Pset1 Exercises

Your-name-here

Part 1: Vectors and functions

Let chrNum be a vector that contains numeric information about the ploidy of each humnan chromosome. Given a normal, diploid genome, we get a vector of length 23, with each element a value of 2. Variable `chrNum` is created with the code below.

1. What data type is `chrNum`? Use the `class()` function on `chrNum`. What is the input argument and output of this function? When you write your answer in words, you can write it outside of the code section.

```
## [1] "numeric"
```

2. What are the 2nd and 4th elements of `chrNum`? (We are thinking about it as the ploidy of chr 2 and chr 4.)

```
## chr2
##    2
```

```
## chr4
##    2
```

Aneuploidy is one of the most common genetic alterations in cancer, in which entire chromosomes' ploidy change, especially during meiosis. Let us encode this change by changing the the values in `chrNum`.

2. Modify `chrNum` to the following: chr 1 and 12 have three copies, and chr 2 and 9 has one copy. At the end of each problem for this homework, use the `cat`()`function  to print out the results on text (ie.`cat(chrNum)‘)

```
## 3 1 2 2 2 2 2 2 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2
```

3. Then, chr 14 doubles. and chr 18 triples. Also, chr 21 is the sum of chr 2, 3, and 4. Tip: Is there a more algebraic way of writing it instead of `chrNum[14] = 4`? Consider `x = x * 2`.

```
## 3 1 2 2 2 2 2 2 1 2 2 3 2 4 2 2 2 6 2 2 5 2 2
```

4. What are the mean and median of `chrNum`? Use the functions `mean()` and `median()`. What are the input arguments and return values of these functions? To learn about what a function does, in your R Console, use type `?functionName` to look it up. For instance, `?mean`. You can use strings in `cat()` to add text to your printouts: ie. `cat("My answer is: ", mean(x))`.

```
## The mean is:  2.391304
## The median is:  2
```

5. Subset `chrNum` to elements that are diploid (have a value of 2) and store it in a vector called `diploids`. Hint: Use comparison operators to create logical indexing vector.

```
## 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

7. What is the length of `diploids`?

```
## Length:  16
```

8. Subset `chrNum` to elements that are non-diploid (have a value that is not 2) and store it in a vector called `nonDiploids`. What is its length?

```
## 3 1 1 3 4 6 5
```

```
## [1] 7
```

9. Now, suppose you change `chrNum`. Does that affect `nonDiploids`?
10. Modify `chrNum` so that all non-diploid chromosomes double in copy number. Tip: Use ideas from question 3.

Compare this to `nonDiploids`. How are these two vectors similar and different?

```
## chrNum: 6 2 2 2 2 2 2 2 2 2 2 6 2 8 2 2 2 12 2 2 10 2 2
```

```
## nonDiploids: 3 1 1 3 4 6 5
```

11. A property of Vectors that remains unexplored is the Names property. Use the `names()` function on chrNum to look at the names of the vector. Bonus: Rename the name of chr 23 to "chr X/Y", and give it one copy.

```
## chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12 chr13 chr14 chr15 chr16 chr17 chr18 c
```

```
## 6 2 2 2 2 2 2 2 2 2 2 6 2 8 2 2 2 12 2 2 10 2 1
```

Part 2: Dataframes

Take a look at the `iris` dataframe. It is a default variable loaded into the environment.

Use `head()` and `tail()` functions to look at the first few and last few rows of the dataframe.

Use `nrow()`, `ncol()`, `dim()`, `colnames()` functions to access properties of the dataframe.

Use `View()` function to examine the dataframe as a spreadsheet in a new tab.

1. What is the number of columns and rows for `iris`? What do you think each row corresponds to?

```
## Rows: 150
## Columns: 5
```

2. What are the means of the `Sepal.Length` and `Sepal.Width`?

```
## Mean Sepal Length:  5.843333
```

```
## Mean Sepal Width:  3.057333
```

3. The function `table()` prints out a frequency table from a string vector as the input argument. What is the frequency table for the `Species` column?

```
## Species Count:  50 50 50
```

4. What are the 2nd and 4th elements of the `Sepal.Length` column?

```
## [1] 4.9
```

```
## [1] 4.6
```

5. Examine the mean `Sepal.Length` between the three `Species`: which Species have the highest mean? Hint: Use a comparison operator on the `Species` column to create a logical indexing vector. This logical indexing vector is used to subset on `Sepal.Length` column.

```
## Setosa:  5.006
```

```
## Versicolor:  5.936
```

```
## Virginica:  6.588
```

6. Create a new dataframe consisting of only “setosa” rows and only consists of `Sepal.Length`, `Sepal.Width`, and `Species` columns. Compute the mean of Sepal.Length. Does it match with your previous answer?

```
## Sepal length for setosa:  5.006
```

7. Create a new dataframe consisting of only Septal.Length > 5.0. What is its Sepal.Length mean?

```
## 6.129661
```

8. Take the dataframe from problem 7 and subset it further to “setosa” species. What is its Sepal.Length mean?

```
## 5.313636
```

9. Bonus: The two subsetting steps from Problems 7 and 8 can be combined into a single step. Create a logical indexing vector that is TRUE if Sepal.Length > 5.0 AND the Speciies is “setosa”. Then use it to subset on the original `iris` dataframe. Does the Sepal.Length matches the answer from Problem 8?

```
## [1] 0.2772727
```

9. Consider the following dataframe:

Print the frequency table of whether Your Favorite Gene (YFG) is mutated or not.

What is the mean gene expression of all cell lines?

what is the mean gene expression of mutated cell lines?

What is the mean gene expression of non-mutated cell lines?

```
##
## FALSE  TRUE
##     3     3
```

```
## [1] 2.566667
```

```
## [1] 0.7333333
```

```
## [1] 4.4
```