TEB2164 Introduction to Data Science
Lab 4a- Factors, matrices, array

Learning Outcomes

At the end of the session, you will be able to:

- Write, run, and explain the difference of factors matrices, and array in R

Activity

1. Factors

1.1. Factor Construction

- Write and run the following in R. Make your conclusion about the code:

```
# Create a vector as input.
data <-
c("East","West","East","North","North","East","West","West","West","
East","North")
print(data)
print(is.factor(data))
# Apply the factor function.
factor_data <- factor(data)
print(factor_data)
print(is.factor(factor_data))
length(factor_data)
```

1.2. Accessing Factors Elements

- Write and run the following in R. Make your conclusion about the code:

```
data <-
factor(c("East","West","East","North","North","East","West","West","
West","East","North"))
data[3]
```

1.3. Factors Elements Manipulation

- Write and run the following in R. Make your conclusion about the code:

```
data <-
factor(c("East","West","East","North","North","East","West","West","
West","East","North"))
data[3] <- "NorthWest"
```

1.4. Changing the Order of Levels

- Write and run the following in R. Make your conclusion about the code:

```
data <- c("East","West","East","North","North","East","West",
"West","West","East","North")
# Create the factors
factor_data <- factor(data)
print(factor_data)
# Apply the factor function with required order of the level.
new_order_data <- factor(factor_data,levels =
c("East","West","North"))
print(new_order_data)
```

1.5. Factor Level Generation

- Write and run the following in R. Make your conclusion about the code:

```
v <- gl(3, 4, labels = c("Tampa", "Seattle","Boston"))
print(v)
```

2. Matrices

2.1. Matrix Construction

- Write and run the following in R. Make your conclusion about the code:

```
# Elements are arranged sequentially by row.
M <- matrix(c(3:14), nrow = 4, byrow = TRUE)
print(M)
# Elements are arranged sequentially by column.
N <- matrix(c(3:14), nrow = 4, byrow = FALSE)
print(N)
# Define the column and row names.
rownames = c("row1", "row2", "row3", "row4")
colnames = c("col1", "col2", "col3")
P <- matrix(c(3:14), nrow = 4, byrow = TRUE, dimnames =
list(rownames, colnames))
print(P)
```

2.2. Accessing Elements of a Matrix

- Write and run the following in R. Make your conclusion about the code:

```
# Define the column and row names.
rownames = c("row1", "row2", "row3", "row4")
colnames = c("col1", "col2", "col3")
# Create the matrix.
P <- matrix(c(3:14), nrow = 4, byrow = TRUE, dimnames =
list(rownames, colnames))
# Access the element at 3rd column and 1st row.
print(P[1,3])
# Access the element at 2nd column and 4th row.
print(P[4,2])
# Access only the 2nd row.
print(P[2,])
# Access only the 3rd column.
print(P[,3])
```

2.3. Matrix Addition & Subtraction

- Write and run the following in R. Make your conclusion about the code:

```
# Create two 2x3 matrices.
matrix1 <- matrix(c(3, 9, -1, 4, 2, 6), nrow = 2)
print(matrix1)
matrix2 <- matrix(c(5, 2, 0, 9, 3, 4), nrow = 2)
print(matrix2)
# Add the matrices.
result <- matrix1 + matrix2
cat("Result of addition","\n")
print(result)
# Subtract the matrices
result <- matrix1 - matrix2
cat("Result of subtraction","\n")
print(result)
```

2.4. Matrix Multiplication & Division

- Write and run the following in R. Make your conclusion about the code:

```
# Create two 2x3 matrices.
matrix1 <- matrix(c(3, 9, -1, 4, 2, 6), nrow = 2)
print(matrix1)
matrix2 <- matrix(c(5, 2, 0, 9, 3, 4), nrow = 2)
print(matrix2)
# Multiply the matrices.
result <- matrix1 * matrix2
cat("Result of multiplication","\n")
print(result)
# Divide the matrices
result <- matrix1 / matrix2
cat("Result of division","\n")
print(result)
```

3. Array

3.1. Array Construction

- Write and run the following in R. Make your conclusion about the code:

```
# Create two vectors of different lengths.
vector1 <- c(5,9,3)
vector2 <- c(10,11,12,13,14,15)
# Take these vectors as input to the array.
result <- array(c(vector1,vector2),dim = c(3,3,2))
print(result)
```

3.2. Naming Columns and Rows

- Write and run the following in R. Make your conclusion about the code:

```
# Create two vectors of different lengths.
vector1 <- c(5,9,3)
vector2 <- c(10,11,12,13,14,15)
column.names <- c("COL1","COL2","COL3")
row.names <- c("ROW1","ROW2","ROW3")
matrix.names <- c("Matrix1","Matrix2")
# Take these vectors as input to the array.
result <- array(c(vector1,vector2),dim = c(3,3,2),dimnames =
list(row.names,column.names,
matrix.names))
print(result)
```

3.3. Accessing Array Elements

- Write and run the following in R. Make your conclusion about the code:

```
# Create two vectors of different lengths.
vector1 <- c(5,9,3)
vector2 <- c(10,11,12,13,14,15)
column.names <- c("COL1","COL2","COL3")
row.names <- c("ROW1","ROW2","ROW3")
matrix.names <- c("Matrix1","Matrix2")
# Take these vectors as input to the array.
```

```
result <- array(c(vector1,vector2),dim = c(3,3,2),dimnames =
list(row.names,
column.names, matrix.names))
# Print the third row of the second matrix of the array.
print(result[3,,2])
# Print the element in the 1st row and 3rd column of the 1st matrix.
print(result[1,3,1])
# Print the 2nd Matrix.
print(result[,,2])
```

3.4. Array Elements Manipulation

- Write and run the following in R. Make your conclusion about the code:

```
# Create two vectors of different lengths.
vector1 <- c(5,9,3)
vector2 <- c(10,11,12,13,14,15)
# Take these vectors as input to the array.
array1 <- array(c(vector1,vector2),dim = c(3,3,2))
# Create two vectors of different lengths.
vector3 <- c(9,1,0)
vector4 <- c(6,0,11,3,14,1,2,6,9)
array2 <- array(c(vector1,vector2),dim = c(3,3,2))
# create matrices from these arrays.
matrix1 <- array1[,,2]
matrix2 <- array2[,,2]
# Add the matrices.
result <- matrix1+matrix2
print(result)
```

3.5. Using function apply() in Array

- Write and run the following in R. Make your conclusion about the code:

```
# Create two vectors of different lengths.
vector1 <- c(5,9,3)
vector2 <- c(10,11,12,13,14,15)
# Take these vectors as input to the array.
new.array <- array(c(vector1,vector2),dim = c(3,3,2))
print(new.array)
# Use apply to calculate the sum of the rows across all the
matrices.
result <- apply(new.array, c(1), sum)
print(result)
```