manipalglobal
Academy of Data Science

File Operations

PRESENTED BY

Vinod Raju
Data Science Practitioner

TOC

manipalglobal
Academy of Data Science

Learning Objectives

By the end of this unit, you will be able to:

- Explain the scan function used to read the data from the console
- List the methods available to read or write data from text and csv files
- Discuss the options available to import XML files

Direct Data Importing

Direct Data Importing

To read the data from the console, the scan function is used.

Pseudocode:

```
x <- scan()
```

- The scan function by default reads numeric data. However, the default behavior can be changed by using the what option.
- On specifying the what=character() or what=“ ”, all the fields will be read as strings.

manipalglobal
Academy of Data Science

TOC

Direct Data Importing

Example:

```
x<- scan()
z<-scan()
1: 10 11
3
Read 2 items
```

TOC

Read/Write Text files

Reading Text Files

read.table is a more generic way to read or write tabular data from or to a file on disk (local file system).

```
data <- read.table(‘<filename>’,
                   header = F,
                   fill = T,
                   sep=“\t” )
```

filename	The file that needs to be imported, it is usually the file path.
header	This argument is used to consider or ignore the first line to be considered as header.
fill = T	On setting this option to T (true), blank spaces are filled for column values which are empty, that is, the rows would be of unequal length.
sep=“\t”	Sep option is used to specify the delimited, in this example ‘\t’ implies it’s a tab delimited file.

Writing to Text Files

write.table works very much similar to read.table(), but the function is to write a data frame instead of reading one.

Pseudocode:

```
    write.table(mydata, “c:/mydata.txt”, sep=“\t”)
```

- A data frame “mydata” is written to a file name called mydata.txt.
- The Pseudocode is similar to read.table except for the change in source and destination.

Writing to Text Files

Data from a data frame can be written to an excel sheet.

install.packages("xlsx") #This needs an active internet connection.

Note: The installation must happen successfully in order to execute the next step.

library(xlsx) # comment: Load the library xlsx

Example:

```
write.xlsx(mydata, "c:/mydata.xlsx")
```

TOC

Read/Write from a CSV File

manipalglobal
Academy of Data Science

Reading from a CSV File

The read.csv function is identical to the read.table, except that the default separator is comma (sep = “,”).

Pseudocode:

```
          data = read.csv(“filename”)
           data = read.csv(“<file.choose()>”)
```

- file.choose() pops up a file browser window to select a file.
- The default is to assume a header row with column names. (header=TRUE) OR (header =FALSE) is to indicate, if the first line of data is to be considered as header or ignored.

Reading from a CSV File

Example:

```
gender_submission =
read.csv(“C:/Users/Administrator/Downloads/gender_submission.csv”)
View(gender_submission)
```

Note: The file and file path for read.csv could be different for each coder.

Writing to a CSV File

The write.csv function is used to write the contents of a data frame into a csv file.

Pseudocode:

```
write.csv(<Data frame name>, “<path to write>/<file_name>.csv”)
```

Example:

```
write.csv(newArrest, file=“USArrest.csv”)
```

Note: The file, USArrest.csv would be created in the current working directory of R studio, in case the full path is not mentioned.

manipalglobal
Academy of Data Science

TOC

Writing to a CSV File

Explanation:

write.csv(newArrest, file=“USArrest.csv”)

- A file named USArrest.csv will be created in working directory or R and contents of the newArrest(data frame) will be written to it, along with the row and the column identifiers.
- Row names will be added by default in the file being written using the csv command and they will be numbers starting from 1.

Writing to a CSV File

The command to change the default naming of rows is as follows:

```
write.csv(newArrest, file=“USArrest.csv”,row.names = F)
```

The command to ignore column names in a csv file created through R is as follows:

```
write.table(newArrest, file=“USArrest.csv”,col.names =
FALSE,sep=“,”)
```

manipalglobal
Academy of Data Science

TOC

Writing to a CSV File

An example demonstrating the write operation, where a data frame’s contents is written into a CSV file with explicit row and column names:

```
a=1:3
b=c(“Ram”,“Sham”,“Bob”)
c=c(“a”,“b”,“c”)
df = data.frame(a,b,c)

rownames(df) = c(“r1”,“r2”,“r3”)
colnames(df) = c(“c1”,“c2”,“c3”)

write.csv(df,file=“rowcol.csv”)
```

Read write operations onto a CSV file

Using Skip command during the import of a CSV file into R:

?

How to write the contents of a data frame into a text file?

```
write.csv(USArrests,file=“<file path>”)
```

?

How to read only the header from a CSV file?

```
header = read.csv(file=“<filename>”,skip=1,header = F,nrows = 1,as.is = T)
```

?

How to skip the first 3 lines from the CSV file when reading it?

```
mydf = read.csv(file = “< filename>”,skip=3,header = F)
```

TOC

Importing XML Files

Importing XML Files

The below are the sequence of steps along with the comments:

Commands	Comment
library(“XML”)	Loads the xml library
library(“methods”)	Loads the other required package
mydf<- xmlToDataFrame(“file path”)	This converts the xml data to data frame and assigns to a variable mydf
print(result)	Displays the contents

Summary

Summary

In this unit, you have learnt:

- **Scan** function is used to read the data from the console.
- **read.table** is a more generic way to read/write tabular data from/to a disk.
- **read.csv** and **write.csv** functions are used to read and write a data frame to a csv file.

manipalglobal
Academy of Data Science

THANK YOU!