Samsung Internship Program

Ritesh Yadav

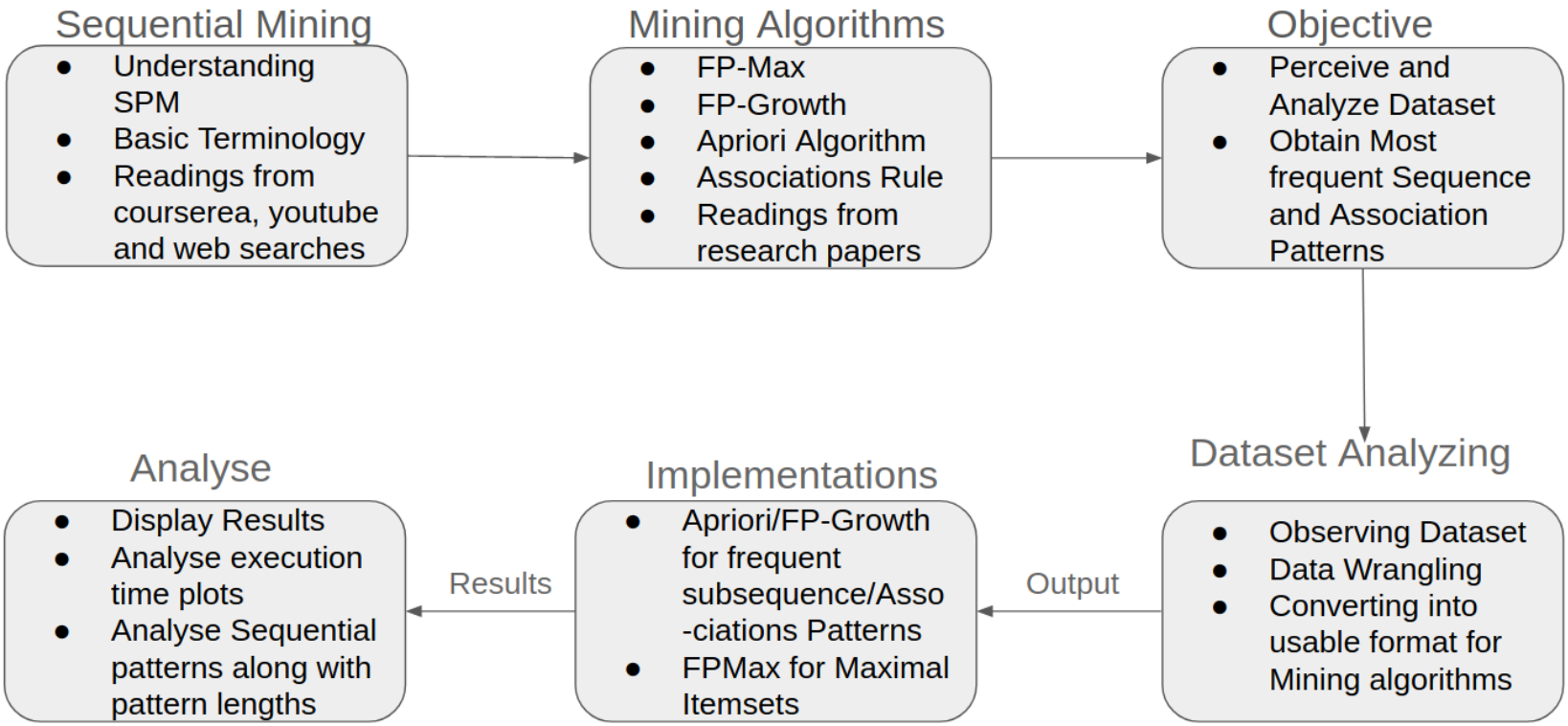
Supervised by: Mr.Ramakant Singh, Mr.Ritesh Dandotia

May-June 2020

Table of Contents

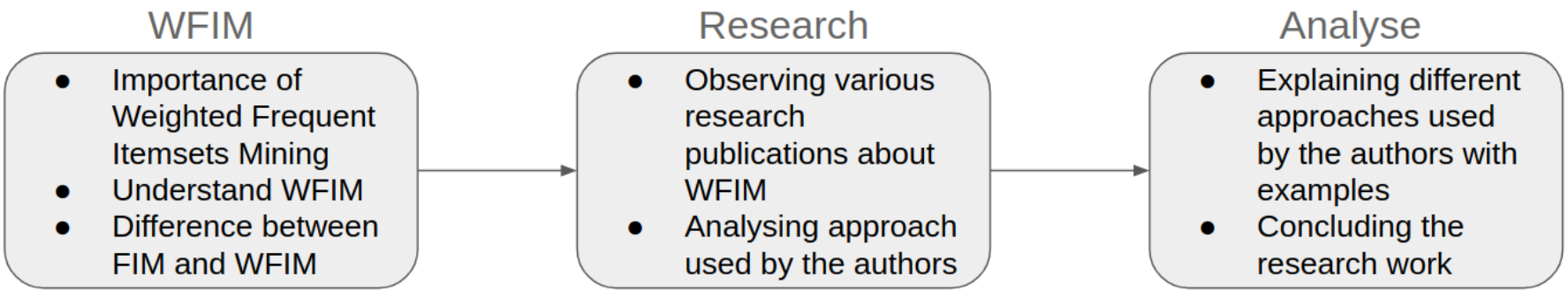
Workflow 1

Sequential Pattern Mining



Workflow 2

Weighted Frequent Itemsets Mining(WFIM)



Sequential Pattern Mining(SPM)

- A sequence pattern consists of sequences of ordered elements or events, recorded with or without a concrete notion of time.
- **Customer shopping sequences** : Purchase a smartphone first, then a sim card, and then a bluetooth earphone,within 6 months.

- It is the mining of frequently occurring ordered events or sub sequences as pattern in sequence database.
- **Sequential Patterns are used for targeted marketing and customer retention.**

Objective

- We are provided by a dataset consisting of screen sequences. Screen sequences are positive integers practically resembling to unique feature screen opened on any electronic device session.
- Screen transitions are denoted by -1. Termination of a session is denoted by -2.
- **Our objective is to perceive dataset by analysing following results:**
  - **Most frequent sequence patterns using different Mining Algorithms**
  - **Most frequent associations pattern**
- Further, research on the possibilities of Weighted frequent Itemsets Mining(WFIM)

Understanding a Sequence Database

A **Sequence Database** :

SID	Sequence
10	<a(abc)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df)cb>
40	<eg(af)cbc>

A **Sequence** : < (ef) (ab) (df) c b >

- The red box in a sequence denotes a single element.
- Alphabets in these elements are called items of that element.
- $< a(ab)d(cf) >$ is a **subsequence** of $< \mathbf{a}(\mathbf{ab}c)(ac)\mathbf{d}(\mathbf{cf}) >$
- For **support thresold** min-sup $= 2$, $< (ab)c >$ is a sequential pattern.

Terminology in Sequential Pattern Mining

- **Support :** Given a pattern A, support of the sequence pattern A is the number of sequences in the database containing the pattern A.
- **Frequent :** An itemset is considered as frequent if it meets a user-specified support threshold.
- **Maximal Itemset :** An itemset is called Maximal if X is frequent and there exists no frequent super-pattern containing X.
- **Minimum Support :** A pattern with support greater than the support threshold.
- Hence given a set of sequences, Sequential Pattern Mining find **the complete set of frequent subsequences which satisfies the min-sup thresold**.
- **Apriori Property :** If a subsequence s is infrequent, none of s's super-sequences can be frequent.

Sequential Pattern Mining Algorithms

- **Association Rule Mining :** Given a set of transactions, we can find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction. **Association Rule** is an implication expression of the form A -> B, where A and B are any 2 itemsets.
- **Apriori :** Apriori algorithm provides us with frequent patterns which satisfy minimum support as provided by user.
- **FP-Growth :** The objective of the FP-Growth algorithm is the same as Apriori. It has emerged as a popular alternative for apriori algorithm due to less computational time over large datasets.
- **FP-Max** : FPMax is variant pf FP-Growth, which helps in obtaining maximal itemsets.

Methodology

- **Data Wrangling :** We will begin by wrangling our dataset by converting it into appropriate usable dataset. In this process we want dataset as multi-dimensional array with rows representing sessions and columns as screens.
- We completed our task by removing -1's and -2's from the dataset and storing the remaining values into a 2-D array.
- Transform dataset into a table format where columns represents screens and row consists of boolean values **True** or **False** depicting presence of a screen in particular session via **Transition Encoder.**
- Apply **FP-Growth** and **Apriori** algorithms for finding most **frequent sub-sequence pattern.**
- Apply **FP-Max** algorithm for **finding Maximal itemsets.**
- Implement **association rules** after apriori for finding **frequent associations pattern.**

Example of SPM

Let us understand all the algorithms with the help of an example. We will mainly focus on FP-Max and Apriori algorithms.

- Let us consider following screen sequences :
  - 2 3 4 1
  - 5 6 1 2
  - 3 1 7 6
  - 2 1 4 5
- **For minimum support = 0.25 :**

FP-Max		
Support	Itemsets	Length
0.25	(1,3,6,7)	4
0.25	(1,2,3,4)	4
0.50	(1,2,4,5)	4
0.50	(1,2,5,6)	4

- **For minimum support = 0.5 :**

Apriori		
Support	Itemsets	Length
1.00	(1)	1
0.75	(2)	1
0.50	(3)	1
0.50	(4)	1
0.50	(5)	1
0.50	(6)	1
0.75	(1,2)	2
0.50	(1,3)	2
0.50	(1,4)	2
0.50	(1,5)	2
0.50	(1,6)	2
0.50	(2,4)	2
0.50	(2,5)	2
0.50	(1,2,4)	3
0.50	(1,2,5)	3

Results And Analysis

- Most frequent Sequence Patterns with min-support = 0.015 :

```
(base) ritesh@ry:~/Desktop/Intern@Samsung$ python fpmax.py
     support        itemsets
0   0.015817         (84731)
1   0.015830         (55335)
2   0.016191         (82719)
3   0.016655         (55831)
4   0.016681         (55343)
5   0.016720         (55367)
6   0.017081        (203729)
7   0.017172         (55875)
8   0.018358         (55551)
9   0.018500         (55543)
10  0.019055         (55287)
11  0.020668         (56769)
12  0.020745         (55387)
13  0.023661         (55295)
14  0.025841         (55291)
15  0.026422         (56761)
16  0.027002         (55283)
17  0.027738         (55327)
18  0.016604  (55323, 55315)
19  0.017546  (55315, 55319)
20  0.029015         (55351)
21  0.029466         (55271)
22  0.015017  (55267, 55319)
23  0.019365  (55323, 55319)
24  0.044083  (55323, 55267)
```

- Most frequent Associations Pattern with min-support = 0.015:

```
(base) ritesh@ry:~/Desktop/Intern@Samsung$ python apriori.py
  antecedents consequents  antecedent support  consequent support   support  confidence       lift  leverage  conviction
0     (55267)     (55319)            0.048586            0.032214  0.015017    0.309081   9.594516  0.013452    1.400723
1     (55319)     (55267)            0.032214            0.048586  0.015017    0.466159   9.594516  0.013452    1.782206
2     (55323)     (55267)            0.044083            0.048586  0.016578    0.376061   7.740104  0.014436    1.524851
3     (55267)     (55323)            0.048586            0.044083  0.016578    0.341211   7.740104  0.014436    1.451020
4     (55315)     (55319)            0.029002            0.032214  0.017546    0.604982  18.779888  0.016611    2.449980
5     (55319)     (55315)            0.032214            0.029002  0.017546    0.544654  18.779888  0.016611    2.132438
6     (55323)     (55315)            0.044083            0.029002  0.016604    0.376646  12.986921  0.015325    1.557700
7     (55315)     (55323)            0.029002            0.044083  0.016604    0.572509  12.986921  0.015325    2.236109
8     (55323)     (55319)            0.044083            0.032214  0.019365    0.439274  13.635972  0.017945    1.725952
9     (55319)     (55323)            0.032214            0.044083  0.019365    0.601121  13.635972  0.017945    2.396510
```

- Most frequent sub-sequence Patterns with min-support = 0.015:

```
(base) ritesh@ry:~/Desktop/Intern@Samsung$ python fpgrowth.py
     support        itemsets
0   0.016655         (55831)
1   0.044083         (55323)
2   0.027738         (55327)
3   0.023661         (55295)
4   0.019055         (55287)
5   0.027002         (55283)
6   0.015817         (84731)
7   0.016681         (55343)
8   0.032214         (55319)
9   0.016720         (55367)
10  0.016191         (82719)
11  0.048586         (55267)
12  0.029466         (55271)
13  0.029015         (55351)
14  0.029002         (55315)
15  0.026422         (56761)
16  0.025841         (55291)
17  0.018358         (55551)
18  0.017172         (55875)
19  0.020668         (56769)
20  0.017081        (203729)
21  0.018500         (55543)
22  0.020745         (55387)
23  0.015830         (55335)
24  0.016578  (55323, 55267)
25  0.015017  (55267, 55319)
26  0.019365  (55323, 55319)
27  0.017546  (55315, 55319)
28  0.016604  (55323, 55315)
```

- Execution Time Results for Minimum Support = 0.015 :

Execution Time Comparison	
Algorithm	Average Time
FPMax	1.21 s
FPGrowth	1.16 s
Apriori	1.43 s
Association Rule	1.56 s

- **For Minimum Support = 0.015 :**
  - FPMax Algorithm :

Pattern Length v/s Count Comparison	
Pattern Length	Count
One	20
Two	5

- FPGrowth Algorithm :

Pattern Length v/s Count Comparison	
Pattern Length	Count
One	24
Two	5

- Apriori Algorithm :

Pattern Length v/s Count Comparison	
Pattern Length	Count
One	24
Two	5

Does all three algorithms produce the same set of patterns ?

No, all three algorithms doesn't generates the same set of patterns. Apriori and FPGrowth algorithms generate the same sets of patterns but pattern generated by FPMax algorithms is different.

Weighted Frequent Itemset Mining(WFIM)

- In recent decades, extensions of FIM such as **weighted frequent itemset mining (WFIM)** and **frequent itemset mining in uncertain databases (UFIM)** have been proposed.
- WFIM considers that items may have distinct weight/importance. It can thus discover itemsets that are more useful and meaningful by ignoring irrelevant itemsets with lower weights.
- In our real world scenarios, the significance (weight) of an item might be widely affected by many factors. Customer's buying behaviors (or interests) are changing with time, so they affect the significances(weights) of products in retail markets.

Research in WFIM

Many research papers and findings related to them for weighted frequent itemsets generation provides different ways of assigning weights to itemsets in transactions:

1. **Efficient Mining of Weighted Quantitative Association Rules and Characterization of Frequent Itemsets** by Arumugam G and Vijayakumar V.K
2. **A Weighted Frequent Itemset Mining Algorithm for Intelligent Decision in Smart Systems** by Xuejian Zhao, Xinhui Zhang, Pan Wang, Songle Chen and Zhixin Sun
3. **Mining Weighted Frequent Itemsets without Candidate Generation in Uncertain Databases** by Chun-Wei Jerry Lin, Wensheng Gan, Philippe Fournier Viger and Tzung-Pei Hong
4. **An Efficient Algorithm for Mining Weighted Frequent Itemsets Using Adaptive Weights** by Hung Long Nguyen
5. **Weighted Frequent Itemset Mining with a weight range and a minimum weight** by Unil Yun and John J. Leggett

Analysis

Research's in [**1,2,3**], suggests assigning distinct weights for each item in the whole dataset and continue with it in the whole dataset of transactions. We can't assign different weights to items in different transactions, the weight assigned must be initialize by the user earlier. They have also consider **uncertain databases** i.e. each item is assigned some uncertainty probability of getting picked in that particular transaction.

- **Example:** Let the total number of items available to user are A, B, C, D.

Item	Weight
A	0.5
B	0.36
C	0.8
D	0.2

Table: Weight Table

Transactions by user :

TID	Transaction(item,probability)
T1	(A, 0.42), (C, 0.31), (D,0.12)
T2	(B, 0.23), (D, 0.56)
T3	(A, 0.48), (B, 0.2), (C, 0.5), (D, 0.35)
T4	(A, 0.82), (B, 0.3)

TABLE: Transaction Table

Here probability represents probability of getting picked.

In [4], they have proceeded correspondingly as above by removing uncertainty probability and adding the option of assigning weights to items batch-wise known as **Adaptive weights.** They have divided the transaction database into k-batches where each batch have distinct-weight items.

- **Example :** We are proceeding here by assigning weights to items batch-wise known as **Adaptive Weights**.We have divided the transaction database into **k-batches** where each batch have distinct-weight items.

Remark

If we put $\mathbf{k = 1}$ we can assign different weights in each transaction.

Let the total number of items available to user are A, B, C, D.

- **Batch-1** :

Item	Weight
A	0.3
B	0.7
C	0.5
D	0.2

TABLE: Weight Table

Transactions by user :

TID	Transaction(item)
T1	B C D
T2	A B D

TABLE: Transaction Table

- **Batch-2 :**

Item	Weight
A	0.6
B	0.1
C	0.6
D	0.3

TABLE: Weight Table

Transaction by User:

TID	Transaction(item)
T3	A B
T4	B A C
T5	A D B

TABLE: Transaction Table

Transactions in above example take place in two different batches.

In [**5**], the author has taken a **Weight Range(WR)** attribute, and weights of the items must lie within this WR in transactions. **Their main approach is to push weight constraints into the pattern growth algorithm and provide ways to keep the downward closure property.**

The End