Assignment 1

anonymous

1 General information

I did not use AI for solving this exercise.

2 Basic probability theory notation and terms

- **probability**: Based on known parameters, It quantifies the chance of an event happening, ranging from 0 (impossible) to 1 (certain).
- **probability mass (function)**: It is a function that assigns probabilities to discrete random variables, showing the likelihood of each possible outcome.
- **probability density (function)**: It describes the relative likelihood of continuous outcomes, often shown as a curve.
- **probability distribution**: It's a mathematical representation showing all possible values of a random variable and their associated probabilities.
- **discrete probability distribution**: It applies to variables with distinct, separate values and their respective probabilities.
- **continuous probability distribution**: It's for variables with an infinite range of values within a specified interval, usually represented as a smooth curve.
- **cumulative distribution function (cdf)**: It gives the probability that a random variable is less than or equal to a particular value.
- **likelihood**: It represents the probability of observing specific data or outcomes, given a particular hypothesis or model.

3 Basic computer skills

Do some setup here. Explain in text what you do.

The following provided R code snippet is used to set up the calculation of the parameters α and β for a Beta distribution based on a given mean and variance. The parameters α and β are calculated based on the equations:

$$\alpha = \mu\left(\frac{\mu(1-\mu)}{\sigma^2} - 1\right), \quad \beta = \frac{\alpha(1-\mu)}{\mu}.$$

```
# Do some setup:
distribution_mean = .2
distribution_variance = .01

# You have to compute the parameters below from the given mean and variance
distribution_alpha = distribution_mean *
  ((distribution_mean * (1 - distribution_mean)) / distribution_variance - 1)
```

```
distribution_beta = distribution_alpha *
  (1 - distribution_mean) / distribution_mean
```

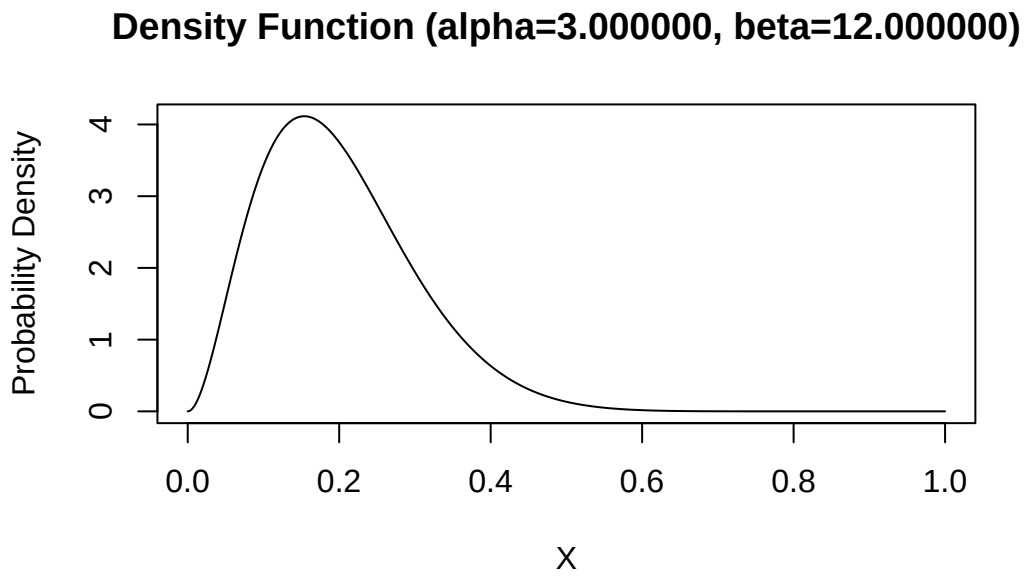
3.1 (a)

Plot the PDF here. Explain in text what you do.

```
# Useful functions: seq(), plot() and dbeta()
x <- seq(0, 1, length.out = 1000)

pdf <- dbeta(x, distribution_alpha, distribution_beta)

plot(x, pdf, type = "l",
     xlab = "X", ylab = "Probability Density",
     main = sprintf("Density Function (alpha=%f, beta=%f)",
                    distribution_alpha, distribution_beta))
```



A step-by-step explanation of the above code:

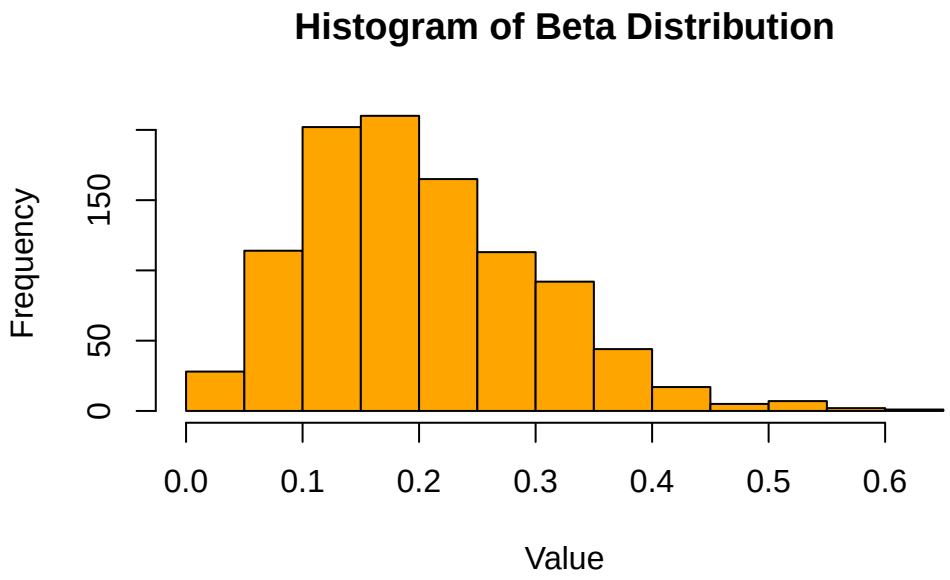
1. **`x <- seq(0, 1, length.out = 1000)`**: This line creates a sequence of 1000 equally spaced values between 0 and 1, which will be used as the x-axis values for the plot.
2. **`pdf <- dbeta(x, distribution_alpha, distribution_beta)`**: This line calculates the PDF values for the Beta distribution at each value of **`x`**.
3. **`plot(...)`**: This line creates the line plot with the axis' labels and plot name.

3.2 (b)

Sample and plot the histogram here. Explain in text what you do.

```
# Useful functions: rbeta() and hist()
n_samples <- 1000
samples <- rbeta(n_samples, distribution_alpha, distribution_beta)
```

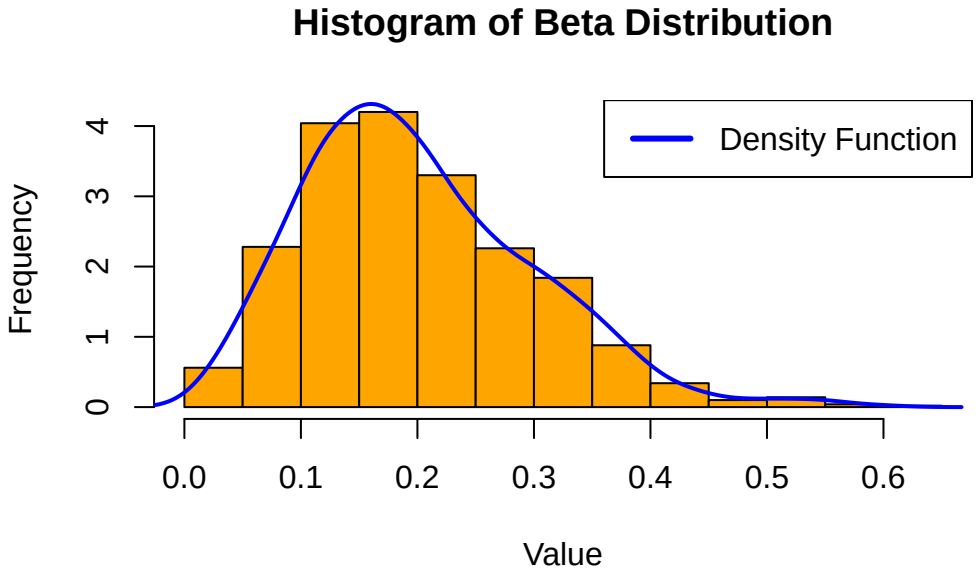
```r
hist(samples, main = "Histogram of Beta Distribution",
     xlab = "Value", ylab = "Frequency",
     col = "orange", border = "black")
```




```r
# This part is for comparing visually the density function
par(mfrow = c(1, 1))

hist(samples, main = "Histogram of Beta Distribution",
     xlab = "Value", ylab = "Frequency",
     col = "orange", border = "black", prob = TRUE)

lines(density(samples), col = "blue", lwd = 2)
legend("topright", legend = sprintf("Density Function"), col = "blue", lwd = 3)
```




```r
par(mfrow = c(2, 1))
```

A step-by-step explanation of the above code:

- **`n_samples <- 1000`**: This line sets the number of samples to be drawn from the Beta distribution to 1000.
- **`samples <- rbeta(n_samples, distribution_alpha, distribution_beta)`**: This line generates a sample of 1000 random numbers from a Beta distribution with shape parameters **`distribution_alpha`** and **`distribution_beta`**.
- **`hist(...)`**: This code creates a histogram of the generated samples.

Basically as you can see, comparing the histogram plot and the density function plot shows that it fits well to density function.

3.3 (c)

Compute the sample mean and variance here. Explain in text what you do.

```
# Useful functions: mean() and var()
sample_mean <- mean(samples)
sample_variance <- var(samples)

cat("True Mean:", distribution_mean, "\n")
```

```
True Mean: 0.2
```

```
cat("True Variance:", distribution_variance, "\n")
```

```
True Variance: 0.01
```

```
cat("Sample Mean:", sample_mean, "\n")
```

```
Sample Mean: 0.2003801
```

```
cat("Sample Variance:", sample_variance, "\n")
```

```
Sample Variance: 0.009724018
```

A step-by-step explanation of the above code:

- **`sample_mean <- mean(samples)`**: This line calculates the sample mean of the drawn samples using the **`mean()`** function.
- **`sample_variance <- var(samples)`**: This line calculates the sample variance of the drawn samples using the **`var()`** function.
- **`cat(...)`**: These lines display
    1. The true mean of the Beta distribution.
    2. The true variance of the Beta distribution.
    3. The calculated sample mean.
    4. The calculated sample variance.

3.4 (d)

Compute the central interval here. Explain in text what you do.

```
# Useful functions: quantile()
lower_quantile <- quantile(samples, 0.025)
upper_quantile <- quantile(samples, 0.975)

cat("Central 95% Probability Interval:",
    lower_quantile, "to", upper_quantile, "\n")
```

```
Central 95% Probability Interval: 0.04803697 to 0.413825
```

A step-by-step explanation of the above code:

- **`quantile(samples, c(0.025, 0.975))`**: This line calculates the quantiles of the **`samples`** data at the 2.5th and 97.5th percentiles, which correspond to the central 95% probability interval.
- **`cat(...)`**: This line displays the central 95% probability interval by printing the lower and upper bounds of the interval.

4 Bayes' theorem 1

4.1 (a)

First of all, we need to calculate the probability that an individual has lung cancer given a positive test result using Bayes' theorem.

Let's denote the following probabilities:

- $P(\text{has cancer}) \approx \frac{1}{1000}$.
- $P(\text{does not have cancer}) = 1 - P(\text{has cancer})$.
- $P(\text{Test gives positive} \mid \text{Subject has lung cancer}) = 0.98$.
- $P(\text{Test gives positive} \mid \text{Subject does not have lung cancer}) = 0.04$.

Now, by applying Bayes' theorem, we have:

$$P(\text{has cancer} \mid \text{test result is positive}) = \frac{P(\text{test result is positive} \mid \text{has cancer}) \cdot P(\text{has cancer})}{P(\text{test result is positive} \mid \text{has cancer}) \cdot P(\text{has cancer}) + P(\text{test result is positive} \mid \text{does not have cancer}) \cdot P(\text{does not have cancer})}$$

$$P(\text{has cancer} \mid \text{test result is positive}) = \frac{0.98 \cdot (1/1000)}{0.98 \cdot (1/1000) + 0.04 \cdot (999/1000)}$$

$$P(\text{has cancer} \mid \text{test result is positive}) = \frac{0.00098}{0.00098 + 0.03996} \approx 0.0239$$

So, $P(\text{has cancer} \mid \text{test result is positive})$ is 2.39%. Based on this calculation, the probability that a person has lung cancer, given a positive test result, is quite low. This means that the test has a high rate of false positives, which may lead to unnecessary procedures, or surgery for those who do not actually have lung cancer. The researchers should consider reducing false positives before marketing it widely.

5 Bayes' theorem 2

You will need to change the numbers to the numbers in the exercise.

```r
boxes_test <- matrix(c(2, 4, 1, 5, 1, 3), ncol = 2,
          dimnames = list(c("A", "B", "C"), c("red", "white")))
print(boxes_test)
```

```
  red white
A   2     5
B   4     1
C   1     3
```

5.1 (a)

Keep the below name and format for the function to work with `markmyassignment`:

```r
p_red <- function(boxes) {
    # Do computation here, and return as below.
    # This is the correct return value for the test data provided above.
    prob_A <- 0.40
    prob_B <- 0.10
    prob_C <- 0.50
    p_red_given_A <- boxes["A", "red"] / sum(boxes["A", ])
    p_red_given_B <- boxes["B", "red"] / sum(boxes["B", ])
    p_red_given_C <- boxes["C", "red"] / sum(boxes["C", ])
    probability_red <- prob_A * p_red_given_A +
      prob_B * p_red_given_B +
      prob_C * p_red_given_C

  return(probability_red)
}
probability_red <- p_red(boxes_test)
cat("Probability of picking a red ball:", probability_red, "\n")
```

```
Probability of picking a red ball: 0.3192857
```

5.2 (b)

Keep the below name and format for the function to work with `markmyassignment`:

```r
p_box <- function(boxes) {
    # Do computation here, and return as below.
    # This is the correct return value for the test data provided above.
    probability_red <- p_red(boxes)
    p_box_A <- 0.40
    p_box_B <- 0.10
    p_box_C <- 0.50

    p_red_given_A <- boxes["A", "red"] / sum(boxes["A", ])
    p_red_given_B <- boxes["B", "red"] / sum(boxes["B", ])
    p_red_given_C <- boxes["C", "red"] / sum(boxes["C", ])

    denominator <- probability_red * p_box_A +
```

```
      probability_red * p_box_B +
      probability_red * p_box_C

    p_A_given_red <- (p_red_given_A * p_box_A) / denominator
    p_B_given_red <- (p_red_given_B * p_box_B) / denominator
    p_C_given_red <- (p_red_given_C * p_box_C) / denominator

    return(c(p_A_given_red, p_B_given_red, p_C_given_red))
}

box_probabilities <- p_box(boxes_test)
cat("Probabilities of each box (A, B, C)
    has been chosen given a picked red ball:\n")
```

```
Probabilities of each box (A, B, C)
    has been chosen given a picked red ball:
```

```
cat(sprintf("Box A: %.3f%% \nBox B: %.3f%% \nBox C: %.3f%%",
            box_probabilities[1]*100,
            box_probabilities[2]*100,
            box_probabilities[3]*100))
```

```
Box A: 35.794%
Box B: 25.056%
Box C: 39.150%
```

The most probable box is box C.

6 Bayes' theorem 3

6.1 (a)

First of all, we want to calculate $P(\text{Identical} \mid \text{Elvis had a twin brother})$. We know that the probabilities are as follows:

- $P(Identical) = \frac{1}{400}$
- $P(\text{Elvis had a twin brother} \mid \text{Identical}) = 1$, this is because we already know that Elvis is male, and definitely his twin is a male.
- $P(\text{Elvis had a twin brother}) = P(\text{Identical}) + P(\text{Fraternal}) * \frac{1}{2}$, this $\frac{1}{2}$ is because we are looking for twins that Elvis had brother in.

You will need to change the numbers to the numbers in the exercise.

```
fraternal_prob = 1/150
identical_prob = 1/400
```

Keep the below name and format for the function to work with `markmyassignment`:

```r
p_identical_twin <- function(fraternal_prob, identical_prob) {
    # Do computation here, and return as below.
    # This is the correct return value for the test data provided above.
    p_elvis_had_a_twin_brother_given_identical = 1
    p_identical_given_elvis_has_a_twin_brother =
      p_elvis_had_a_twin_brother_given_identical * identical_prob /
      (fraternal_prob * 0.5 + identical_prob)
  return(p_identical_given_elvis_has_a_twin_brother)
}
result = p_identical_twin(fraternal_prob, identical_prob)
cat(sprintf("The probability that Elvis was an identical twin is %.3f%%", result * 100))
```

```
The probability that Elvis was an identical twin is 42.857%
```

7 The three steps of Bayesian data analysis

7.1 (a)

1. Setting up a full probability model—a joint probability distribution for all observable and unobservable quantities in a problem. The model should be consistent with knowledge about the underlying scientific problem and the data collection process.
2. Conditioning on observed data: calculating and interpreting the appropriate posterior distribution—the conditional probability distribution of the unobserved quantities of ultimate interest, given the observed data.
3. Evaluating the fit of the model and the implications of the resulting posterior distribution: how well does the model fit the data, are the substantive conclusions reasonable, and how sensitive are the results to the modeling assumptions in step 1? In response, one can alter or expand the model and repeat the three steps.