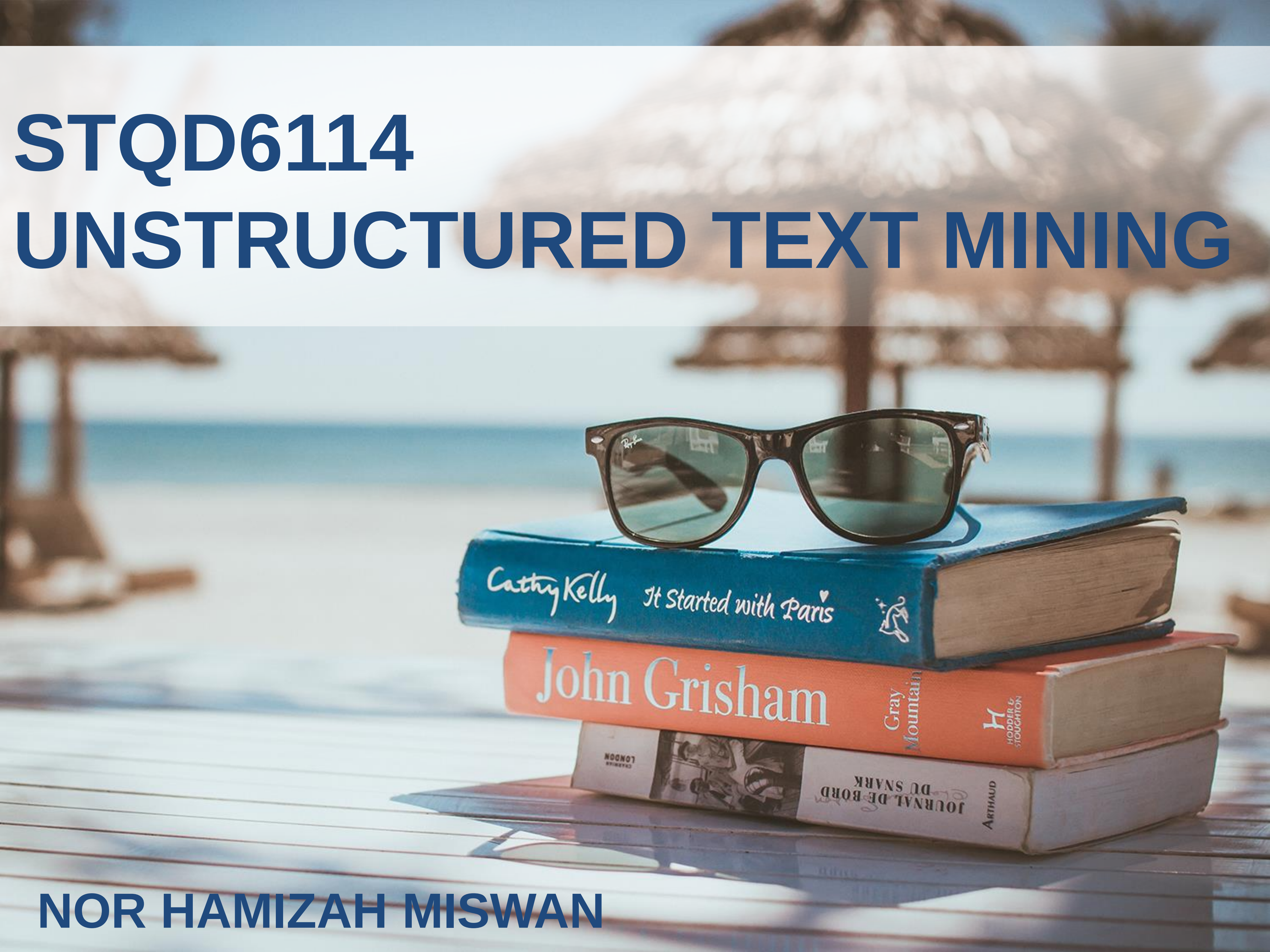

STQD6114 UNSTRUCTURED TEXT MINING

NOR HAMIZAH MISWAN

Retrieving Textual Data

- ❖ Text mining (also called text data mining or text analytics) is at its simplest, a method for drawing out content based on meaning and context from a large bodies of text.
- ❖ It is a method for gathering structured information from unstructured text.
- ❖ Text data is everywhere. According to estimates, 80% of the world's data is in "unstructured text format".
- ❖ We need methods to extract, summarize, and analyze useful information from unstructured/text data.
- ❖ Text mining seeks to automatically discover useful knowledge from the massive amount of data.
- ❖ Use of computational techniques to extract high quality information from text.

Retrieving Textual Data

- Unstructured text is present in various forms, and in huge and ever-increasing quantities:
  - books
  - financial and other business reports
  - various kinds of business and administrative documents
  - new articles
  - blog posts
  - wiki
  - messages/posts on social networking and social media sites

Extract Text from Files

- You may use the usual `read.table()` or `read.csv()` function.
- However, by using `read.table()` or `read.csv()` function, one can only run text operations on one document at a time.
- A common method for retrieving textual data in R is by using the tm package.
- The main data structure in the package is a corpus.
- A corpus is a collection of text documents.
- The benefir of using corpus is that it provides the ability to run text operations on the entire corpus.

Extract Text from Files

- There are two types of corpus: Vcorpus and Pcorpus.
- Vcorpus is the default implementation in R, which is short for volatile corpus. It is volatile since once the R object is destroyed, the whole corpus is gone.
- Meanwhile, the Pcorpus is short for permanent corpus since the documents are physically stored outside of R.
- There are predefined sources in the package:
  - DirSource – handle a directory
  - VectorSource – handle a vector
  - DataframeSource – handle data rframe like structures

Web Scraping

- There is a massive amount of data available on the web.
- However, getting the data into an analyzable format is difficult.
- Assessing online data of this sort is called “web scraping”.
- A key challenge is finding a way to unpack the data from a web page full of other elements.
- You may use the `readLines()` function that does not require any package. This function reads some or all text lines from a connection. It allows for simple access to webpage source.
- You may also use the httr package. Within this package, there is a function `GET()` which access the web.

Web Scraping

- Reading form the web will gives you a messy result. The result sometimes need to be restructured and parsed. The `htm_treeParse()` function from the XML package is tailored for this task.
- Another common package is the rvest package. In this package, there is a functuion `read_html()` which reads url into memory similar to the way we read csv files using `read.csv()` function. Then, we can extract pieces out of the HTML documents using Xpath and css selectors (selector gadget). By using css selectors to scrap, web scraping can be performed for targeted content since not all content on the web page is gold.
- We can also perform web scraping on multiple pages.

Appendix: Codes

Part 1: Extract Text from Files

```
eg1<-read.table(file.choose(),fill=T,header=F) #Data CG.txt
eg1[1,]
eg2<-read.csv(file.choose(),header=F) #Data CG.csv
eg2[1,]

#Using tm package
Library(tm)
eg3<-c("Hi!","Welcome to STQD6114","Tuesday, 11-1pm")
mytext<-VectorSource(eg3)
mycorpus<-VCorpus(mytext)
inspect(mycorpus)
as.character(mycorpus[[1]])
```

Appendix: Codes

Part 1: Extract Text from Files

```
#Example using VectorSource
eg4<-t(eg1) #From example 1
a<-sapply(1:7,function(x)
trimws(paste(eg4[,x],collapse=" "),"right"))
mytext<-VectorSource(a)
mycorpus<-VCorpus(mytext)
inspect(mycorpus)
as.character(mycorpus[[1]])
```

```
#Example using DirSource
mytext<-DirSource("movies")
mycorpus<-VCorpus(mytext)
inspect(mycorpus)
as.character(mycorpus[[1]])
```

```
#Example using DataFrameSource
eg5<-read.csv(file.choose(),header=F) #Using doc6.csv
docs<-data.frame(doc_id=c("doc_1","doc_2"),
text=c(as.character(eg5[1,]),as.character(eg5[2,])),
dmeta1=1:2,dmeta2=letters[1:2],stringsAsFactors=F)
mytext<-DataframeSource(docs)
mycorpus<-VCorpus(mytext)
inspect(mycorpus)
as.character(mycorpus[[1]])
```

Appendix: Codes

Part 2: Web scrapping

```
eg6<-readLines("https://en.wikipedia.org/wiki/Data_science")
eg6[grep("\\h2",eg6)]
eg6[grep("\\p",eg6)] #paragraph

#Using library XML
library(XML)
doc<-htmlParse(eg6)
doc.text<-unlist(xpathApply(doc,'//p',xmlValue))
unlist(xpathApply(doc,'//h2',xmlValue))

#Using library httr
eg7<-GET("https://www.edureka.co/blog/what-is-data-science/")
doc<-htmlParse(eg7)
doc.text<-unlist(xpathApply(doc,'//p',xmlValue))
```

Appendix: Codes

Part 2: Web scrapping

```
#Using library rvest
library(rvest
eg8<-read_html("https://www.edureka.co/blog/what-is-data-science/")
nodes<-html_nodes(eg8,'.color-4a div span , .btn-become-profesional-link+ p')
texts<-html_text(nodes)

#Selecting multiple pages
pages<-
paste0('https://www.amazon.co.jp/s?k=skincare&crid=28HIW1TYLV9UM&sprefix=skincare%2Caps%2C268&r
ef=nb_sb_noss_1&page=',0:9)
eg10<-read_html(pages[1])
nodes<-html_nodes(eg10,'.a-price-whole')
texts<-html_text(nodes)

Price<-function(page){
url<-read_html(page)
nodes<-html_nodes(url ,'.a-price-whole ')
html_text(nodes)}

sapply(pages,Price)
do.call("c",lapply(pages,Price))
```