Medium Test Analysis

Introduction

The medium test aims to **replicate** the **analysis** conducted in Section 41.1.3 using functions from the **XML** package. This test focuses on transforming XML data into a **structured data frame** for further analysis and reporting. The XML package, being an older package, offers a straightforward way to convert XML data into data frames, which is particularly useful for handling nested XML structures.The XML document provided in the test is a nested structure, with each <**node**> element containing potentially **nested** <**node**> elements (and some containing attributes). The goal is to extract specific information from this XML document, such as the **text** values, **length**, **attributes**,**name**,**children** etc. of nested <node> elements under the root <node> element.

Setting Up the Environment

Section 1: Loading Libraries and parsing XML Content

```r
library(XML)

xml_content <- c(
 '<?xml version="1.0" encoding="UTF-8"?>',
 '<movies>',
 '<movie mins="126" lang="eng">',
 '<title>Good Will Hunting</title>',
 '<director>',
 '<first_name>Gus</first_name>',
 '<last_name>Van Sant</last_name>',
 '</director>',
 '<year>1998</year>',
 '<genre>drama</genre>',
 '</movie>',
 '<movie mins="106" lang="spa">',
 '<title>Y tu mama tambien</title>',
 '<director>',
 '<first_name>Alfonso</first_name>',
 '<last_name>Cuaron</last_name>',
 '</director>',
 '<year>2001</year>',
 '<genre>drama</genre>',
 '</movie>',
 '</movies>'
)
```

Explanation

- The **XML** library is loaded to handle XML data in R.
- An XML content string representing a list of movies is defined, including details like **title**, **director**, **year**, and **genre**.

```r
doc <- xmlTreeParse(paste(xml_content, collapse = ''), useInternalNodes = TRUE)
doc
```

```
## <?xml version="1.0" encoding="UTF-8"?>
## <movies>
##   <movie mins="126" lang="eng">
##     <title>Good Will Hunting</title>
##     <director>
##       <first_name>Gus</first_name>
##       <last_name>Van Sant</last_name>
##     </director>
##     <year>1998</year>
##     <genre>drama</genre>
##   </movie>
##   <movie mins="106" lang="spa">
##     <title>Y tu mama tambien</title>
##     <director>
##       <first_name>Alfonso</first_name>
##       <last_name>Cuaron</last_name>
##     </director>
##     <year>2001</year>
##     <genre>drama</genre>
##   </movie>
## </movies>
##
```

Explanation

- The **xmlTreeParse** function from the XML package is used to parse the XML string into an XML document object.
- The **paste** function with **collapse = ' '** is used to concatenate the XML string into a single string before parsing.
- The **useInternalNodes** = **TRUE** argument specifies that the function should return an internal node, which is more efficient for extracting parts of the XML document1.
- The **parsed** XML document is stored in the variable xml_doc.

Section 2: Navigation of XML Tree

```r
# Get the root node of the XML document
movies <- xmlRoot(doc)
movies
```

2.1 Access the root Node

```
## <movies>
##   <movie mins="126" lang="eng">
##     <title>Good Will Hunting</title>
##     <director>
##       <first_name>Gus</first_name>
##       <last_name>Van Sant</last_name>
##     </director>
##     <year>1998</year>
##     <genre>drama</genre>
##   </movie>
##   <movie mins="106" lang="spa">
##     <title>Y tu mama tambien</title>
##     <director>
##       <first_name>Alfonso</first_name>
##       <last_name>Cuaron</last_name>
##     </director>
##     <year>2001</year>
##     <genre>drama</genre>
##   </movie>
## </movies>
```

```r
# Check if the XML document and the root node are identical
identical(doc, movies)
```

```
## [1] FALSE
```

It turns out that `doc` and `movies` are not actually identical

Explanation

- The **xmlRoot** function extracts the root node of the XML document, which is stored in movies.
- The identical function checks if the **root node** is the same as the **original document**, demonstrating the structure of the XML document.

```r

# Access the child nodes of the root node
xmlChildren(movies)
```

2.2 Access the children of movies node

```
## $movie
## <movie mins="126" lang="eng">
##   <title>Good Will Hunting</title>
##   <director>
##     <first_name>Gus</first_name>
##     <last_name>Van Sant</last_name>
##   </director>
##   <year>1998</year>
##   <genre>drama</genre>
## </movie>
##
## $movie
## <movie mins="106" lang="spa">
##   <title>Y tu mama tambien</title>
##   <director>
##     <first_name>Alfonso</first_name>
##     <last_name>Cuaron</last_name>
##   </director>
##   <year>2001</year>
##   <genre>drama</genre>
## </movie>
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```r
# Access the first movie node
good_will <- xmlChildren(movies)[[1]]
good_will
```

```
## <movie mins="126" lang="eng">
##   <title>Good Will Hunting</title>
##   <director>
##     <first_name>Gus</first_name>
##     <last_name>Van Sant</last_name>
##   </director>
##   <year>1998</year>
##   <genre>drama</genre>
## </movie>
```

```r
# Access the second movie node
tu_mama <- xmlChildren(movies)[[2]]
tu_mama
```

```
## <movie mins="106" lang="spa">
##   <title>Y tu mama tambien</title>
##   <director>
##     <first_name>Alfonso</first_name>
##     <last_name>Cuaron</last_name>
##   </director>
##   <year>2001</year>
##   <genre>drama</genre>
## </movie>
```

Explanation

- **xmlChildren(movies)** retrieves the child nodes of the node "movies".
- **xmlChildren(movies)**[[**1**]] accesses the first movie node from the child nodes of "movies".
- **xmlChildren(movies)**[[**2**]] accesses the second movie node from the child nodes of "movies".

Section 3: Inspecting first node

```r

# Access the children nodes of 'good_will'
xmlChildren(good_will)
```

3.1 Inspecting contents of the children of movies node

```
## $title
## <title>Good Will Hunting</title>
##
## $director
## <director>
##   <first_name>Gus</first_name>
##   <last_name>Van Sant</last_name>
## </director>
##
## $year
## <year>1998</year>
##
## $genre
## <genre>drama</genre>
```

```
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```r
# Access the children nodes of 'tu_mama'
xmlChildren(tu_mama)
```

```
## $title
## <title>Y tu mama tambien</title>
##
## $director
## <director>
##   <first_name>Alfonso</first_name>
##   <last_name>Cuaron</last_name>
## </director>
##
## $year
## <year>2001</year>
##
## $genre
## <genre>drama</genre>
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```r
# Get the name of the 'good_will' node
xmlName(good_will)
```

```
## [1] "movie"
```

```r
# Get the attributes of the 'good_will' node
xmlAttrs(good_will)
```

```
##  mins  lang
## "126" "eng"
```

```r
# Get the size (number of children) of the 'good_will' node
xmlSize(good_will)
```

```
## [1] 4
```

Explanation

- The **xmlName** function is used to get the name of the **good_will** node.
- The **xmlAttrs** function is used to get the attributes of the root node.
- The **xmlChildren** function lists all child nodes of the root node, which represent individual movies.

```r

# Iterate over each child node of 'good_will' and print their names
children_nodes <- xmlChildren(good_will)
for (node in children_nodes) {
  print(xmlName(node))
}
```

3.2 Inspecting contents of good_will node

```
## [1] "title"
## [1] "director"
## [1] "year"
## [1] "genre"
```

```
# Access the title node of 'good_will'
title1 <- xmlChildren(good_will)[["title"]]
title1
```

```
## <title>Good Will Hunting</title>
```

```
# Access the children nodes of 'title1'
xmlChildren(title1)
```

```
## $text
## Good Will Hunting
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```
# Get the text content of 'title1'
xmlValue(title1)
```

```
## [1] "Good Will Hunting"
```

Explanation

- **xmlChildren(good_will)** retrieves the child nodes of the 'good_will' node.
- **xmlChildren(good_will)**[[**"title"**]] accesses the 'title' node within the 'good_will' node.
- **xmlChildren(title1)** accesses the child nodes of the 'title1' node
- **xmlValue(title1)** extracts the text content of the 'title1' node, representing the title of the movie.

Section 4: Inspecting director node

```
# Access the director node of 'good_will'
dir1 <- xmlChildren(good_will)[["director"]]
dir1
```

```
## <director>
##   <first_name>Gus</first_name>
##   <last_name>Van Sant</last_name>
## </director>
```

```
# Access the children nodes of 'dir1'
xmlChildren(dir1)
```

```
## $first_name
## <first_name>Gus</first_name>
##
## $last_name
## <last_name>Van Sant</last_name>
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```
# Get the text content of 'dir1'
xmlValue(dir1)
```

```
## [1] "GusVan Sant"
```

Explanation

- **xmlChildren(good_will)**[[**"director"**]] accesses the 'director' node within the 'good_will' node.
- **xmlChildren(dir1)** accesses the child nodes of the 'dir1' node.
- **xmlValue(dir1)** extracts the text content of the 'dir1' node, representing the director's name.

The following **results** obtained from the code can be compared with the required section outlinedexample data set in Section 41.1.3 of Computing with Data