Experiment1.4

1. Aim:

Demonstration of FP Growth algorithm on SunBai.

2. Objective:

learning about FP Growth algorithm

3. Script and Output:

- The FP-Growth Algorithm is an alternative way to find frequent item sets without using candidate generations, thus improving performance.

- **Confidence:** In data mining, confidence is a measure of the reliability or support for a given association rule.

  Confidence (X => Y) = (Number of transactions containing X and Y) / (Number of transactions containing X)

- **Support**: In data mining, support refers to the relative frequency of an item set in a dataset.

  Support(X)= (Number of transactions containing X) / (Total number of transactions)

Example:

Given data is a hypothetical dataset of transactions with each letter representing an item.

Transaction ID	Items
T1	{E, K, M, N, O, Y}
T2	{D, E, K, N, **O**, Y}
T3	{A, E, K, M}
T4	{C, K, M, U, Y}
T5	{C, E, I, K, O, O}

The frequency of each individual item is computed:

Item	Frequency
A	1
C	2
D	1
E	4
I	1
K	5
M	3
N	2
0	3
U	1
Y	3

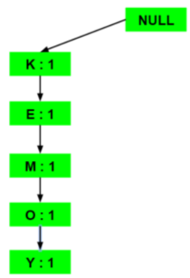
L = {K: 5, E: 4, M: 3, O: 3, Y: 3}

The following table is built for all the transactions:

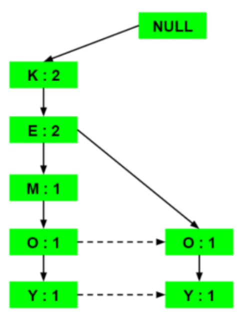
Transaction ID	Items	Ordered-Item Set
T1	{E, K, M, N, O, Y}	{K, E, M, O, Y}
T2	{D, E, K, N, O, Y}	{K, E, O, Y}
T3	{ A, E, K, M}	{K, E, M}
T4	{C, K, M, U, Y}	{K, M, Y}
T5	{C, E, I, K, O, O}	{K, E, O}

Now, all the Ordered-Item sets are inserted into a Trie Data Structure.

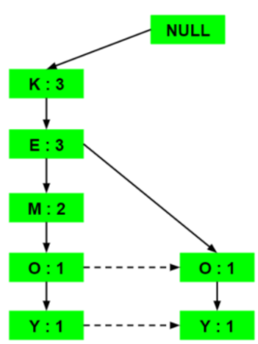
a) Inserting the set {K, E, M, O, Y}:



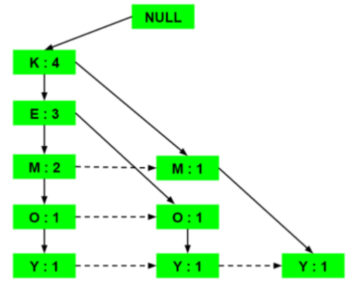
b) Inserting the set {K, E, O, Y}:



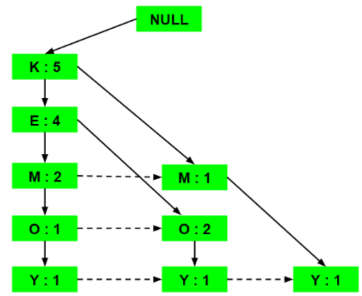
c) Inserting the set {K, E, M}:



d) Inserting the set {K, M, Y}:



e) Inserting the set {K, E, O}:



- Conditional Pattern Base:

Items	Conditional Pattern Base
Y	{{K,E,M,O : 1}, {K,E,O : 1}, {K,M : 1}}
O	{{K,E,M : 1}, {K,E : 2}}
M	{{K,E : 2}, {K : 1}}
E	{K : 4}
K	

- Conditional Frequent Pattern Tree is built:

Items	Conditional Pattern Base	Conditional Frequent Pattern Tree
Y	{{K,E,M,O : 1}, {K,E,O : 1}, {K,M : 1}}	{K : 3}
O	{{K,E,M : 1}, {K,E : 2}}	{K,E : 3}
M	{{K,E : 2}, {K : 1}}	{K : 3}
E	{K : 4}	{K : 4}
K		

- Frequent Pattern rules:

Items	Frequent Pattern Generated
Y	{<K,Y : 3>}
O	{<K,O : 3>, <E,O : 3>, <E,K,O : 3>}
M	{<K,M : 3>}
E	{<E,K : 4>}
K	

CODE AND OUTPUT-

1. **Installing Packages for the given experiment**

```
Source on Save
#Saumyamani Bhardwaz_20BCS1682

#fpgrowth algorithm program

setwd("D:\\info\\F")
intsall.packages("aruleviz")
intsall.packages("arules")
library("arules")
library("aruleviz")
```

2. **Choosing data set for performing the FP Growth algorithm and printing the data set and summary**

```
data("SunBai")
print(SunBai)
summary(SunBai)
```

Output:

```
Console   Terminal ×   Background Jobs ×
R 4.2.2 · D:/info/F/
> data("SunBai")
> print(SunBai)
transactions in sparse format with
 6 transactions (rows) and
 8 items (columns)
> summary(SunBai)
transactions as itemMatrix in sparse format with
 6 rows (elements/itemsets/transactions) and
 8 columns (items) and a density of 0.375

most frequent items:
      A       C       G       B       F (Other)
      4       3       3       2       2       4

element (itemset/transaction) length distribution:
sizes
1 2 3 4 5
1 1 2 1 1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    2.25    3.00    3.00    3.75    5.00

includes extended item information - examples:
  labels
1      A
2      B
3      C

includes extended transaction information - examples:
  transactionID    weight
1           100 0.5176528
2           200 0.4362571
3           300 0.2321374
```

3. Printing FP rules and inpect the data set from 1 to 5 row

```
fprules <- fim4r(SunBai, method = "fpgrowth", target = "rules", supp = 50, conf = 40)

fprules
print(fprules)

inspect(fprules[1:5])
```

Output-

```
> fprules <- fim4r(SunBai, method = "fpgrowth", target = "rules", supp = 50, conf = 40)
> fprules
set of 9 rules
> print(fprules)
set of 9 rules
> inspect(fprules[1:5])
    lhs    rhs support   confidence lift     count
[1] {}  => {C} 0.5000000 0.5000000  1.000000 3
[2] {}  => {A} 0.6666667 0.6666667  1.000000 4
[3] {G} => {C} 0.3333333 0.6666667  1.333333 2
[4] {C} => {G} 0.3333333 0.6666667  1.333333 2
[5] {}  => {G} 0.5000000 0.5000000  1.000000 3
```

4. Storing the SunBai data set to the directory in data farme form

```
# this will convert the values which are inside in fprules in the type of data frame
x <- as(fprules, "data.frame")
print(x)

# this will write the data in csv format
write.csv(x, file="sunbairules.csv")
```

Output-

```
> # this will convert the values which are inside in fprules in the type of data frame
> x <- as(fprules, "data.frame")
> print(x)
       rules   support confidence     lift count
1  {} => {C} 0.5000000  0.5000000 1.000000     3
2  {} => {A} 0.6666667  0.6666667 1.000000     4
3 {G} => {C} 0.3333333  0.6666667 1.333333     2
4 {C} => {G} 0.3333333  0.6666667 1.333333     2
5  {} => {G} 0.5000000  0.5000000 1.000000     3
6 {A} => {B} 0.3333333  0.5000000 1.500000     2
7 {C} => {F} 0.3333333  0.6666667 2.000000     2
8 {G} => {F} 0.3333333  0.6666667 2.000000     2
9 {G} => {H} 0.3333333  0.6666667 2.000000     2
> # this will write the data in csv format
> write.csv(x, file="sunbairules.csv")
```

5. CSV file of the SunBai

	A	B	C	D	E	F	G
1		rules	support	confidenc	lift	count	
2	1	{} => {C}	0.5	0.5	1	3	
3	2	{} => {A}	0.666667	0.666667	1	4	
4	3	{G} => {C}	0.333333	0.666667	1.333333	2	
5	4	{C} => {G}	0.333333	0.666667	1.333333	2	
6	5	{} => {G}	0.5	0.5	1	3	
7	6	{A} => {B}	0.333333	0.5	1.5	2	
8	7	{C} => {F}	0.333333	0.666667	2	2	
9	8	{G} => {F}	0.333333	0.666667	2	2	
10	9	{G} => {H}	0.333333	0.666667	2	2	
11							
12							