Assignment 1

anonymous

2023-09-08

General information

Setup

This block will only be visible in your HTML output, but will be hidden when rendering to PDF with quarto for the submission. **Make sure that this does not get displayed in the PDF!**

This is the template for assignment 1. You can download the qmd-file or copy the code from this rendered document after clicking on `</> Code` in the top right corner.

Please replace the instructions in this template by your own text, explaining what you are doing in each exercise.

The following will set-up `markmyassignment` to check your functions at the end of the notebook:

```
library(markmyassignment)
assignment_path = paste("https://github.com/avehtari/BDA_course_Aalto/",
"blob/master/assignments/tests/assignment1.yml", sep="")
set_assignment(assignment_path)
```

```
## Assignment set:
## assignment1: Bayesian Data Analysis: Assignment 1
## The assignment contain the following (3) tasks:
## - p_red
## - p_box
## - p_identical_twin
```

Basic probability theory notation and terms

1.probability: Probability is a numerical measure used to assess the likelihood of an event occurring.

2.probability mass (function): A probability mass function is a function that describes the probabilities of a discrete random variable at specific values.

3.probability density (function): The probability density function is a function that describes the probability distribution of continuous random variables, which describes the probability of a random variable taking values in any given interval.

4.probability distribution: A probability distribution is a function that describes the probability distribution of a random variable, which gives the probability of an event occurring under various possible states or outcomes.

5.discrete probability distribution: It describes the probability of each possible outcome in a finite or countable sample space.

6.continuous probability distribution:Continuous probability distribution is a function that describes the probability distribution of random variables taking values in continuous intervals.

7.cumulative distribution function (cdf): The cumulative distribution function is the integral of the probability density function, which describes the probability distribution of a real random variable X, indicating the probability that the random variable X is less than a certain value.

8.likelihood:It is used to indicate the possibility that a hypothesis is correct given the evidence.

Basic computer skills

I just used the R to applied the provided formula to calculate alpha and beta.

```
# 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_
distribution_beta <- distribution_alpha * (1 - distribution_mean) / distribution_mean
```

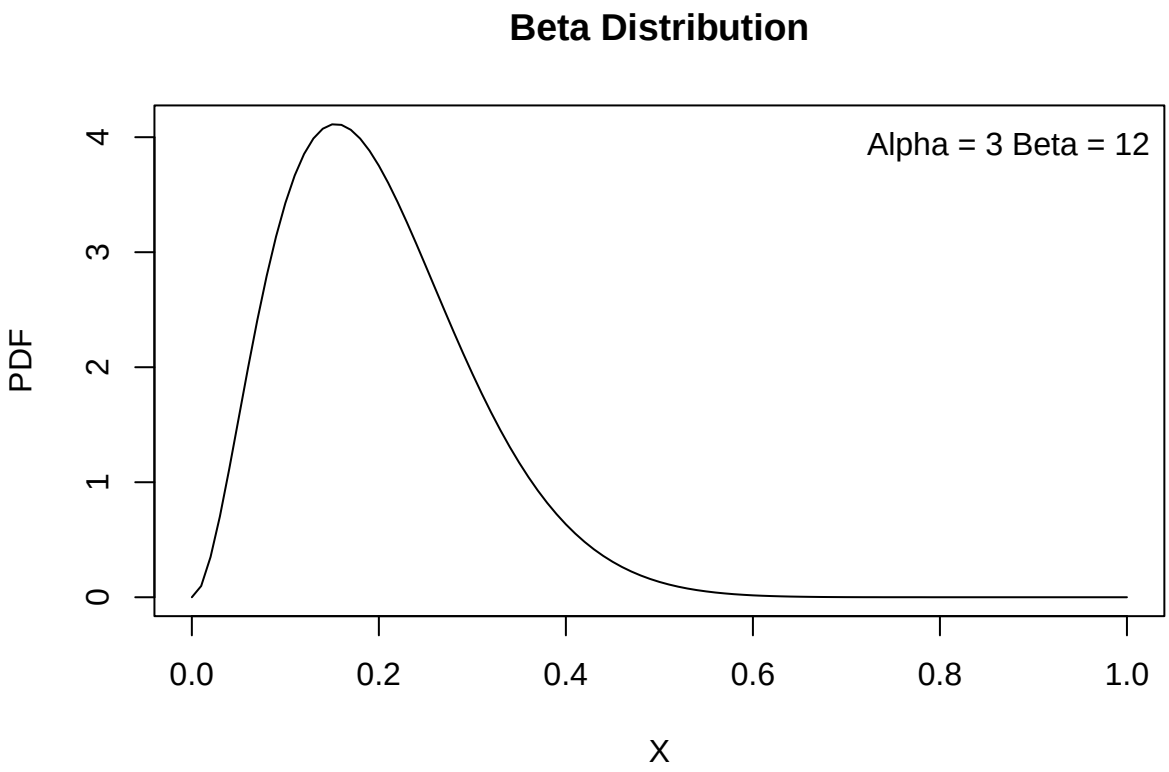
(a)

I choose a sequence of numbers ranging from 0 to 1 with an increment of 0.01. This sequence represents the possible values of a Beta distribution.

Then, the probability density function (PDF) of a Beta distribution is calculated for each value in the sequence. The shape parameters of the Beta distribution, denoted as alpha and beta, are used in this calculation.

Finally, I create a plot where the x-axis represents the possible values of the Beta distribution and the y-axis represents the corresponding PDF values.

```
# Useful functions: seq(), plot() and dbeta()
x <- seq(0, 1, by = 0.01)
pdf_values <- dbeta(x, shape1 = distribution_alpha, shape2 = distribution_beta)
plot(x, pdf_values, type = "l", xlab = "X", ylab = "PDF", main = "Beta Distribution")
legend("topright", legend = paste("Alpha =", round(distribution_alpha, 2), "Beta =", round(distribution_
```



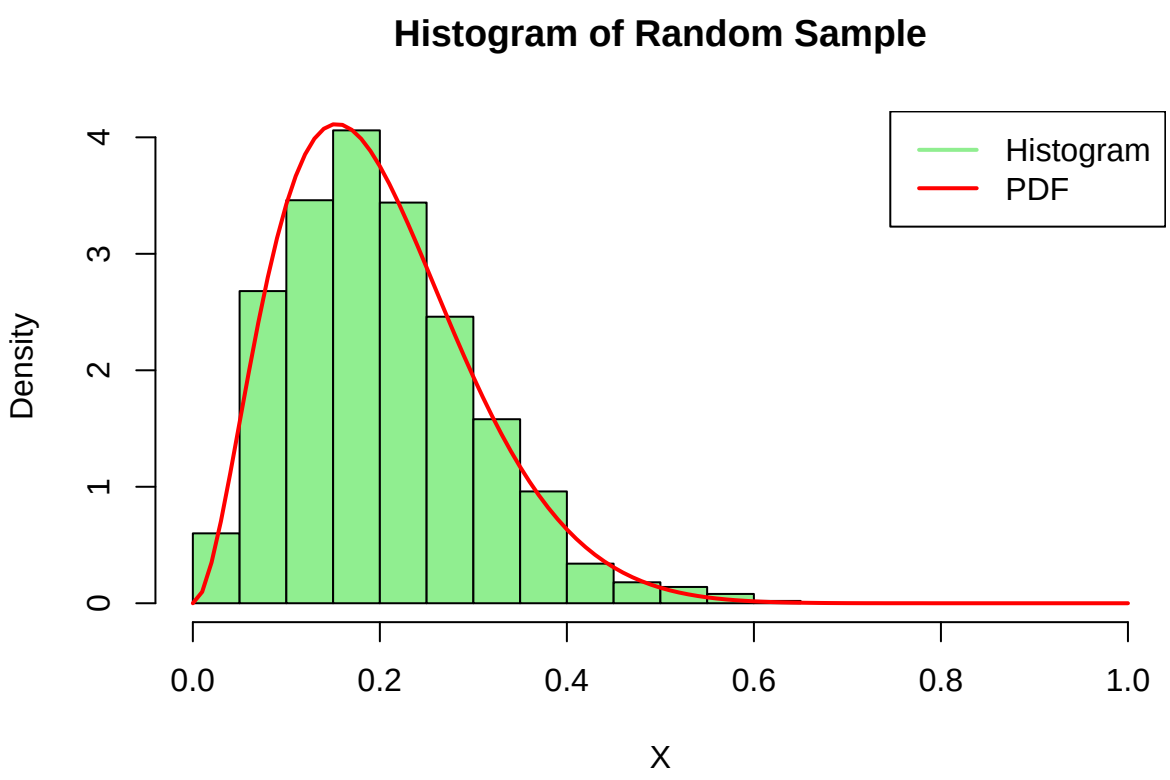
(b)

First, set the random seed to ensure reproducibility when generating random numbers. Generate a random sample of 1000 data points from a Beta distribution with two parameters. Create a histogram to display these random data points, with options for probability density, color, number of breaks, and labels for the x- and y-axes. Overlay a probability density function (PDF) curve on the histogram, represented in red. Add a legend in the top-right corner of the plot, indicating "Histogram" and "PDF" with corresponding colors and line widths.

```
# Useful functions: rbeta() and hist()
# Set the seed for reproducibility
set.seed(123)

# Generate a random sample from the Beta distribution
random_sample <- rbeta(1000, shape1 = distribution_alpha, shape2 = distribution_beta)

# Plot a histogram of the random sample
hist(random_sample, probability = TRUE, col = "lightgreen", breaks = 20, xlab = "X", ylab = "Density", m
lines(x, pdf_values, col = "red", lwd = 2)
legend("topright", legend = c("Histogram", "PDF"), col = c("lightgreen", "red"), lwd = 2)
```



(c)

I used the mean() and var() functions to calculate the sample mean and sample variance of the random sample, and then compared them with the true mean and true variance.

```
# Useful functions: mean() and var()
# Compute the sample mean and sample variance
sample_mean <- mean(random_sample)
sample_variance <- var(random_sample)

# True mean and variance
true_mean <- distribution_mean
true_variance <- distribution_variance

# Display the computed and true mean and variance
cat("Sample Mean:", sample_mean, "\n")
```

```
## Sample Mean: 0.202496
```

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

```
## Sample Variance: 0.01021622
```

```r
cat("True Mean:", true_mean, "\n")
```

```
## True Mean: 0.2
```

```r
cat("True Variance:", true_variance, "\n")
```

```
## True Variance: 0.01
```

Based on the results, I found that the sample mean and sample variance are close to the true mean and true variance.

(d)

First, determine the confidence level. In this example, the confidence level is 95%, which means we want the interval we obtain to have a 95% probability of containing the true parameter value. Then, calculate the lower and upper quantiles. In this example, we want to find the central 95% confidence interval, so we need to find the 2.5th and 97.5th quantiles. Finally, use the quantile() function to estimate the lower and upper quantiles for a given random sample. These two values constitute the central 95% confidence interval.

```r
# Useful functions: quantile()
# Compute the central 95% probability interval
confidence_level <- 0.95
lower_quantile <- (1 - confidence_level) / 2
upper_quantile <- 1 - lower_quantile

# Estimate the interval using quantiles
interval <- quantile(random_sample, c(lower_quantile, upper_quantile))

# Display the estimated central 95% probability interval
cat("Central 95% Probability Interval:", interval[1], "to", interval[2], "\n")
```

```
## Central 95% Probability Interval: 0.04506154 to 0.4290916
```

Bayes' theorem 1

(a)

I have defined the parameters of a medical test used to detect lung cancer, including the probability of testing positive in the presence of lung cancer, the probability of testing positive in the absence of lung cancer, and the probability of having lung cancer in a population. The Bayes' Theorem:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Then, I used Bayes' Theorem(above formula) to calculate the probability that a person actually has lung cancer when tested positive.

$$P(cancer|positive) = \frac{P(positive|cancer) \cdot P(cancer)}{P(positive)}$$

From the question we can know that P(Pos|LC) = 0.98, P(Neg|no LC) = 0.96, P(LC) = 0.001, based on this we can use the Bayes' theorem to calculate the probability that a person actually has lung cancer given a positive test.

```
# Define parameters
p_positive_given_cancer <- 0.98  # Probability of a positive test given lung cancer
p_negative_given_no_caner <- 0.96
p_positive_given_no_cancer <- 1 - p_negative_given_no_caner  # Probability of a positive test given no
p_cancer <- 0.001  # Probability of having lung cancer in the population
p_no_cancer <- 1 - p_cancer  # Probability of not having lung cancer in the population
p_positive_and_cancer <- p_cancer * p_positive_given_cancer
print(paste("Probability of test being positive and the subject having lung cancer:",p_positive_and_can
```

```
## [1] "Probability of test being positive and the subject having lung cancer: 0.00098"
```

```
# Use Bayes' theorem to calculate the probability that a person actually has lung cancer given a positi
p_cancer_given_positive <- (p_positive_given_cancer * p_cancer) /
                           (p_positive_given_cancer * p_cancer + p_positive_given_no_cancer * p_no_canc

# Output the result
print(paste("The probability that a person actually has lung cancer given a positive test is: ",p_cancer
```

```
## [1] "The probability that a person actually has lung cancer given a positive test is:  0.02393746946
```

Based on the above results, my advice to the researchers would be to continue refining their test to reduce the false positive rate before bringing it to market.

Bayes' theorem 2

Below code is a basic setup.

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

(a)

From the question we can know,p(box_a)=0.4,p(box_b)=0.1,p(box_c)=0.5, and then we can compute the possibility of red ball from box a, red ball from box b, red ball from box c. Based on this, we can calculate the possibility of picking a red ball using the law of total probability.

```
p_a<-0.4
p_b<-0.1
p_c<-1-p_a-p_b
p_red <- function(boxes) {
    # Do computation here, and return as below.
    p_r_given_a <- boxes[1,1] / (boxes[1,1] + boxes[1,2])
    p_r_given_b <- boxes[2,1] / (boxes[2,1] + boxes[2,2])
    p_r_given_c <- boxes[3,1] / (boxes[3,1] + boxes[3,2])
   # Probability of picking a red ball using the law of total probability
```

```
    p_ballRed <- p_r_given_a * p_a + p_r_given_b * p_b + p_r_given_c * p_c
    return(p_ballRed)
}
#test
#p_r_given_a <- boxes_test["A", "red"] / (boxes_test["A", "red"] + boxes_test["A", "white"])
#p_r_given_b <- boxes_test["B", "red"] / (boxes_test["B", "red"] + boxes_test["B", "white"])
#p_r_given_c <- boxes_test["C", "red"] / (boxes_test["C", "red"] + boxes_test["C", "white"])
#Probability of picking a red ball using the law of total probability
#p_ballRed <- p_r_given_a * p_a + p_r_given_b * p_b + p_r_given_c * p_c
```

(b)

First, I calculated the probability of taking the red ball from each box. Then, the total probability of picking out the red ball was calculated using the full probability formula. Then, I calculated the numerator of the probability of each box being selected. Finally, I calculated the conditional probability of each box.

```
p_box <- function(boxes) {
    # Do computation here, and return as below.
    # Probability of picking a red ball from each box
    p_r_given_a <- boxes[1,1] / (boxes[1,1] + boxes[1,2])
    p_r_given_b <- boxes[2,1] / (boxes[2,1] + boxes[2,2])
    p_r_given_c <- boxes[3,1] / (boxes[3,1] + boxes[3,2])
   # Probability of picking a red ball using the law of total probability
    p_ballRed <- p_r_given_a * p_a + p_r_given_b * p_b + p_r_given_c * p_c
    # Calculate the numerator of the probability of each box being chosen
    numerator_A <- p_r_given_a * p_a
    numerator_B <- p_r_given_b * p_b
    numerator_C <- p_r_given_c * p_c

    # Calculate the conditional probability for each box
    P_A_given_R <- numerator_A / p_ballRed
    P_B_given_R <- numerator_B / p_ballRed
    P_C_given_R <- numerator_C / p_ballRed

    return(c(P_A_given_R, P_B_given_R, P_C_given_R))
}
#test
#p_r_given_a <- boxes_test["A", "red"] / (boxes_test["A", "red"] + boxes_test["A", "white"])
#p_r_given_b <- boxes_test["B", "red"] / (boxes_test["B", "red"] + boxes_test["B", "white"])
#p_r_given_c <- boxes_test["C", "red"] / (boxes_test["C", "red"] + boxes_test["C", "white"])
#Probability of picking a red ball using the law of total probability
#p_ballRed <- p_r_given_a * p_a + p_r_given_b * p_b + p_r_given_c * p_c
# Calculate the numerator of the probability of each box being chosen
#numerator_A <- p_r_given_a * p_a
#numerator_B <- p_r_given_b * p_b
#numerator_C <- p_r_given_c * p_c

# Calculate the conditional probability for each box
#P_A_given_R <- numerator_A / p_ballRed
#P_B_given_R <- numerator_B / p_ballRed
#P_C_given_R <- numerator_C / p_ballRed
#print(p_ballRed)
#print(c(P_A_given_R, P_B_given_R, P_C_given_R))
```

Bayes' theorem 3

(a)

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

p_B_given_Identical = 1 (If Elvis was an identical twin, then the probability that he had a twin brother is 100%). We can set p(B) is the total probability that Elvis had a twin brother. Then we can calculate the p(B). Finally we can use Bayes' theorem to calculate the p_Identical_given_B.

```
p_identical_twin <- function(fraternal_prob, identical_prob) {
  # Do computation here, and return as below.
  p_B_given_Identical <- 1
  p_B <- (identical_prob * 1) + (fraternal_prob * 0.5)

  # Use Bayes' theorem to calculate the posterior probability.
  p_Identical_given_B <- (p_B_given_Identical * identical_prob) / p_B

  return(p_Identical_given_B)
}
```

The three steps of Bayesian data analysis

(a)

1.Create a comprehensive probability model that includes all observable and unobservable variables in the problem. This model should be based on our understanding of the scientific problem and the process of collecting data.

2.Calculate and interpret the posterior distribution, which is the conditional probability distribution of the unobserved variables given the observed data.

3.Assess the model's fit to the data and the implications of the posterior distribution. Consider whether the model fits the data well, whether the conclusions are reasonable, and how sensitive the results are to the assumptions made in step 1. If necessary, modify or extend the model and repeat these three steps.

markmyassignment

This block will only be visible in your HTML output, but will be hidden when rendering to PDF with quarto for the submission. **Make sure that this does not get displayed in the PDF!**

The following will check the functions for which `markmyassignment` has been set up:

```
mark_my_assignment()
```

```
## v | F W S  OK | Context
## / |         0 | task-1-subtask-1-tests                                          / |         0 | p_rec
## / |         0 | task-2-subtask-1-tests                                          / |         0 | p_bo
## / |         0 | task-3-subtask-1-tests                                          / |         0 | p_id
##
## == Results ===============================================================
## [ FAIL 0 | WARN 0 | SKIP 0 | PASS 13 ]
## Yay! All done!
```