[ABY-Statistic-Assignment-1-anhn4@illinois.edu](mailto:ABY-Statistic-Assignment-1-anhn4@illinois.edu)

# QUESTION 1.

*Some film review aggregation websites publish ranked lists of movies based on the number of positive critical reviews out of a total number counted for each movie. Sometimes an “adjusted score” is used, for which a movie with a higher approval percentage can actually rank lower on the list*

*Consider the following hypothetical scenario:*

*Movie 1: 9 positive reviews out of 10 (90%)* *Movie 2: 425 positive reviews out of 500 (85%)* *Assume that reviews of Movie i have a common probability pi of being positive (depending on the movie) and are independent (conditional on pi). Assume a U(0, 1) prior on each pi.*

## 1a

*Determine the posterior distribution of p1 and of p2 (separately). (Name the type of distribution and give the values of its defining constants.)*

**Solution**

* Based on the desc above and since there are positive and negative review, for both movies, we will have **binomial likelihood**
* Because the desc assumed a U(0,1) equivilant to uniform distribution on interval [0,1] therefore we could assume both movies have **uniform prior**

This then lead into **Beta posterior** based on conjugate prior

Movie 1: - prior: Beta (1,1) - positive: 9 - Total review: 10 - negative = 10 - 9 = 1

**Postetior** ~ Beta(prior alpha + positive, prior beta + negative) = Beta(1+9,1+1) **Movie1\_Postetior** ~ Beta(10,2)

Movie 2: - prior: Beta (1,1) - positive: 425 - Total review: 500 - negative = 500 - 425 = 75

**Postetior** ~ Beta(prior alpha + positive, prior beta + negative) = Beta(1+425,1+75) **Movie1\_Postetior** ~ Beta(426,76)

## 1b

*Which movie ranks higher according to posterior mean? According to posterior median? According to posterior mode? Show your computations. (For median, use R function qbeta. For mean and mode, use formulas in BDA3, Table A.1. Do not use simulation, as it may not be sufficiently accurate.*

find\_higher <- function(value1, value2, name1 = "Value 1", name2 = "Value 2") {  
 cat(sprintf("%s: %.4f\n", name1, value1))  
 cat(sprintf("%s: %.4f\n", name2, value2))  
   
 if (value1 > value2) {  
 cat(sprintf("\n%s is higher by %.4f\n", name1, value1 - value2))  
 return(value1)  
 } else if (value2 > value1) {  
 cat(sprintf("\n%s is higher by %.4f\n", name2, value2 - value1))  
 return(value2)  
 } else {  
 cat("\nValues are equal\n")  
 return(value1)  
 }  
}

Posterior Mean:

# Utiliy fuction for Posterior Mean  
post\_mean <- function(alpha,beta){  
 mean = alpha/(alpha + beta)  
 return (mean)  
}  
  
# Movie 1 Posterior Mean ~ 83.33%  
movie1\_mean = post\_mean(10,2)  
  
# Movie 2 Posterior Mean ~ 84.86%  
movie2\_mean = post\_mean(426,76)   
  
# Output  
result <- find\_higher(post\_mean(10,2),post\_mean(426,76),"Post\_mean\_movie\_1","Post\_mean\_movie\_1")

## Post\_mean\_movie\_1: 0.8333  
## Post\_mean\_movie\_1: 0.8486  
##   
## Post\_mean\_movie\_1 is higher by 0.0153

Posterior Mod

# Util function for Posterior Mod  
post\_mod <- function(alpha,beta){  
 mod = (alpha - 1)/(alpha + beta - 2)  
 return (mod)  
}  
  
# Output  
result <- find\_higher(post\_mod(10,2),post\_mod(426,76),"Post\_mod\_movie\_1","Post\_mod\_movie\_1")

## Post\_mod\_movie\_1: 0.9000  
## Post\_mod\_movie\_1: 0.8500  
##   
## Post\_mod\_movie\_1 is higher by 0.0500

Posterior Median

median1 <- qbeta(0.5, 10, 2)  
median2 <- qbeta(0.5, 426, 76)  
  
# Output  
result <- find\_higher(median1,median2,"Post\_median\_movie\_1","Post\_median\_movie\_2")

## Post\_median\_movie\_1: 0.8520  
## Post\_median\_movie\_2: 0.8491  
##   
## Post\_median\_movie\_1 is higher by 0.0030

# QUESTION 2.

*File randomwikipedia.txt contains the ID number and number of bytes in length for 20 randomly selected English Wikipedia*

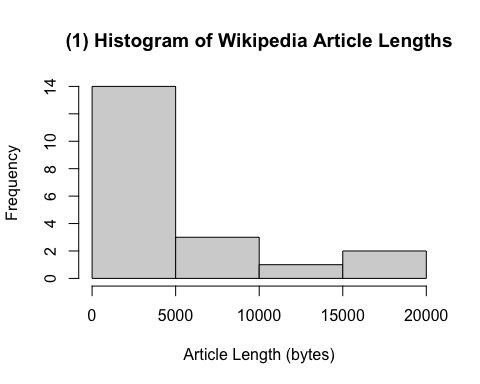
data <- read.table("randomwikipedia.txt", header=TRUE)  
head(data)

## pageID bytes  
## 1 3952653 18284  
## 2 22091611 2222  
## 3 72611703 4037  
## 4 1995807 7533  
## 5 64302262 984  
## 6 12066155 10881

## 2a.

*Diplay a histogram of article length and describe the distribution*

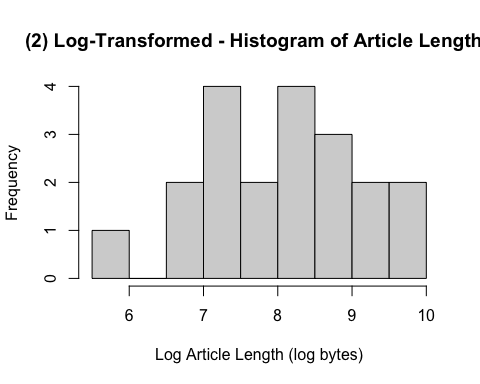
# Display histogram  
hist(data$bytes,   
 main = "(1) Histogram of Wikipedia Article Lengths",  
 xlab = "Article Length (bytes)")



Looking at this historam We can see the shape appears to be very heavy right-skew

*Transform article length to the (natural) log scale. Then re-display the histogram and describe the distribution.*

hist(log(data$bytes),  
 main = "(2) Log-Transformed - Histogram of Article Lengths",  
 xlab = "Log Article Length (log bytes)")



After the log transformation, the distribution’s right-skew is significantly reduced, provide more symmetric (normal-like) distribution.

## 2b.

*Let yi be length of article i on the log scale: the natural logarithm of the number of bytes. Compute the sample mean and sample variance of y1, . . . , y20*

*In the remaining parts, assume the yis have a normal sampling distribution with mean μ and variance σ2*

# set y as length of data on log scale  
y <- log(data$bytes)  
  
#output  
cat("Mean:", mean(y), "bytes\n")

## Mean: 8.050194 bytes

cat("variance:", var(y), "bytes\n")

## variance: 1.147371 bytes

## 2c.

*Assume σ2 is known to equal the sample variance. Consider a flat prior for μ. Use it to:*

*(i) [3 pts] Compute the posterior mean of μ, posterior variance of μ, and posterior precision of μ.*

With information above

y <- log(data$bytes)  
n <- length(y) #size  
y\_bar <- mean(y) #mean  
sigma2 <- var(y) #variance

Since the log-transformation length is close to Normal Distribition. Will assume **likelihood distibution** to be ~ N(mean(y),var(y)).

Consider a **flat prior** for μ and known σ2, the posterior distribution should be: **μ|y ~ Normal(y\_bar, σ²/n)**

post\_mean <- y\_bar  
post\_variance <- sigma2/n  
post\_precision <- 1/post\_variance  
  
#output  
cat("Posterior\_Mean:", post\_mean, "bytes\n")

## Posterior\_Mean: 8.050194 bytes

cat("Posterior\_Variance:", post\_variance, "bytes\n")

## Posterior\_Variance: 0.05736854 bytes

cat("Posterior\_Precision:", post\_precision, "bytes\n")

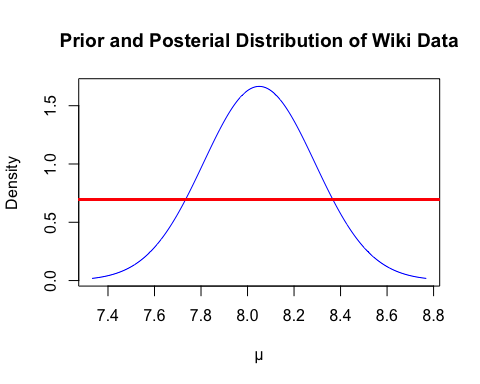
## Posterior\_Precision: 17.43116 bytes

*(ii) [3 pts] Plot a prior density of μ and a posterior density of μ together in a single plot. Label which density is which.*

# SD  
post\_sd = sqrt(post\_variance)  
  
# Generate X values for density  
x\_vals <- seq(post\_mean - 3\*post\_sd, post\_mean + 3\*post\_sd, length.out = 1000)  
  
# Calculate density  
prior\_density <- rep(1/diff(range(x\_vals)),length(x\_vals))  
post\_density <- dnorm(x\_vals, mean = post\_mean, sd = post\_sd)

Now is the plot…

# Simple Plot  
plot(x\_vals,post\_density,  
 type = 'l',  
 col = "blue",  
 xlab = "μ",  
 ylab = "Density",  
 main = "Prior and Posterial Distribution of Wiki Data")  
  
# Add Flat prior  
abline(h = prior\_density, col = "red", lwd = 2)



1. [2 pts] Compute a 95% central posterior interval for μ.

# Utility Function for posterior Interval  
calculate\_posterior\_interval <- function(post\_mean, post\_sd, conf\_level = 0.95) {  
 # Calculate alpha for the confidence level  
 alpha <- (1 - conf\_level) / 2  
   
 # Calculate the lower and upper bounds  
 lower\_bound <- qnorm(alpha, mean = post\_mean, sd = post\_sd)  
 upper\_bound <- qnorm(1 - alpha, mean = post\_mean, sd = post\_sd)  
   
 return(c(lower = lower\_bound, upper = upper\_bound))  
}  
  
# Output  
interval <- calculate\_posterior\_interval(post\_mean, post\_sd)  
cat("95% Central Posterior Interval:", "\n")

## 95% Central Posterior Interval:

cat("Lower bound:", round(interval[1], 4), "\n")

## Lower bound: 7.5807

cat("Upper bound:", round(interval[2], 4), "\n")

## Upper bound: 8.5196

## 2d.

*Now treat σ2 as unknown, and let μ and σ2 have prior*

*(i) [3 pts] Compute the posterior mean of μ, posterior variance of μ, and posterior precision of μ. (If you cannot compute explicitly, use a good computational approximation.)*

Recall information above:

y <- log(data$bytes)  
n <- length(y) #size  
y\_bar <- mean(y) #mean  
sigma2 <- var(y) #variance

Follow non-informative:

mu0 <- log(3) #Using the same log as class  
signma.2.0 <- 1.16 #Using the same info in class  
kappa0 <- 1  
nu0 <- 1

Using slide deck’s formula in class we have

mu.n <- (kappa0\*mu0 + n\*y\_bar)/(kappa0 + n)  
  
nu.n <- nu0 + n  
  
signma.2.n <- (nu0 \* signma.2.0 + (n-1)\*sigma2 + kappa0\*n\*(y\_bar - mu0)^2/(kappa0+n))/nu.n  
  
post\_precision\_mu <- n/signma.2.n  
  
#output  
cat("Posterior\_Mean\_of\_μ:", mu.n, "bytes\n")

## Posterior\_Mean\_of\_μ: 7.719166 bytes

cat("Posterior\_Variance\_of\_μ:", signma.2.n, "bytes\n")

## Posterior\_Variance\_of\_μ: 3.284922 bytes

cat("Posterior\_Precision\_μ:", post\_precision\_mu, "bytes\n")

## Posterior\_Precision\_μ: 6.088424 bytes

*(ii) [2 pts] Approximate a 95% central posterior interval for μ. (Make sure your approximation is reasonably accurate.)*

# Using t-distribution quantiles  
df <- n - 1  
alpha <- 0.05  
t\_crit <- qt(1 - alpha/2, df)  
margin <- t\_crit \* sqrt(sigma2/n)  
mu\_interval <- c(y\_bar - margin, y\_bar + margin)  
  
cat("\n95% CI for μ:", "\n")

##   
## 95% CI for μ:

cat("Lower bound:", mu\_interval[1], "\n")

## Lower bound: 7.548879

cat("Upper bound:", mu\_interval[2], "\n")

## Upper bound: 8.55151

*(iii) [2 pts] Approximate a 95% central posterior interval for σ2. (Make sure your approximation is reasonably accurate.)*

# Using chi-square distribution  
chi\_lower <- qchisq(alpha/2, df)  
chi\_upper <- qchisq(1 - alpha/2, df)  
sigma2\_interval <- c((n-1)\*sigma2/chi\_upper, (n-1)\*sigma2/chi\_lower)  
  
cat("\n95% CI for σ²:", "\n")

##   
## 95% CI for σ²:

cat("Lower bound:", sigma2\_interval[1], "\n")

## Lower bound: 0.6635769

cat("Upper bound:", sigma2\_interval[2], "\n")

## Upper bound: 2.447651

## 2e.

*Assume the prior of the previous part. Use simulation in R to answer the following,based on 1,000,000 independent draws from the posterior.*

# Set seed for reproducibility  
set.seed(123)  
  
# Get original data parameters  
y <- log(data$bytes)  
n <- length(y)  
y\_bar <- mean(y)  
sigma2 <- var(y)  
min\_bytes <- min(data$bytes)  
  
# Number of simulations  
nsim <- 1000000

*(i) [2 pts] Approximate a 95% central posterior predictive interval for the length (in bytes) of a single (new) randomly selected article. (Note that this interval is on the original scale, not the log scale.)*

# Set sigma2 draw from posterior and mu draw from posterior  
sigma2\_draws <- ((n-1)\*sigma2)/rchisq(nsim, n-1) #using inverse-gamma  
mu\_draws <- rnorm(nsim, mean=y\_bar, sd=sqrt(sigma2\_draws/n)) #using normal dist

# Generate new observation  
new\_log\_obs <- rnorm(nsim, mean=mu\_draws, sd=sqrt(sigma2\_draws))  
new\_obs <- exp(new\_log\_obs)  
  
# Calc interval  
pred\_interval <- quantile(new\_obs, c(0.025, 0.975))  
cat("95% Posterior Predictive Interval (bytes):", pred\_interval, "\n")

## 95% Posterior Predictive Interval (bytes): 314.9036 30991.68

*(ii) [2 pts] Approximate the posterior predictive probability that the length of a single (new) randomly selected article will be less than the minimum article length in the data.*

prob\_less\_min <- mean(new\_obs < min\_bytes)  
cat("Probability of new article being smaller than minimum:", prob\_less\_min, "\n")

## Probability of new article being smaller than minimum: 0.026153

*(iii) [2 pts] Approximate the posterior predictive probability that the minimum length of 20 (new) randomly selected articles will be less than the minimum article length in the data. (Be careful! All 20 randomly selected articles have the same value for μ and for σ2, since they all come from the same population.*

min\_20\_articles <- numeric(nsim)  
for(i in 1:nsim) {  
 # Generate 20 observations using the same μ and σ²  
 new\_log\_obs\_20 <- rnorm(20, mean=mu\_draws[i], sd=sqrt(sigma2\_draws[i]))  
 new\_obs\_20 <- exp(new\_log\_obs\_20)  
 min\_20\_articles[i] <- min(new\_obs\_20)  
}  
  
prob\_min\_20\_less <- mean(min\_20\_articles < min\_bytes)  
cat("Probability minimum of 20 new articles is smaller than minimum:", prob\_min\_20\_less, "\n")

## Probability minimum of 20 new articles is smaller than minimum: 0.352766