SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Artificial Intelligence and Data Science

Course Name – Big Data Analytics
III Year / V Semester

Unit 2 – Data Science using Python

Topic - R Programming

R Programming

Matrices

A matrix is a two dimensional data set with columns and rows.

A column is a vertical representation of data, while a row is a horizontal representation of data.

A matrix can be created with the matrix() function. Specify the nrow and ncol parameters to get the amount of rows and columns:

```
# Create a matrix
thismatrix <- matrix(c(1,2,3,4,5,6), nrow = 3, ncol = 2)
# Print the matrix
thismatrix
```

You can also create a matrix with strings:

Example

```
thismatrix <- matrix(c("apple", "banana", "cherry", "orange"), nrow = 2, ncol = 2)
thismatrix
```

R Programming

Access Matrix Items

You can access the items by using [ ] brackets. The first number "1" in the bracket specifies the row-position, while the second number "2" specifies the column-position:

Example

```
thismatrix <- matrix(c("apple", "banana", "cherry", "orange"), nrow = 2, ncol = 2)
thismatrix[1, 2]
```

Arrays

Compared to matrices, arrays can have more than two dimensions.

We can use the array() function to create an array, and the dim parameter to specify the dimensions:

Example

```
# An array with one dimension with values ranging from 1 to 24
thisarray <- c(1:24)
thisarray
```

R Programming

Access Array Items

You can access the array elements by referring to the index position. You can use the [] brackets to access the desired elements from an array:

Example

```
thisarray <- c(1:24)
multiarray <- array(thisarray, dim = c(4, 3, 2))
multiarray[2, 3, 2]
```

R Programming

Data Frames

Data Frames are data displayed in a format as a table.

Data Frames can have different types of data inside it. While the first column can be character, the second and third can be numeric or logical. However, each column should have the same type of data.

Use the data.frame() function to create a data frame:

Example

```
# Create a data frame
Data_Frame <- data.frame (
  Training = c("Strength", "Stamina", "Other"),
  Pulse = c(100, 150, 120),
  Duration = c(60, 30, 45)
)
```

R Programming

Factors

Factors are used to categorize data. Examples of factors are:

Demography: Male/Female
Music: Rock, Pop, Classic, Jazz
Training: Strength, Stamina

To create a factor, use the factor() function and add a vector as argument:

Example

```
# Create a factor
music_genre <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
# Print the factor
music_genre
```

Result:

```
[1] Jazz    Rock    Classic Classic Pop     Jazz    Rock    Jazz
Levels: Classic Jazz Pop Rock
```

R Programming

You can see from the example above that that the factor has four levels (categories): Classic, Jazz, Pop and Rock.

To only print the levels, use the levels() function:

Example

```
music_genre <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
levels(music_genre)
```

Result:

```
[1] "Classic" "Jazz"    "Pop"     "Rock"
```

THANK YOU