

CS4070-Applied Data Analytics

Challenge 1 (A1) Submission

Name: Madeeha Abdul Rahim Solanki

Misis no: m00984984

Date of Submission: 12/11/2023

Submitted to: Dr.Krishnadas Nanath

**Instructions:**

1. Data Generation: Generate 6 vectors, each containing 10 random elements. These random elements should follow a normal distribution.

2. Correlation Computation: Calculate the Pearson correlation coefficients between all pairs of vectors. There will be a total of 6\*6 correlations.

3. Printing Results: Print the all the correlation coefficients and all P-values in a structured format.

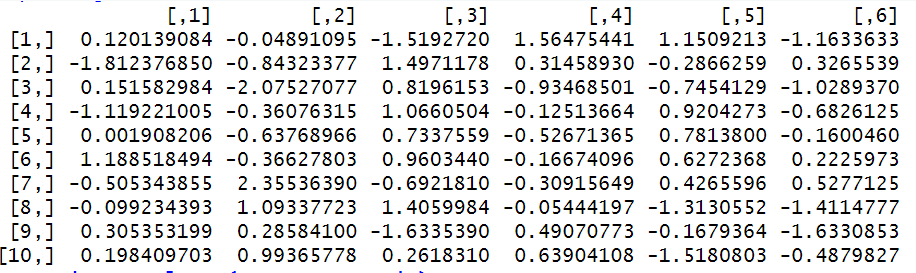
4. Interpretation: Interpret the results and discuss any correlations that are statistically significant ( p - value less than 0.10). Discuss the importance of p-values in determining the significance of correlations. Do you get any statistically significant correlations? If so, how can you justify it given that the variables are random and should not have any correlation? Format to be followed: Attempt these sub-challenges in order. Each sub-challenge should follow this format: [a] Method overview [b] Code /Output [c] Explanation of Results [d] Critical Analysis of results [e] Explanations (If something is not applicable, don’t include it).

Challenge 1: A1

Method Overview:

To generate 6 vector each containing 10 random elements first we will use the **set.seed()** function to ensure we get the same results(vector values) for the vectors each time we generate an output . If we don’t use the set.seed() function with a set number we will get different valuesIn addition, this function will prevent from random values being generated each time an output is generated . Secondly, we store the number of vectors and the number of elements inside variables named **nvector** and **no\_ofelements** ,The reason this is done is because I found it more convenient to write variable names so that we are able to access these numbers , but directly writing these numbers works the same way. Thirdly, to generate a matrix we will write an equation where we include the **matrix()** function as we want to generate a matrix, then we use the **rnorm()** function as we want to generate a random number matrix which follows normal distribution we then multiply the number of vectors we want to generate with the number of elements and we also specify the number of rows and columns that we want. Lastly, we print the matrix using the **print()** function. The reason why I chose this method is because of its efficiency .The alternate way would be to write a code for the generation of vectors again and again till we achieve the number of vectors we want or by using a loop.

b. Code /Output:

* Code:
* > set.seed(55)
* > nvector=6
* > no\_ofelements=10
* > generate\_matrix=matrix(rnorm(nvector\*no\_ofelements),ncol=6,nrow=10)
* > print(generate\_matrix)
* Output: 

[c]Explanation of Results:

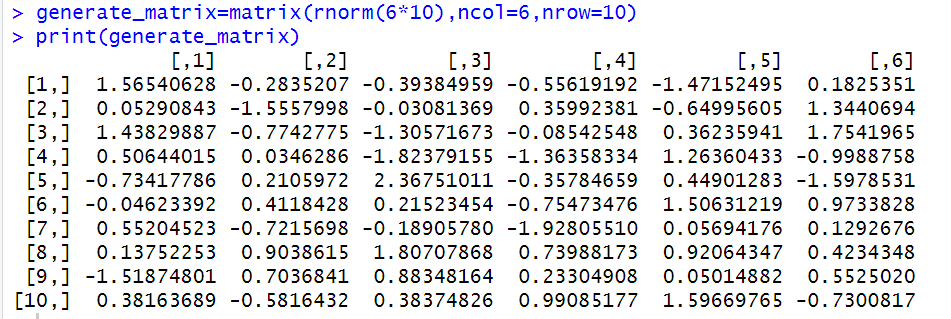
A 10x6 matrix is generated where each element is a random number following a normal distribution. Furthermore, the matrix is visible with 6 columns and 10 rows.

* Rows , columns and values:

This matrix consists of 6 columns and 10 rows each one consisting of a random element following random distribution. for the values in each cell, in the matrix a random number is generated following normal distribution where each row represents a distinct set of values and the rows represent an aspect(variable).The generated matrix can be used for analysis and testing purposes.

[d]Critical Analysis:

**Using variables to store the numbers –** the variables nvector and no\_ofelements can be easily replaced with just numbers without requiring a variable and significantly reducing the amount of code:



[e]Explanation:

set.seed()-A function which ensures the vector values remain the same each time the code is run.

rnorm()-A function which generates random values at the same time ensuring normal distribution.

Matrix()-A function which will generate the matrix in r.

2. Correlation Computation: Calculate the Pearson correlation coefficients between all pairs of vectors. There will be a total of 6\*6 correlations.

Method Overview:

The following code is trying to install the “Hmisc” package using a function in r called the **install.packages()** function.A warning is prompted for the installation of Rtools , However the required dependencies are downloaded .The code installs the “Hmisc” package and then loads it’s library using **library(“Hmisc”)** function.

b. Code /Output:

* Code:
* > install.packages("Hmisc")

library("Hmisc")

* Output :

The following code installs all the packages necessary to perform the pearson correlation and find out the p-values

[c]Explanation of Results:

The output displays the step by step installation of various dependencies and the installation of the “Hmisc” package.

**[d]Critical Analysis:**

The warning message prompt is a common message for windows systems.However, it does not show the presence of an error but simply a pre-requisite that needs to be installed for certain packages.The download of dependencies and “Hmisc” package shows that the pre-requisites are available for future use.

[e]Explanation:

RTools: The warning generated for RTools is common for all windows systems.It shows the need of the tools that would enable the compiling of the packages and is required for packages including C/Fortran code.It has to be paid attention if you plan to install packages in the future without error.

**3.[a]Generating Pearson Matrix correlation**

**Method Overview:**

The code calculates the pearson matrix correlation with the **rcorr()** function and the type is “pearson” and it is assigned to the variable named **generate\_matrix.** The following is printed together later on in the code along with the p-values to prevent extensive lines of codes.the pearson Matrix correlation is a statistical measure that shows the strength of linear relationships between variables.Moreover, the correlation coefficient has ranges which indicates a positive or a negative correlation often denoted with r:

* R=0 – This means that there is no linear correlation between the variables.
* R=1 – this denotes that there is a perfect and positive correlation of 100 %.
* R=-1 – this indicates that there is a negative correlation

There are steps to generate the pearson matrix correlation:

1.Input the data: this has been done as the vectors have been specified using the generate\_matrix variable.

2.Calculation for each pair of variables: since the correlation coefficient is calculated for each pair of variables it will result in a symmetric matrix and represent the correlation according the co-efficient r values.

[b]Code/Output:

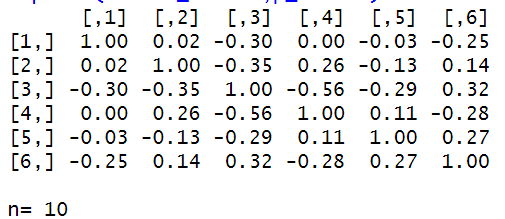
Code:

> matrix\_correl=rcorr(generate\_matrix,type="pearson")

#calculating pearson matrix correlation

Output:

\*the following matrix has been printed along with the p-values later in the code to avoid extensive lines of code.



[c].Explanation of results:

The correlation co-efficients calculated above denote the following :

* Positive correlation: if the value of r is greater than 0 it means that if one variable increases the other one tends to increase as well. I f the value is closer to 1 it means that there is a very strong correlation between the 2 variables.
* Negative correlation: This states that if the values of r is less than 0 which means that would result in, if one variable increases the other one decreases.
* No correlation: A correlation with the values 0 indicates that there is no relationship between the variables.

[d].Critical Analysis:

* Positive correlation: the values in [1,1],[2,2],[3,3],[4,4],[5,5]and[6,6] show a positive correlation indicating a linear relationship that as one variable increases the other one would increase as well.The other co-efficient values greater than 0 and closer to 1 indicates the same linear relationship but there is no 100% co-efficiency unlike the ones mentioned above.
* Negative correlation: values less than 0 for example :[3,1],[3,2] and more indicate a negative correlation which means that if one variable increase the other decreases hence they have a negative linear relationship.
* No correlation: the co-efficient values in [1,4] of 0.00 indicates that there is no correlation and linear relationship between the variables.

[e].Explanation:

There are many other ways to generate pearson co-efficient step by step including:

1. Calculate the mean:

The calc\_mean() function will calculate the mean value for the vector.

1. Pearson calculation function:

The calc\_pearson() function will calculate the pearson correlation co-efficient.

1. Manual loop:

The code can use loops to iterate through the values in the matrix.

1. Printing the result:

After all this, the co-efficients can be printed.

This method is not efficient hence, I have not used it. The method That I have used avoids the iteration through the values as it automticall y calculates them without the need of a loop to iterate. The rcorr() function aliong with the type pearson would ensure that the mean and the pearson correlation co-efficient are calculated in the same step hence making the code more feasible.

3.[b] **Calculating the p-values**

[a]Method overview:

The code below is extracting the p-values which is associated with each correlation co-efficient that is obtained from the result of the **rcorr()** function to calculate to pearson correlation co-efficient.

* **Matrix\_correl$P** is used to access the Matrix\_correl to get the coefficient values by the object which is labelled as **P**.
* **p\_values = matrix\_correl$P** the line of code above is assigned to the variable p\_values to make it convenient to print the p\_values.
* The p-values and the pearson correlation co-efficient are printed together to avois extra lines of code.

[b].Code/Output:

* Input:

>p\_values=matrix\_correl$P

>print(matrix\_correl,p\_values)

>#calculating the p-values

#printing the pearson matrix correlation and p-values

* Output:

[,1] [,2] [,3] [,4] [,5] [,6]

[1,] 1.00 0.02 -0.30 0.00 -0.03 -0.25

[2,] 0.02 1.00 -0.35 0.26 -0.13 0.14

[3,] -0.30 -0.35 1.00 -0.56 -0.29 0.32

[4,] 0.00 0.26 -0.56 1.00 0.11 -0.28

[5,] -0.03 -0.13 -0.29 0.11 1.00 0.27

[6,] -0.25 0.14 0.32 -0.28 0.27 1.00

n= 10

P

[,1] [,2] [,3] [,4] [,5] [,6]

[1,] 0.9464 0.3978 0.9981 0.9391 0.4771

[2,] 0.9464 0.3187 0.4712 0.7293 0.7060

[3,] 0.3978 0.3187 0.0918 0.4181 0.3648

[4,] 0.9981 0.4712 0.0918 0.7556 0.4371

[5,] 0.9391 0.7293 0.4181 0.7556 0.4518

[6,] 0.4771 0.7060 0.3648 0.4371 0.4518

[c].Explanation of results:

The p-values above indicate the correspondence to the significant level of the correlation coefficients. A low p-value which should be below 0.05 would let the users know that the correlation is statistically significant. The p-value for the corresponding correlations are higher than 0.05 and 0.1 stating that there is no significance observed among them.

[d].Critical Analysis:

The higher p-values generated indicate that these observed correlations are most likely due to chance as random generated vectors will not have any correlation between them and this can also relate to the Type 1 error which also known as false positive.

[e].Explanation:

There is also an alternate way to calculate p-values which is through using nested loops and calculating manually the p-values for all pairs of vectors .

1. cor.test()- This function would calculate the p-values for the vectors manually.

2. p\_value variable – the p-values generated from the cor.test() function would be stored in the p-value variable.

3. loops – The use of loops is what makes the alternate code inefficient and that’s why I have calculated the p-values as done in the code.

3.[c] – Printing the pearson correlation coefficients and the p-values in a structured format

[a].Method overview:

the following code will display the pearson correlation coefficient and the p-values side by side in a structured format .

* matrix\_data=c(matrix\_correl$r)- this code will extract all the values which were generated from the rcorr() function’s output and stores it in matrix data.
* pvalue\_data=c(matrix\_correl$P)- this code will extract all the p-values generated from the rcorr() function’s output and stores it in the pvalue\_data variable.
* structure=data.frame(matrix\_data,pvalue\_data)- this code will store the

values of the co-efficiants and p-values inside a data frame assigned to the variable

structure and finally when we print the variable structure the values are displayed side

by side in a structured format.

[b].code/output:

Input:

> #printing the matrix correlation and p-values in a structured format

> matrix\_data=c(matrix\_correl$r)

> pvalue\_data=c(matrix\_correl$P)

> structure=data.frame(matrix\_data,pvalue\_data)

> print(structure)

Output:

matrix\_data pvalue\_data

1 1.0000000000 NA

2 0.0244961325 0.94644685

3 -0.3011788825 0.39775042

4 -0.0008854487 0.99806308

5 -0.0278433465 0.93913988

6 -0.2549959681 0.47707323

7 0.0244961325 0.94644685

8 1.0000000000 NA

9 -0.3518481693 0.31874598

10 0.2583138321 0.47115727

11 -0.1257347874 0.72926231

12 0.1369361375 0.70600626

13 -0.3011788825 0.39775042

14 -0.3518481693 0.31874598

15 1.0000000000 NA

16 -0.5607217600 0.09176394

17 -0.2889413991 0.41811874

18 0.3216415367 0.36479107

19 -0.0008854487 0.99806308

20 0.2583138321 0.47115727

21 -0.5607217600 0.09176394

22 1.0000000000 NA

23 0.1131713689 0.75558406

24 -0.2777863685 0.43710130

25 -0.0278433465 0.93913988

26 -0.1257347874 0.72926231

27 -0.2889413991 0.41811874

28 0.1131713689 0.75558406

29 1.0000000000 NA

30 0.2692760597 0.45184367

31 -0.2549959681 0.47707323

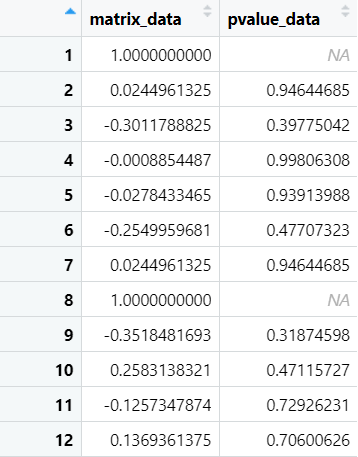
32 0.1369361375 0.70600626

33 0.3216415367 0.36479107

34 -0.2777863685 0.43710130

35 0.2692760597 0.45184367

36 1.0000000000 NA



[c].Explanation of results:

* The p-values and the pearson correlation co-efficients will be displayed side by side for each value in a structural and tabular format.

[d].Critical Analysis:

* This code uses the rcorr() function to generate the correlation matrix and extracts the correlation co-efficients and p-values ising “r” and “P”.

[e].Explanation:

* The reason why we will print the p-values and correlation coefficients side by side is because it is easier to read and understand when the date is organized in a structured format.
* 4. Interpretation –

1. Interpret the results and discuss any correlations that are statistically significant ( p - value less than 0.10).
2. Discuss the importance of p-values in determining the significance of correlations.
3. Do you get any statistically significant correlations? If so, how can you justify it given that the variables are random and should not have any correlation?

Output:

P

[,1] [,2] [,3] [,4] [,5] [,6]

[1,] 0.9464 0.3978 0.9981 0.9391 0.4771

[2,] 0.9464 0.3187 0.4712 0.7293 0.7060

[3,] 0.3978 0.3187 0.0918 0.4181 0.3648

[4,] 0.9981 0.4712 0.0918 0.7556 0.4371

[5,] 0.9391 0.7293 0.4181 0.7556 0.4518

[6,] 0.4771 0.7060 0.3648 0.4371 0.4518

Answer 4.a – Most of the p-values generated are greater than 0.1 which shows that there is no significance.However, there are 2 p-values one in [4,3] and the other one in [3,4]. A lower p-value is usually an evidence for in against with the null hypothesis.However, randomly generated vectors should not have any correlation between them and a possibility why this might be the case is purely by chance, coincidence or external factors like noise(error).

4.b – p-values determines the significance of correlation by providing statistical measure against the null hypothesis which would mean that there is no correlation between the variables of the vectors.

* Null hypothesis- this indicates that there is no significance in the correlation between the variables so a higher p-value indicates that there is no significance in the correlation.
* Against the null hypothesis: when there are p-values generated that are lower than 0.1 they usually are evidences against the null hypothesis and indicate a significance which shows that there is significance in the correlation between the variables in the vector and the more the p-values less than 0.1 mean that this is not by chance.

4.c. I have gotten 2 p-values less than 0.1 but since this is a random generated vector there is no possibility of any correlation among the variables as this is purely by chance and by the involvement of noise. Any p-values which are less than 0.05 hold a big significance and can be paid attention to. Since, this is a random generated vector and have p-values greater than 0.05 any significance will not apply.