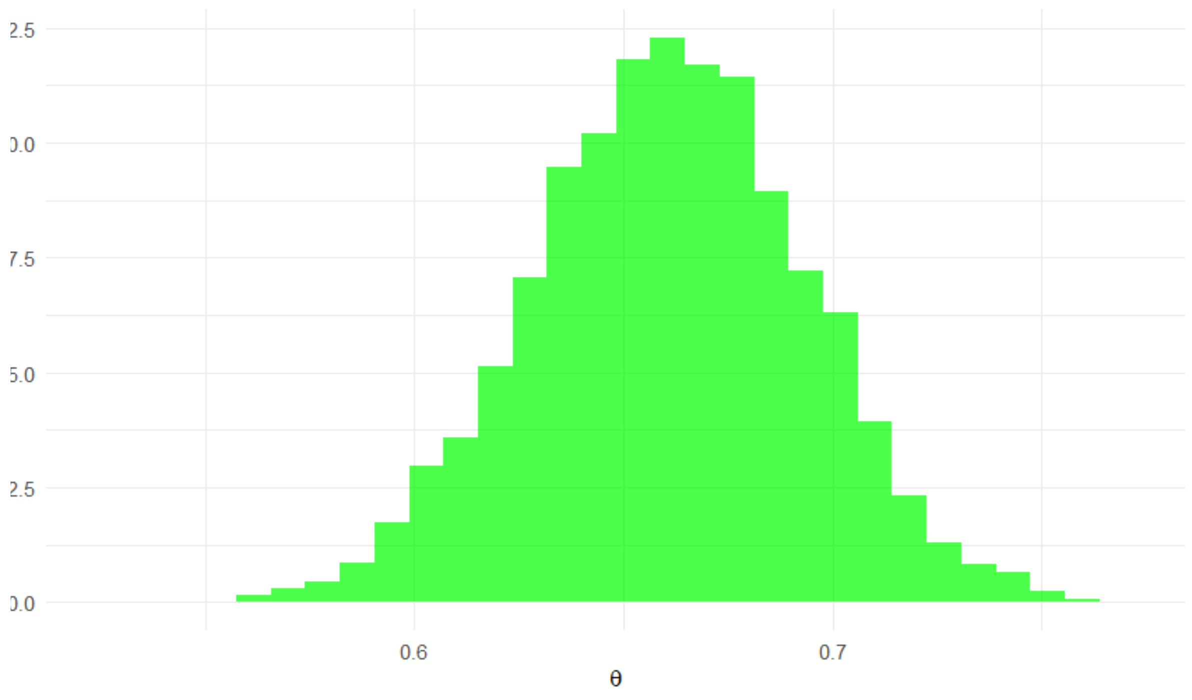
Submitted By::Anil Yadav (210138)

Part 1: Information-theoretic measures and cross-validation

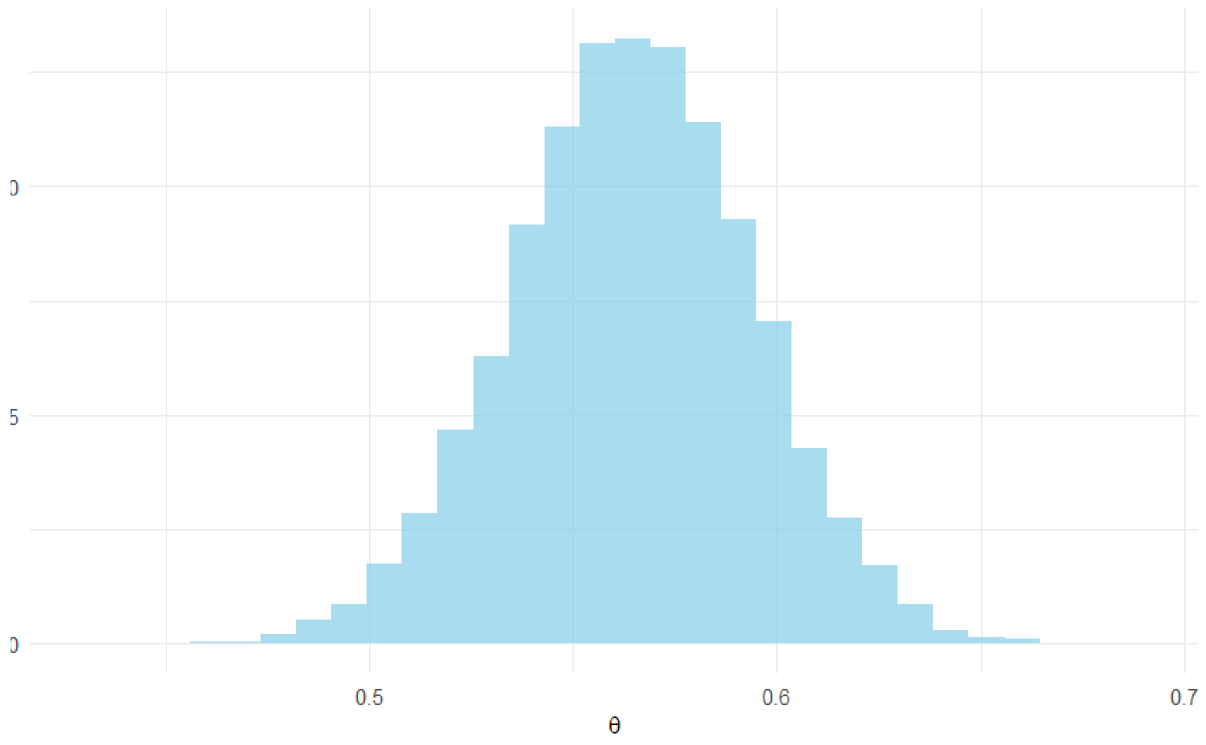
1.1

```
> library(ggplot2)
> alpha1_post <- 140
> beta1_post <- 72
>
> # Posterior parameters for Model 2
> alpha2_post <- 158
> beta2_post <- 122
>
> # Number of samples for posterior distribution
> num_samples <- 10000
>
> # Generate samples from the posterior distributions
> posterior_samples_model1 <- rbeta(num_samples, alpha1_post, beta1_post)
> posterior_samples_model2 <- rbeta(num_samples, alpha2_post, beta2_post)
>
> # Create data frames for plotting
> df_model1 <- data.frame(theta = posterior_samples_model1)
> df_model2 <- data.frame(theta = posterior_samples_model2)
>
> # Plot the histogram for Model 1

> ggplot(df_model1, aes(x = theta)) +
+  geom_histogram(aes(y = ..density..), bins = 30, fill = "green", alpha = 0.7)
+
+  labs(title = "Posterior Distribution of θ for Model 1",
+       x = expression(theta),
+       y = "Density") +
+  theme_minimal()
>
>  # Plot the histogram for Model 2

> ggplot(df_model2, aes(x = theta)) +
+  geom_histogram(aes(y = ..density..), bins = 30, fill = "skyblue", alpha =
0.7) +
+  labs(title = "Posterior Distribution of θ for Model 2",
+       x = expression(theta),
+       y = "Density") +
+  theme_minimal()
```

Posterior Distribution of θ for Model 1



Posterior Distribution of θ for Model 2



1.2

```
> y <- c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
>
> N_obs <- length(y)
>
> # Sum of data points
> sum_y <- sum(y)
>
> # Model 1 prior parameters
> alpha1_prior <- 6
> beta1_prior <- 6
>
> alpha2_prior <- 20
> beta2_prior <- 60
>
> num_samples <- 1000
>
> # Compute lppd for Model 1
> lppd_m1 <- 0
> for(i in 1:N_obs){
+  sample_theta <- rbeta(num_samples, alpha1_prior + sum_y, beta1_prior + N_obs *
20 - sum_y)
+  lpd_i <- log(mean(dbinom(y[i], 20, sample_theta)))
+  lppd_m1 <- lppd_m1 + lpd_i
+ }
>
> # Compute lppd for Model 2
> lppd_m2 <- 0
> for(i in 1:N_obs){
+  sample_theta <- rbeta(num_samples, alpha2_prior + sum_y, beta2_prior + N_obs *
20 - sum_y)
+  lpd_i <- log(mean(dbinom(y[i], 20, sample_theta)))
+  lppd_m2 <- lppd_m2 + lpd_i
+ }
>
> # Print results
> lppd_m1
[1] -20.3759
> lppd_m2
[1] -25.92667
```

1.3 #for Model 1

```
> in_sample_deviance_m1 <- -2 * lppd_m1
>
```

```
> for Model 2
> in_sample_deviance_m2 <- -2 * lppd_m2
>
> # Print results
> in_sample_deviance_m1
[1] 40.78028
> in_sample_deviance_m2
[1] 51.81876
```

1.4 Model 1 is a better fit for data.

1.5 `ynew <- c(5, 6, 10, 8, 9)`

```
> # Function to compute log predictive density for each data point and model
> compute_lppd <- function(ynew, sample_theta) {
+  lppd <- 0
+  for (i in 1:length(ynew)) {
+    lpd_i <- log(mean(dbinom(ynew[i], 20, sample_theta)))
+    lppd <- lppd + lpd_i
+  }
+  return(lppd)
+ }
> # Model 1
> lppd_m1 <- 0
> for (i in 1:length(ynew)) {
+  sample_theta <- rbeta(1000, 6 + sum(ynew), 6 + length(ynew) * 20 - sum(ynew))
+  lppd_m1 <- lppd_m1 + compute_lppd(ynew[i], sample_theta)
+ }
> deviance_m1 <- -2 * lppd_m1
> cat("Out-of-sample deviance for Model 1:", deviance_m1, "\n")
Out-of-sample deviance for Model 1: 50.645 ## more deviance
> # Model 2
> lppd_m2 <- 0
> for (i in 1:length(ynew)) {
+  sample_theta <- rbeta(1000, 20 + sum(ynew), 60 + length(ynew) * 20 - sum(ynew))
+  lppd_m2 <- lppd_m2 + compute_lppd(ynew[i], sample_theta)
+ }
> deviance_m2 <- -2 * lppd_m2
> cat("Out-of-sample deviance for Model 2:", deviance_m2, "\n")
Out-of-sample deviance for Model 2: 31.67027 ## High accuracy
```

1.6

```
 # Example data (replace with your actual data)
> y <- c(5, 6, 10, 8, 9)  # Assuming these are your data points
> N_obs <- length(y)  # Number of observations
>
```

```
> # Function to compute log predictive density for leave-one-out cross-validation
> compute_lppd_loo <- function(y, i, model_params) {
+  ytrain <- y[-i]
+  ytest <- y[i]
+  sample_theta <- rbeta(1000, model_params[1] + sum(ytrain), model_params[2] +
(N_obs - 1) * 20 - sum(ytrain))
+  lpd_i <- log(mean(dbinom(ytest, 20, sample_theta)))
+  return(lpd_i)
+ }
>
> # Model 1
> lppd_m1 <- 0
> for (i in 1:N_obs) {
+  lpd_i <- compute_lppd_loo(y, i, c(6, 6))
+  lppd_m1 <- lppd_m1 + lpd_i
+ }
> deviance_m1 <- -2 * lppd_m1
> cat("Out-of-sample deviance (LOO-CV) for Model 1:", deviance_m1, "\n")
Out-of-sample deviance (LOO-CV) for Model 1: 42.2530
>
> # Model 2
> lppd_m2 <- 0
> for (i in 1:N_obs) {
+  lpd_i <- compute_lppd_loo(y, i, c(20, 60))
+  lppd_m2 <- lppd_m2 + lpd_i
+ }
> deviance_m2 <- -2 * lppd_m2
> cat("Out-of-sample deviance (LOO-CV) for Model 2:", deviance_m2, "\n")
Out-of-sample deviance (LOO-CV) for Model 2: 54.457 ## more deviance
```

```
*****************************************************************************************
```

Part 2: Marginal likelihood and prior sensitivity

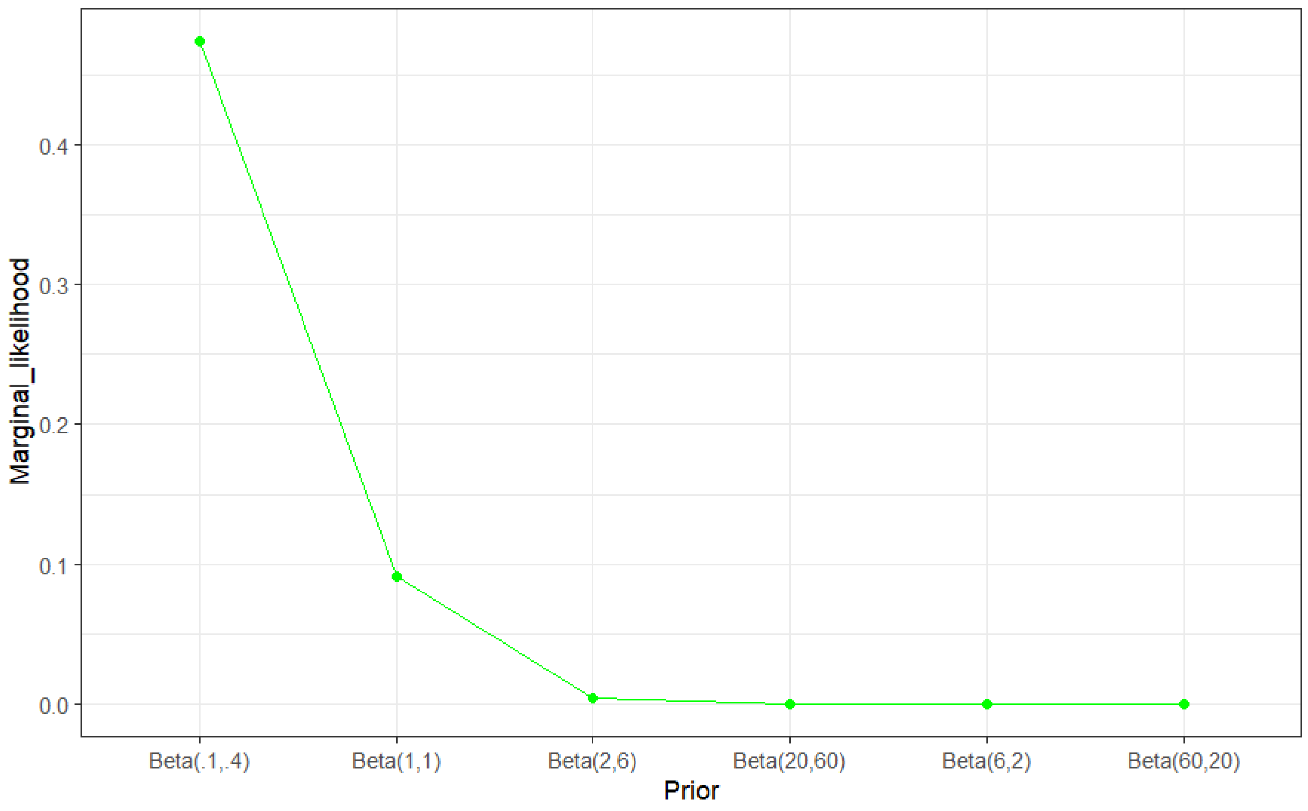
2.1

```
> 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)
+ }
>
```

```
> # Given values
> k <- 2
> n <- 10
>

> # Priors on theta: Beta(a, b)
> priors <- list(
+  Beta_1 = c(0.1, 0.4),
+  Beta_2 = c(1, 1),
+  Beta_3 = c(2, 6),
+  Beta_4 = c(6, 2),
+  Beta_5 = c(20, 60),
+  Beta_6 = c(60, 20)
+ )
>

> # Calculate marginal likelihood for each prior
> results <- sapply(priors, function(prior) {
+  ML_binomial(k, n, prior[1], prior[2])
+ })
>
> # Print results
> print(results)
      Beta_1       Beta_2       Beta_3       Beta_4       Beta_5       Beta_6 
4.739564e-01 9.090909e-02 4.726891e-03 2.313863e-04 5.079397e-21 1.506630e-23
```

2.2

```
 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)
+ }
>
> # Given parameters
> k <- 2
> n <- 10
>
> # Calculate marginal likelihood for each prior
> ML1 <- ML_binomial(k, n, 0.1, 0.4)
> ML2 <- ML_binomial(k, n, 1, 1)
> ML3 <- ML_binomial(k, n, 2, 6)
> ML4 <- ML_binomial(k, n, 6, 2)
> ML5 <- ML_binomial(k, n, 20, 60)
> ML6 <- ML_binomial(k, n, 60, 20)
>
> # Create data frame
> df.ml <- data.frame(
+  Prior = c("Beta(.1,.4)", "Beta(1,1)", "Beta(2,6)", "Beta(6,2)", "Beta(20,60)",
"Beta(60,20)"),
+  Marginal_likelihood = c(ML1, ML2, ML3, ML4, ML5, ML6)
+ )
>
> # Plot the data
> library(ggplot2)
> p <- ggplot(df.ml, aes(x = Prior, y = Marginal_likelihood, group = 1))
> p + geom_line(color = "green") + geom_point(size = 1.5, color = "green") +
theme_bw()
```


Marginal_likelihood
Beta(.1,.4)
Beta(1,1)
Beta(2,6)
Beta(20,60)
Beta(6,2)
Beta(60,20)
Prior