STOCKHOLM UNIVERSITY

Department of Statistics

Parfait Munezero

R PROGRAMMING EXAM

October 25, 2023

Time : 14:00 - 19:00

Results will be announced no later than November 9, 2023.

The exam should be submitted no later than 19.00 (7PM), please carefully read the provided instructions to digital exams in Safe Exam Browser (SEB). The system does not allow submission after deadline. Therefore, start the submission well in advance.

The exam is to be written as a R Markdown report. You need to submit the R markdown file (`.Rmd`) as well as an output file either in `.pdf` or `.html` format. Failure to do so will result in point reductions.

Task 1: Working with factors (5 points)

Suppose we have the following data

```
  Voted_for_last_time Will_vote_for
1                  No           Yes
2                 Yes           Yes
3                  No            No
4                  No            No
5                 Yes      Not sure
```

which consist of election-polling data for candidate X who is running for re-election. In this task you are going to analyze the above data.

1. Write you own code to create the polling data for candidate X. Present your data in a data frame. (**2 points**)

2. Create the contingency table for the above data. Arrange your output such that the rows represent frequencies for the variable "`Voted_for_last_time`". (**1 point**)

3. Report the row and column sums of the contingency table (**Hint**: Use the `apply` function). (**1 point**)

4. Turn the contingency table into proportions by dividing the table in `question 2` by the total count (the sum of the marginal counts computed in `question 3`). Then, print a message presenting the cell with the highest proportion. An example of the message is: *"The highest proportion is --- % and is observed in the category of individuals who previously voted --- (Yes/No) and will vote --- (Yes/No/Not sure) for candidate X".* Use the function `paste` to create the message; the values to fill in the blanks (---) should be passed as arguments. (**1 point**)

Task 2: Monte Carlo Simulation (15 points)

Use Monte Carlo simulation to estimate the expected value of the following game in which when you roll a fair die:

```
- You lose one dollar, if the outcome is a one or two spots.
- You win three dollars, if six spots are obtained.
- Otherwise, you earn nothing.
```

1. Compute and plot the probability mass function of the dice game (use the base R plot function with the appropriate "*type*" argument). (**3 points**)

   Note that for a fair die:

   - All the six outcome events have equal probabilities.
   - The probability of observing event A or B is $p(AUB) = p(A) + p(B)$.

2. Use R to calculate the theoretical expected value of the game. Note that the expected value of a discrete random variable X is $E(X) = \sum xp(X = x)$. (**2 points**)

3. Use Monte Carlo simulation to estimate the expected value of the game. Write a function called **dice_game** with the following input argument **reps**: the number of times to repeat the simulation. The function should return an estimate of the expected value of the game i.e. how much money you can expect to win per play of the game, on average. Use set.seed(535) inside the function. (**3 points**)

4. Run the function **dice_game** with 10 repetitions of the game and then increase the number of repetitions to 10000. Compare the two values with each other and the theoretical value. Comment on your result. (**2 points**)

5. Do a convergence analysis of the **dice_game**. This analysis can be done in R by computing the cumulative mean of the output from **dice_game** over a sequence of iterations.

Hint:

a. Create a sequence of iteration numbers from 1 to 10000 with an increment step of 10.
b. For each iteration value, run the **dice_game** with the argument **reps** set to the iteration value.
c. Compute the cumulative mean of the series of expected values computed in (b). The final output should be a vector of the same length as the iteration sequence.

Plot the cumulative means (should be a line plot) and provide appropriate labels and title. Then add a horizontal line representing the theoretical expected value computed in `Question 2`. Differentiate the cumulative mean's curve and the horizontal line by setting different line colors. Comment on your results. (**5 points**)

Task 3: String Manipulation (15 points)

Consider the following text

```
txt <- c( "The", "license", "for", "most", "software", "is","designed", "to",
"take", "away", "your", "freedom","to", "share", "and", "change", "it","By",
"contrast", "the", "GNU", "General", "Public", "License","is", "intended",
"to", "guarantee", "your", "freedom", "to","share", "and", "change", "free",
"software","to", "make", "sure", "the", "software", "is","free", "for",
"all", "its", "user")
```

Notice that every word in `txt` has been transformed to its singular form.

1. Write an R code that returns all words that contain at least one upper case letter. (**2 points**)

2. Write an R code to turn all words in `txt` to lower case. Save your results in the variable `txt_lower` and print it. (**1 point**)

3. Use `grep` to report all words in `txt_lower` that have at least two consecutive vowels. (**3 points**)

4. Consider the following part of speech classifications of the words in `txt_lower`.

```
prepositions <- c("for", "by")
pronouns <- c("it", "the", "its", "your")
adverbs <- c("all", "most", "away" )
nouns <- c("license", "software", "freedom", "user")
verbs <- c("is", "designed", "take", "share", "change", "guarantee",
"make")
```

Not all are correctly classified, therefore, any word which is not classified will be put in the category of *others*.

Write a function called **speech_class_count** with the input argument **word_class**: One of the classes such as preposition, pronoun, etc. The function should return the frequency of the input class; i.e, the sum of the number of times each individual word belonging to the class appeared in `txt_lower`. (**3 points**)

Hint: you may use `grep` and you may need a for loop to get the counts of all words in the class.

5. Create a data frame with two columns: **word_class** and **count** representing, respectively, the word class and its frequency. Remember to include the category of *others* which includes all words in `txt_lower` that do not appear in any of the five classes. Comment on your results. (**3 points**)

6. Use `ggplot2` to create a horizontal bar chart of the different word classes. Reorder the bars and add appropriate title and axes labels. (**3 points**)

Task 4: Linear Algebra (15 points)

1. Use R to solve the following system of equations:

$$x1 + 2x2 + 3x3 + 4x4 + 5x5 = 7$$

$$2x1 + x2 + 2x3 + 3x4 + 4x5 = -1$$

$$3x1 + 2x2 + x3 + 2x4 + 3x5 = -3$$

$$4x1 + 3x2 + 2x3 + x4 + 2x5 = 5$$

$$5x1 + 4x2 + 3x3 + 2x4 + x5 = 17$$

by considering an appropriate matrix equation **Ax = b**. (**3 points**)

2. Calculate (**2 points**)

$$\sum_{i=1}^{20}\sum_{j=1}^{5}\frac{i^4}{3+j}$$

3. Find the number of entries in each row which are greater than 4 in the following matrix. (**2 points**)

```
set.seed(75)
my_mat <- matrix(sample(10, size=60, replace=T), nrow = 6)
print(my_mat)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    8    8    8    7    7    5    2    2    6     5
[2,]    9    5    2    6    6    1    6    6    3     7
[3,]    5    1   10    2    5    6    8    9   10     8
[4,]    9    3    1    1    6   10   10    7    9    10
[5,]    7    3    3    3    6    4    4    6   10     2
[6,]    9    3    3    4    1    2    1   10    6     1
```

4. Consider the following matrix:

```
A <- matrix(c(1,1,3,5,2,6,-2,-1,-3), nrow = 3, byrow = T)
```

Check that A^3 is a 3 × 3 matrix with every entry equal to zero. (**2 points**)

5. Consider the following iterative equation:

$$x_n = r x_{n-1}(1 - x_{n-1}),$$

with starting value x_1.

a. Write a function `quad_function(x1, r, n)` which returns the vector ($x_1, \ldots, x_n$) computed recursively according to the above equation. **Hint**: use a loop and notice that x_n depends on the previous computed value. (**2 points**)

b. Try out the function you have written with r=2 and 0<x_1<1 and n=15 (**Hint**: Choose x_1 randomly within the accepted limits). Comment on your results. (**1 point**)

c. Run your `quad_function` with $x_1 = 0.95$ and $r = 2.97$ and $n = 500$. Then plot the function's curve for the last 200 values (use R base plot function with type="l"). Add a horizontal line representing the mean value of your vector. Label the y axis to "xn" and the x axis to "iterations". Comment on your plot. (**3 points**)

Good luck!