DATE __/__/____
PAGE NO _____

33304

mycompanion

Assignment - 05

* **Aim :-** Perform different operations using R / Python

* **Problem statement :-** Perform the following operations using R / Python on the Amazon book review and facebook metrics dataset.

* **Objective :-** 1. To learn R / Python programming
2. To learn different data preprocessing Techniques

* **Theory :-**

- **Subsetting Data →**
R has powerful indexing features for accessing object elements. These features can be used to select and exclude variables and observations. The following code snippets demonstrate ways to keep or delete variables and observations and to take random samples of from a dataset

- **Selecting variables →**
```
# select variables v1, v2, v3 : myvars ← c("v1", "v2", "v3")
                                newdata ← mydata [myvars]
```

- **Excluding (dropping variables) →**
```
# exclude variables v1, v2, v3
myvars ← names (mydata) %in% c ("v1", "v2", "v3")
newdata ← mydata [!myvars]
```

- **Selecting observations →**
```
# first 5 observations
newdata ← mydata [1:5]
```

- **Selection using the subset function →**
The subset() function is easiest way to select variables and

observations. In the following example we select all rows that have a value of age greater than or equal to 20 or age less than 10. We keep the ID and weight columns

```
# using subset function
new data ← subset (mydata , age >= 20 | age < 10,
select = c (ID , weight)
```

- Merging data →

① Adding columns :

To merge two data frames ~~simultaneously~~ horizontally, use the merge function. In most cases, you join two data frames by one or more common key variables.

```
# merge two data frames by ID
total ← merge (data frame A , data frame B , by = "ID")
```

② Adding rows :

To join two data frames vertically, use the rbind function. The two dataframes must have same variables, but they do not have to be in same order

```
total ← rbind (data frameA , data frame B)
```

- Sorting data →

To sort a data frame in R, use the order () function. By default, sorting is ASCENDING. Prepend the sorting variable, by a minus to indicate DESCENDING order. Here are some examples.

```
# Sorting examples using mtcars dataset
attach (mtcars)

# sort by mpg
newdata ← mtcars [order (mpg)]
```

- **Transpose →**

Use the t() function to Transpose a matrix or a data frame. In the latter case, row names become variable (column) names.

```
# example using built-in dataset mtcars
t (mtcars)
```

- **The Reshape package →**

Basically you melt data so that each row is a unique id-variable combination. Then you cast the melted data into any shape you would like. Here is a very simple example.

```
# example of melt function
library (reshape)
mdata ← melt (mydata , id = c ("id" , "time"))

# cast melted data
# cast (data , formula , function)
subj means ← cast (mdata , id-variable , mean)
time means ← cast (mdata , id ~ variable , mean)
```

33304

* Conclusion :- Thus we have studied and learnt different data preprocessing techniques.