STAT 578 - Advanced Bayesian Modeling - Fall 2019
Assignment 1

Xiaoming Ji

Problem 1

The film review aggregator website rottentomatoes.com publishes ranked lists of movies based on the number of positive critical reviews out of a total number counted for each movie. See, for example, https://www.rottentomatoes.com/top/bestofrt/?year=2018. Because the site uses an "Adjusted Score" a movie with a higher approval percentage sometimes ranks lower on the list. Consider the following hypothetical scenario: * Movie 1: 150 positive reviews out of 200 (75%) * Movie 2: 4 positive reviews out of 5 (80%)

Assume that reviews of Movie `i` are independent with a common probability p_i of being positive (depending on the movie). Assume a U(0, 1) prior on each p_i.

Problem 1.a [4 pts]

Determine the posterior distribution of p_1 and of p_2 (separately).
[**Solution**]
$p_1 \sim Beta(\alpha = 151, \beta = 51)$
$p_2 \sim Beta(\alpha = 5, \beta = 2)$

Problem 1.b [3 pts]

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.)
[**Solution**]

```
p1_alpha_post = 151
p1_beta_post = 51
p2_alpha_post = 5
p2_beta_post = 2

p1_post_mean =  p1_alpha_post / (p1_alpha_post + p1_beta_post)
p2_post_mean =  p2_alpha_post / (p2_alpha_post + p2_beta_post)

p1_post_median = qbeta(0.5, 151, 51)
p2_post_median = qbeta(0.5, 5, 2)

p1_post_mode =  (p1_alpha_post - 1) / (p1_alpha_post + p1_beta_post - 2)
p2_post_mode =  (p2_alpha_post - 1) / (p2_alpha_post + p2_beta_post - 2)

cat("P1 Posterior Median:", p1_post_median, ", P2 Posterior Median:", p2_post_median, "\n")
```

```
## P1 Posterior Median: 0.748343 , P2 Posterior Median: 0.73555
```

```
cat("P1 Posterior Mean:", p1_post_mean, ", P2 Posterior Mean:", p2_post_mean, "\n")
```

```
## P1 Posterior Mean: 0.7475248 , P2 Posterior Mean: 0.7142857
```

```
cat("P1 Posterior Mode:", p1_post_mode, ", P2 Posterior Mode:", p2_post_mode, "\n")
```

```
## P1 Posterior Mode: 0.75 , P2 Posterior Mode: 0.8
```

Accorind to the calculations, **Movie 1** rank higer on **posterior median and mean**. Movie 2 rank higher on **posterior mode**.

Problem 2

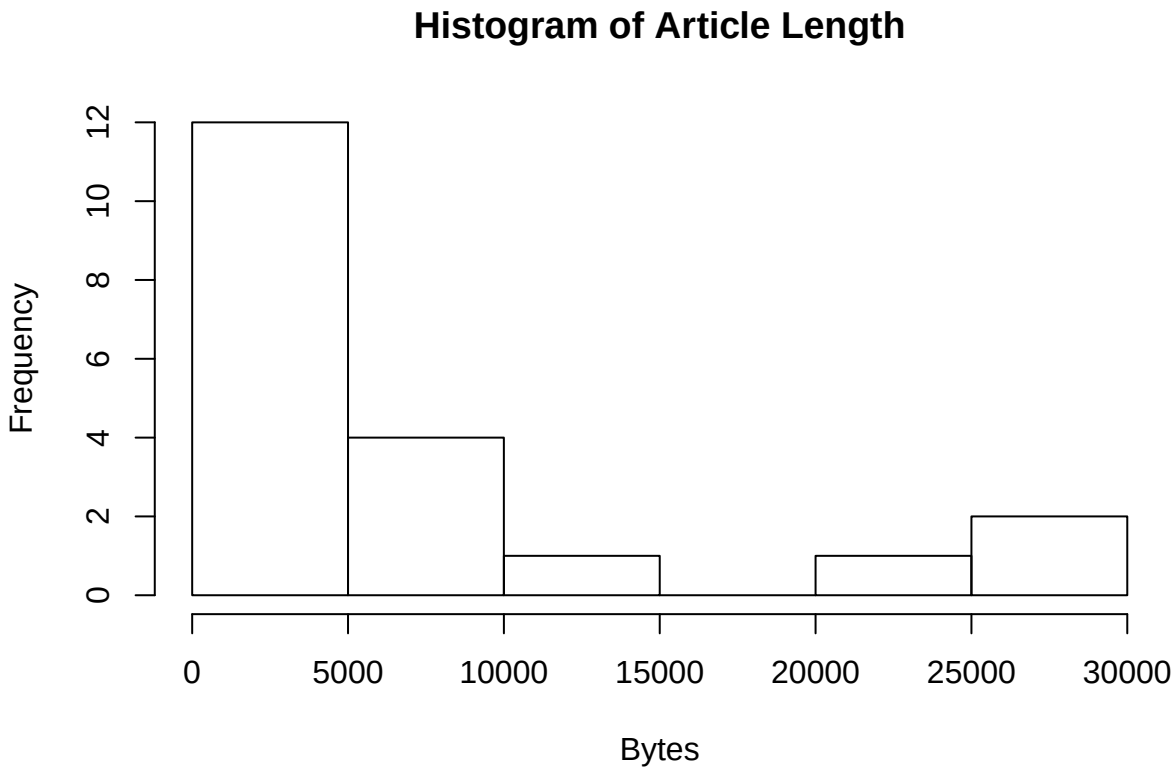
File `randomwikipedia.txt` contains the ID number and number of bytes in length for 20 randomly selected English Wikipedia articles.

Problem 2.a

(i)[2 pts]

Draw a histogram of article length, and describe the distribution.
[Solution]

```
wiki_data = read.csv('randomwikipedia.txt', sep="")
hist(wiki_data$bytes, xlab = "Bytes", main = "Histogram of Article Length")
```

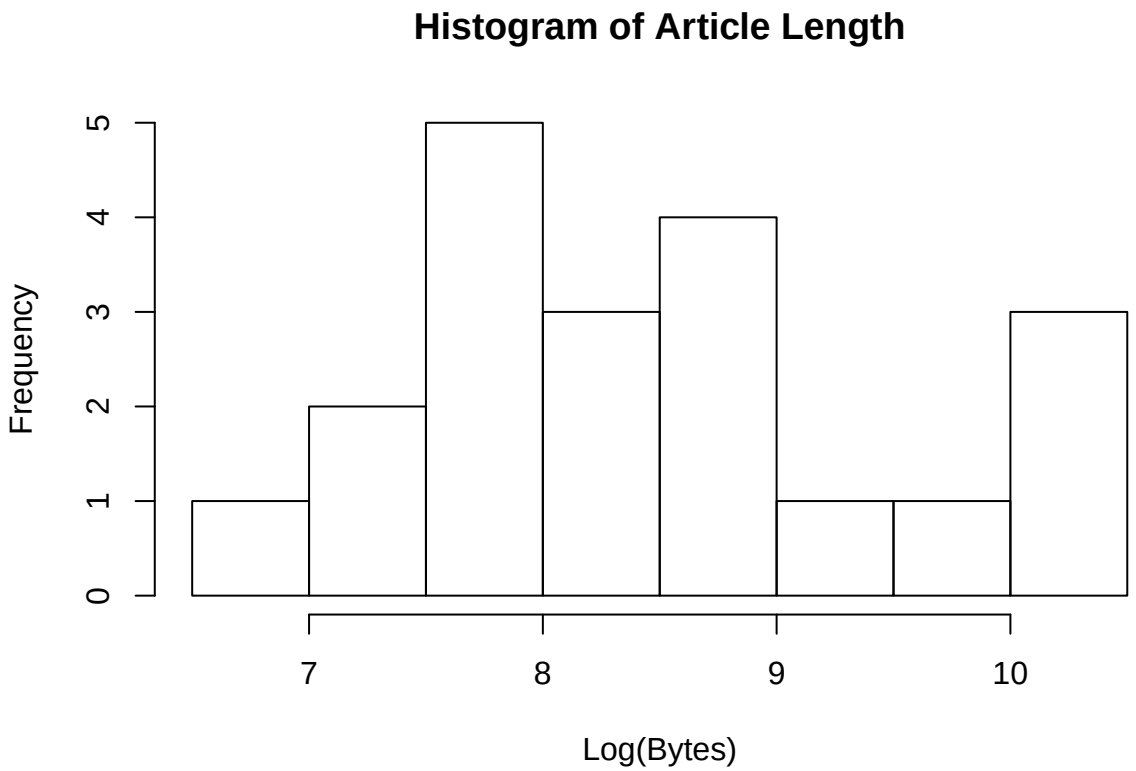


The distribution is right skewed.

(ii) [2 pts]

Transform article length to the (natural) log scale. Then re-draw the histogram and describe the distribution.
[Solution]

```r
log_bytes = log(wiki_data$bytes)
hist(log_bytes, xlab = "Log(Bytes)", main = "Histogram of Article Length")
```



The log scale distribution is close to normal distribution but with outliers on high bytes value.

(iii) [1 pt]

Based on your histograms, explain why the log scale would be better to use for the remainder of the analysis. (Read below.)
[**Solution**]
We see after log scale, the distribution has much less skew and close to normal distribution. This simplified parameters inference by using normal distribution models for later analysis.

Problem 2.b [2 pts]

Let y_i be length of article i on the log scale (i.e., the natural logarithm of the number of bytes). Compute the sample mean and sample variance of $y_i,\ldots,y_{20}$.
In the remaining parts, assume the $y_i s$ have a normal sampling distribution with mean μ and variance σ^2.

[**Solution**]

```r
n = length(log_bytes)
sample_mean = mean(log_bytes)
sample_variance = var(log_bytes)
cat("(log) Sample mean:", sample_mean, ", (log) Sample variance:", sample_variance)
```

```
## (log) Sample mean: 8.454133 , (log) Sample variance: 1.014709
```

Problem 2.c

Assume σ^2 is known to equal the sample variance. Consider a plat prior for μ. Use it to:

(i) [3 pts]

Compute the posterior mean, posterior variance, and posterior precision of μ.

[Solution]

```
sigma.2 = sample_variance
ybar = sample_mean
mu_mean_post = ybar
mu_var_post = sigma.2 / n
mu_pre_post = 1 / mu_var_post

cat("Posterior mean, vairance and precision of mu are:",
    mu_mean_post, ",", mu_var_post, ",", mu_pre_post)
```

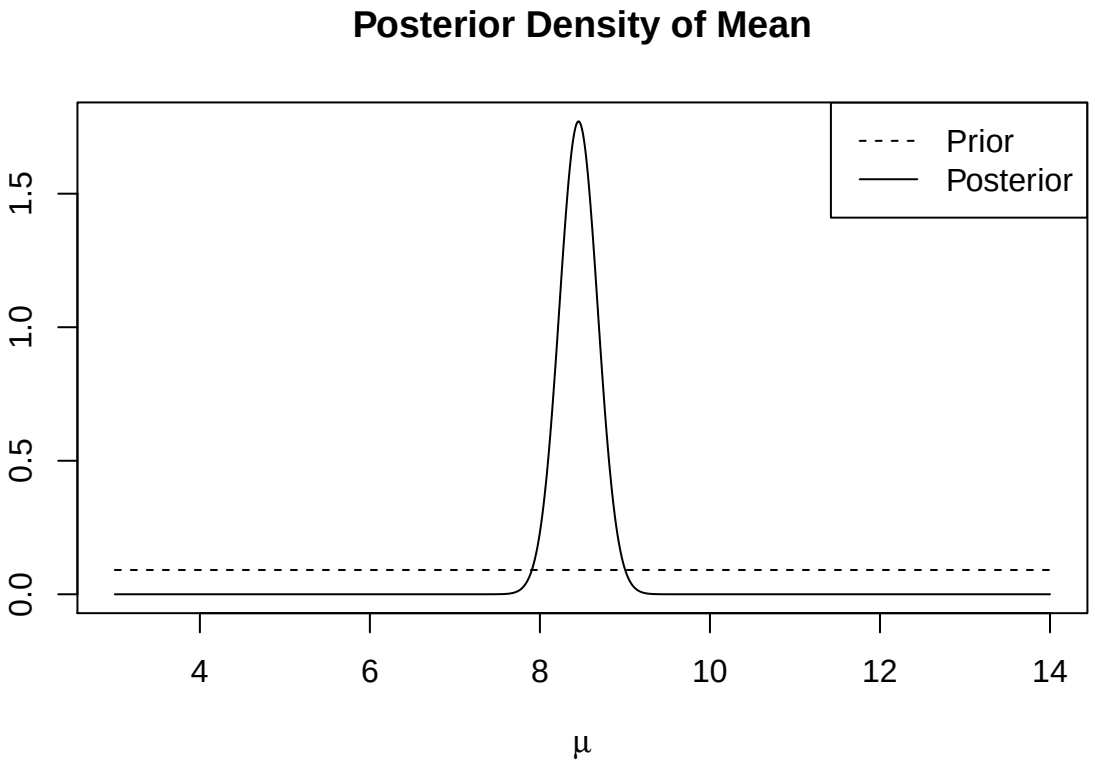
```
## Posterior mean, vairance and precision of mu are: 8.454133 , 0.05073543 , 19.71009
```

(ii) [2 pts]

Plot the prior density and the posterior density of μ together in a single plot. Label which is which.

[Solution]

```
x_start = 3
x_end = 14
curve(dnorm(x,mu_mean_post, sqrt(mu_var_post)), x_start, x_end, n=1000,
      xlab=expression(mu), ylab="", main="Posterior Density of Mean")
curve(dunif(x, x_start, x_end), x_start, x_end, add=TRUE, lty=2)
legend("topright", legend = c("Prior", "Posterior"), lty = c(2, 1))
```



(iii) [2 pts]

Compute a 95% central posterior interval for μ.
[Solution]

```
intervals = qnorm(c(0.025, 0.975), mu_mean_post, sqrt(mu_var_post))
cat("95% central posterior interval is: (", intervals[1], ",", intervals[2], ")")
```

```
## 95% central posterior interval is: ( 8.012661 , 8.895606 )
```

Problem 2.d

Now let μ and σ^2 have prior: $p(\mu, \sigma^2) \propto (\sigma^2)$. Use it to:

(i) [3 pts]

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

```
set.seed(816)
simulation_count = 1000000
sigma.2_post_sim = (n - 1) * sigma.2 / rchisq(simulation_count, n - 1)
mu_post_sim = rnorm(simulation_count, ybar, sqrt(sigma.2_post_sim / n))

mu_mean_post_sim = mean(mu_post_sim)
mu_var_post_sim = var(mu_post_sim)
mu_pre_post_sim = 1 / mu_var_post_sim
```

```r
cat("(simulated) Posterior mean, vairance and precision of mu are:",
    mu_mean_post_sim, ",", mu_var_post_sim, ",", mu_pre_post_sim)
```

```
## (simulated) Posterior mean, vairance and precision of mu are: 8.454055 , 0.05661536 , 17.66305
```

(ii) [2 pts]

Approximate a 95% central posterior interval for μ.
[Solution]

```r
print(quantile(mu_post_sim, c(0.025,0.975)))
```

```
##     2.5%    97.5%
## 7.982723 8.925314
```

(iii) [2 pts]

Approximate a 95% central posterior interval for σ^2.
[Solution]

```r
print(quantile(sigma.2_post_sim, c(0.025,0.975)))
```

```
##      2.5%     97.5%
## 0.5860402 2.1668347
```

Problem 2.e

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

(i) [2 pts]

Approximate a 95% central posterior predictive interval for the length (in bytes) of a single (new) randomly drawn article. (Note that this is on the original scale, not the log scale.)

[Solution]

```r
#simulation_count2 = 1000000
#sigma.2_post_sim2 = (n - 1) * sigma.2 / rchisq(simulation_count2, n - 1)
#mu_post_sim = rnorm(simulation_count, ybar, sqrt(sigma.2_post_sim / n))
pred_post_sim = rnorm(simulation_count, mu_post_sim, sqrt(sigma.2_post_sim))
interval = exp(quantile(pred_post_sim, c(0.025, 0.975)))
cat("95% central posterior predictive interval (in bytes):(",
    round(interval[[1]]),",", round(interval[[2]]), ")")
```

```
## 95% central posterior predictive interval (in bytes):( 544 , 40764 )
```

(ii) [2 pts]

Approximate the posterior predictive probability that the length of a single (new) randomly drawn article will exceed the maximum article length in the data.
[Solution]

```r
mean(pred_post_sim > log(max(wiki_data$bytes)))
```

```
## [1] 0.049057
```

(iii) [2 pts]

Approximate the posterior predictive probability that the maximum length of 20 (new) randomly drawn articles will exceed the maximum article length in the data. (Be careful! The 20 randomly drawn articles have the same value for μ and for σ^2.)

[Solution]

```
max_byte_log = log(max(wiki_data$bytes))

compare = function(i) {
   max(rnorm(20, mu_post_sim[i], sqrt(sigma.2_post_sim[i]))) > max_byte_log
}
mean(sapply(1:simulation_count, compare))
```

```
## [1] 0.552632
```