

MODUL PRAKTIKUM

MATA KULIAH DATA MINING

PERTEMUAN 5

SEMESTER GENAP

TAHUN AJARAN 2024/2025



Disusun oleh:

Dwi Welly Sukma Nirad S.Kom, M.T

Aina Hubby Aziira M.Eng

Rizka Kurnia Illahi

Daffa Agustian Saadi

DEPARTEMEN SISTEM INFORMASI

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

TAHUN 2025

IDENTITAS PRAKTIKUM

IDENTITAS MATA KULIAH

Kode mata kuliah	JSI62122
Nama mata kuliah	Data Mining
CPMK yang dibebankan pada praktikum	CPMK-3, CPMK-4 Mahasiswa mampu menjelaskan teknik asosiasi dalam data mining (CP-2).
Materi Praktikum Pertemuan 5	Konsep Algoritma FP-Growth
	Struktur FP-Tree
	Implementasi Algoritma FP-Growth
	Analisis Pola Asosiasi

IDENTITAS DOSEN DAN ASISTEN MAHASISWA

Nama Dosen Pengampu	<ol style="list-style-type: none"> 1. Dwi Welly Sukma Nirad S.Kom, M.T 2. Aina Hubby Aziira M.Eng
Nama Asisten Mahasiswa (Kelas A)	<ol style="list-style-type: none"> 1. 2211523034 - Muhammad Fariz 2. 2211521012 - Rizka Kurnia Illahi 3. 2211521010 - Dhiya Gustita Aqila 4. 2211522013 - Benni Putra Chaniago 5. 2211521017 - Ghina Anfasha Nurhadi 6. 2211523022 - Daffa Agustian Saadi 7. 2211521007 - Annisa Nurul Hakim 8. 2211522021 - Rifqi Asverian Putra 9. 2211521009 - Miftahul Khaira 10. 2211521015- Nurul Afani 11. 2211523028 - M.Faiz Al-Dzikro
Nama Asisten Mahasiswa (Kelas B)	<ol style="list-style-type: none"> 1. 2211523034 - Muhammad Fariz 2. 2211521012 - Rizka Kurnia Illahi 3. 2211521010 - Dhiya Gustita Aqila 4. 2211522013 - Benni Putra Chaniago 5. 2211521017 - Ghina Anfasha Nurhadi 6. 2211523022 - Daffa Agustian Saadi 7. 2211521007 - Annisa Nurul Hakim 8. 2211522021 - Rifqi Asverian Putra 9. 2211521009 - Miftahul Khaira 10. 2211521015- Nurul Afani 11. 2211523028 - M.Faiz Al-Dzikro

DAFTAR ISI

IDENTITAS PRAKTIKUM	2
IDENTITAS MATA KULIAH	2
IDENTITAS DOSEN DAN ASISTEN MAHASISWA	2
DAFTAR ISI	3
A. Konsep Algoritma FP-Growth	4
B. Struktur FP-Tree	4
C. Implementasi Algoritma FP-Growth.....	5
D. Analisis Pola Asosiasi	12
REFERENSI	13

ALGORITMA FP-GROWTH

A. Konsep Algoritma FP-Growth

Algoritma FP-Growth (Frequent Pattern Growth) adalah salah satu algoritma yang digunakan untuk menemukan pola yang sering muncul dalam kumpulan data besar. Metode ini lebih cepat dan lebih efisien daripada algoritma Apriori karena tidak perlu memindai seluruh basis data secara berulang. Struktur data yang digunakan untuk mencari frequent itemset dengan algoritma FP-growth adalah perluasan dari penggunaan sebuah pohon prefix, yang biasa disebut adalah FP-tree. Dengan menggunakan struktur FP-tree dan berfokus pada set item yang diurutkan, FP-Growth secara efisien menambang frequent itemset, menjadikannya solusi yang lebih cepat dan lebih terukur untuk set data besar.

Cara kerja FP-Growth secara sederhana yaitu :

1) Kompresi Data

Pertama, FP-Growth mengompresi kumpulan data menjadi struktur yang lebih kecil yang disebut Frequent Pattern Tree (FP-Tree) . Pohon ini menyimpan informasi tentang itemset (kumpulan item) dan frekuensinya, tanpa perlu membuat kumpulan kandidat seperti yang dilakukan Apriori.

2) Menambang Pohon

Algoritma kemudian memeriksa pohon ini untuk mengidentifikasi pola yang sering muncul, berdasarkan ambang batas dukungan minimum. Algoritma ini melakukannya dengan memecah pohon menjadi pohon "bersyarat" yang lebih kecil untuk setiap item, sehingga prosesnya menjadi lebih efisien.

3) Menghasilkan Pola

Setelah pohon dibangun dan dianalisis, algoritma menghasilkan pola yang sering muncul (itemset) dan aturan yang menggambarkan hubungan antar item.

B. Struktur FP-Tree

FP-tree merupakan struktur penyimpanan data yang dimampatkan. FP-tree dibangun dengan memetakan setiap data transaksi ke dalam setiap lintasan tertentu dalam FP-tree. Karena dalam setiap transaksi yang dipetakan, mungkin ada transaksi yang memiliki item yang sama, maka

lintasannya memungkinkan untuk saling menimpa. Semakin banyak data transaksi yang memiliki item yang sama, maka proses pemampatan dengan struktur data FP-tree semakin efektif. Kelebihan dari FP-tree adalah hanya memerlukan dua kali pemindaian data transaksi yang terbukti sangat efisien.

FP-Tree dibentuk oleh sebuah akar yang diberi label null, sekumpulan pohon yang beranggotakan item-item tertentu, dan sebuah tabel frequent header. Setiap simpul dalam FP-tree mengandung tiga informasi penting, yaitu:

- a. Label item, menginformasikan jenis item yang direpresentasikan simpul tersebut
- b. Support count, merepresentasikan jumlah lintasan transaksi yang melalui simpul tersebut
- c. Pointer penghubung, menghubungkan simpul-simpul dengan label item sama antar lintasan, ditandai dengan garis panah putus-putus.

C. Implementasi Algoritma FP-Growth

Metode FP-Growth dibagi menjadi tiga tahapan utama, yaitu:

1) Tahap pembangkitan conditional pattern base

Pattern Base merupakan subdatabase yang berisi prefix path (lintasan prefix) dan suffix pattern (pola akhiran). Pembangkitan conditional pattern base didapatkan melalui FP-tree yang telah dibangun sebelumnya.

2) Tahap pembangkitan conditional FP-Tree

Pada tahap ini, support count dari setiap item pada setiap conditional pattern base dijumlahkan, lalu setiap item yang memiliki jumlah support count lebih besar sama dengan minimum support count akan dibangkitkan dengan conditional FPtree.

3) Tahap pencarian frequent itemset

Apabila Conditional FP-tree merupakan lintasan tunggal (single path), maka didapatkan frequent itemset dengan melakukan kombinasi item untuk setiap conditional FP-tree. Jika bukan lintasan tunggal, maka dilakukan pembangkitan FP-Growth secara rekursif.

Contoh Penerapan Algoritma FP-Growth untuk menemukan hubungan antar itemset pada suatu dataset dengan menggunakan dataset “groceries”.

Penyelesaian :

1. Membaca Dataset

```
[3]: import pandas as pd
```

```
df = pd.read_csv('groceries.csv')  
df
```

```
[3]:
```

	Item(s)	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	...	Item 23	Item 24	Item 25	Item 26	Item 27	Item 28	Item 29	Item 30	Item 31
0	4	citrus fruit	semi-finished bread	margarine	ready soups	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	3	tropical fruit	yogurt	coffee	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1	whole milk	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	4	pip fruit	yogurt	cream cheese	meat spreads	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	4	other vegetables	whole milk	condensed milk	long life bakery product	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...
9830	17	sausage	chicken	beef	hamburger meat	citrus fruit	grapes	root vegetables	whole milk	butter	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9831	1	cooking chocolate	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9832	10	chicken	citrus fruit	other vegetables	butter	yogurt	frozen dessert	domestic eggs	rolls/buns	rum	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9833	4	semi-finished bread	bottled water	soda	bottled beer	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9834	5	chicken	tropical fruit	other vegetables	vinegar	shopping bags	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

9835 rows × 33 columns

2. Menampilkan informasi setiap kolom

```
[4]: df.info()  
  
<class 'pandas.core.frame.DataFrame'  
RangeIndex: 9835 entries, 0 to 9834  
Data columns (total 33 columns):  
#   Column      Non-Null Count  Dtype  
---  ---  
0   Item(s)     9835 non-null   int64  
1   Item 1      9835 non-null   object  
2   Item 2      7676 non-null   object  
3   Item 3      6033 non-null   object  
4   Item 4      4734 non-null   object  
5   Item 5      3729 non-null   object  
6   Item 6      2874 non-null   object  
7   Item 7      2229 non-null   object  
8   Item 8      1684 non-null   object  
9   Item 9      1246 non-null   object  
10  Item 10     896 non-null    object  
11  Item 11     650 non-null    object  
12  Item 12     468 non-null    object  
13  Item 13     351 non-null    object  
14  Item 14     273 non-null    object  
15  Item 15     196 non-null    object  
16  Item 16     141 non-null    object  
17  Item 17     95 non-null     object  
18  Item 18     66 non-null     object  
19  Item 19     52 non-null     object  
20  Item 20     38 non-null     object  
21  Item 21     29 non-null     object  
22  Item 22     18 non-null     object  
23  Item 23     14 non-null     object  
24  Item 24     8 non-null      object  
25  Item 25     7 non-null      object  
26  Item 26     7 non-null      object  
27  Item 27     6 non-null      object  
28  Item 28     5 non-null      object  
29  Item 29     4 non-null      object  
30  Item 30     1 non-null      object  
31  Item 31     1 non-null      object  
32  Item 32     1 non-null      object  
dtypes: int64(1), object(32)  
memory usage: 2.5+ MB
```

3. Membuat dataframe baru Bernama df2 dan menghapus kolom item(s)

```
[5]: df2 = df.drop('Item(s)', axis=1)
df2
```

```
[5]:
```

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10	...	Item 23	Item 24	Item 25	Item 26	Item 27	Item 28	Item 29	Item 30	Item 31
0	citrus fruit	semi-finished bread	margarine	ready soups	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	tropical fruit	yogurt	coffee	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	whole milk	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	pip fruit	yogurt	cream cheese	meat spreads	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	other vegetables	whole milk	condensed milk	long life bakery product	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...
9830	sausage	chicken	beef	hamburger meat	citrus fruit	grapes	root vegetables	whole milk	butter	whipped/sour cream	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9831	cooking chocolate	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9832	chicken	citrus fruit	other vegetables	butter	yogurt	frozen dessert	domestic eggs	rolls/buns	rum	cling film/bags	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9833	semi-finished bread	bottled water	soda	bottled beer	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9834	chicken	tropical fruit	other vegetables	vinegar	shopping bags	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

9835 rows × 32 columns

4. Memeriksa adanya nilai null pada data

```
[6]: df2.isnull().sum()
```

```
[6]:
```

Item 1	0
Item 2	2159
Item 3	3802
Item 4	5101
Item 5	6106
Item 6	6961
Item 7	7606
Item 8	8151
Item 9	8589
Item 10	8939
Item 11	9185
Item 12	9367
Item 13	9484
Item 14	9562
Item 15	9639
Item 16	9694
Item 17	9740
Item 18	9769
Item 19	9783
Item 20	9797
Item 21	9806
Item 22	9817
Item 23	9821
Item 24	9827
Item 25	9828
Item 26	9828
Item 27	9829
Item 28	9830
Item 29	9831
Item 30	9834
Item 31	9834
Item 32	9834

dtype: int64

5. Mengubah DataFrame (df2) menjadi list, sambil menghapus nilai NaN dan elemen kosong.

```
[7]: array = df2.fillna('').values.tolist()
array = [[x for x in row if x] for row in array]
array

[7]: [['citrus fruit', 'semi-finished bread', 'margarine', 'ready soups'],
      ['tropical fruit', 'yogurt', 'coffee'],
      ['whole milk'],
      ['pip fruit', 'yogurt', 'cream cheese', 'meat spreads'],
      ['other vegetables',
       'whole milk',
       'condensed milk',
       'long life bakery product'],
      ['whole milk', 'butter', 'yogurt', 'rice', 'abrasive cleaner'],
      ['rolls/buns'],
      ['other vegetables',
       'UHT-milk',
       'rolls/buns',
       'bottled beer',
       'liquor (appetizer)'],
      ['potted plants'],
      ['whole milk', 'cereals'],
      ['tropical fruit']]
```

6. Import library yang diperlukan

- Mengubah daftar transaksi ke dalam format one-hot encoding (True/False) agar bisa diproses oleh FP-Growth.
- Algoritma FP-Growth yang digunakan untuk menemukan **frequent itemsets** (kombinasi barang yang sering muncul bersama).
- Digunakan untuk menghitung aturan asosiasi berdasarkan hasil dari FP-Growth.

```
[9]: from mlxtend.preprocessing import TransactionEncoder
      from mlxtend.frequent_patterns.fpgrowth import fpgrowth
      from mlxtend.frequent_patterns import association_rules
```

7. Membuat objek TransactionEncoder untuk mengonversi data transaksi ke dalam format one-hot encoding. One-hot encoding adalah format di mana setiap item dalam transaksi direpresentasikan sebagai **True (1) atau False (0)** dalam bentuk tabel.

```
[10]: transaksi_encoder = TransactionEncoder()
```

8. Menentukan daftar item unik dari transaksi dan Mengonversi transaksi menjadi format one-hot encoding (array dengan True/False).


```
[11]: transaksi_encoder.fit(array)
      encoded_transaksi = transaksi_encoder.transform(array)
```

9. Mengubah hasil one-hot encoding menjadi DataFrame Pandas agar lebih mudah dibaca dan dianalisis.

```
[12]: encoded_transaksi_df = pd.DataFrame(encoded_transaksi, columns = transaksi_encoder.columns_)
```

10. Mendefinisikan daftar nilai minimum support

```
[13]: min_support_range = [0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5]
```

11. Melakukan iterasi (looping) melalui berbagai nilai minimum support untuk menjalankan algoritma FP-Growth dan menampilkan berapa banyak frequent itemsets

```
[14]: for min_support in min_support_range:
      freq_itemsets = fpgrowth(encoded_transaksi_df, min_support=min_support, use_colnames=True)
      print(f"Minimum Support: {min_support}")
      print(f"Frequent Itemsets: {len(freq_itemsets)}")
      print(freq_itemsets)
      print()
```

```
Minimum Support: 0.01
Frequent Itemsets: 333
  support      itemsets
0  0.082766  (citrus fruit)
1  0.058566  (margarine)
2  0.017692  (semi-finished bread)
3  0.139502  (yogurt)
4  0.104931  (tropical fruit)
..      ...
328 0.010168  (rolls/buns, frozen vegetables)
329 0.012405  (yogurt, frozen vegetables)
330 0.014235  (other vegetables, onions)
331 0.012100  (whole milk, onions)
332 0.010778  (sliced cheese, whole milk)
```

```
[333 rows x 2 columns]
```

```
Minimum Support: 0.05
```

12. Menetapkan minimum support = 0.05

```
[15]: min_support = 0.05
```

13. Mencari **frequent itemsets** dalam dataset transaksi yang telah dikodekan ke dalam bentuk dataframe.

```
[16]: freq_itemsets = fpgrowth(encoded_transaksi_df, min_support=min_support, use_colnames=True)
      freq_itemsets
```

```
[16]:
```

	support	itemsets
0	0.082766	(citrus fruit)
1	0.058566	(margarine)
2	0.139502	(yogurt)
3	0.104931	(tropical fruit)
4	0.058058	(coffee)
5	0.255516	(whole milk)
6	0.075648	(pip fruit)
7	0.193493	(other vegetables)
8	0.055414	(butter)
9	0.183935	(rolls/buns)
10	0.080529	(bottled beer)
11	0.110524	(bottled water)
12	0.053279	(curd)
13	0.052466	(beef)
14	0.174377	(soda)
15	0.058973	(frankfurter)
16	0.079817	(newspapers)
17	0.072293	(fruit/vegetable juice)
18	0.088968	(pastry)
19	0.108998	(root vegetables)
20	0.077682	(canned beer)
21	0.093950	(sausage)
22	0.098526	(shopping bags)
23	0.064870	(brown bread)
24	0.052364	(napkins)
25	0.071683	(whipped/sour cream)
26	0.057651	(pork)
27	0.063447	(domestic eggs)
28	0.056024	(whole milk, yogurt)
29	0.074835	(other vegetables, whole milk)
30	0.056634	(rolls/buns, whole milk)

14. Menggunakan fungsi `association_rules` dari pustaka `mlxtend` untuk membentuk aturan asosiasi berdasarkan frequent itemsets yang telah ditemukan sebelumnya dengan FP-Growth.

```
[17]: from mlxtend.frequent_patterns import association_rules

# Melakukan proses pembentukan aturan asosiasi sebelumnya
association_rules_result = association_rules(freq_itemsets, metric="lift", min_threshold=0.06)

# Menampilkan hasil aturan asosiasi
print(association_rules_result)
```

	antecedents	consequents	antecedent support \
0	(whole milk)	(yogurt)	0.255516
1	(yogurt)	(whole milk)	0.139502
2	(other vegetables)	(whole milk)	0.193493
3	(whole milk)	(other vegetables)	0.255516
4	(rolls/buns)	(whole milk)	0.183935
5	(whole milk)	(rolls/buns)	0.255516

	consequent support	support	confidence	lift	leverage	conviction \
0	0.139502	0.056024	0.219260	1.571735	0.020379	1.102157
1	0.255516	0.056024	0.401603	1.571735	0.020379	1.244132
2	0.255516	0.074835	0.386758	1.513634	0.025394	1.214013
3	0.193493	0.074835	0.292877	1.513634	0.025394	1.140548
4	0.255516	0.056634	0.307905	1.205032	0.009636	1.075696
5	0.183935	0.056634	0.221647	1.205032	0.009636	1.048452

	zhangs_metric
0	0.488608
1	0.422732
2	0.420750
3	0.455803
4	0.208496
5	0.228543

15. Mengurutkan aturan asosiasi berdasarkan nilai confidence dari yang tertinggi ke yang terendah.

```
[18]: sort = association_rules_result.sort_values(['confidence'], ascending=False)
sort
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
1	(yogurt)	(whole milk)	0.139502	0.255516	0.056024	0.401603	1.571735	0.020379	1.244132	0.422732
2	(other vegetables)	(whole milk)	0.193493	0.255516	0.074835	0.386758	1.513634	0.025394	1.214013	0.420750
4	(rolls/buns)	(whole milk)	0.183935	0.255516	0.056634	0.307905	1.205032	0.009636	1.075696	0.208496
3	(whole milk)	(other vegetables)	0.255516	0.193493	0.074835	0.292877	1.513634	0.025394	1.140548	0.455803
5	(whole milk)	(rolls/buns)	0.255516	0.183935	0.056634	0.221647	1.205032	0.009636	1.048452	0.228543
0	(whole milk)	(yogurt)	0.255516	0.139502	0.056024	0.219260	1.571735	0.020379	1.102157	0.488608

D. Analisis Pola Asosiasi

Pada penerapan algoritma FP-Growth ini terdapat parameter penting yang diperlukan untuk pembentukan rules dalam penerapan algoritma FP- Growth. Hal ini dikarenakan FP-Growth bertujuan untuk mencari keterkaitan antara item berdasarkan jumlah item yang muncul dari *association rule* yang terdapat dalam transaksi.

Association rule merupakan suatu proses pada data mining untuk menentukan semua aturan asosiatif yang memenuhi syarat minimum untuk *support* (minsup) dan *confidence* (minconf) pada sebuah database. Kedua syarat tersebut akan digunakan untuk *association rules* dengan dibandingkan dengan batasan yang telah ditentukan, yaitu minsup dan minconf.

Setelah itemset yang sering muncul ditemukan, langkah selanjutnya adalah membentuk aturan asosiasi untuk mengidentifikasi hubungan antar item. Aturan asosiasi dievaluasi berdasarkan dua parameter, yaitu: Support (nilai penunjang), Confidence (nilai kepastian).

a. Support

Support dari suatu association rule adalah presentasi kombinasi item tersebut, dimana jika mempunyai item A dan item B maka support adalah proporsi dari transaksi yang mengandung A dan B. Rumus untuk menghitung nilai support dari dua item tersebut adalah sebagai berikut:

$$\text{Support (A,B)} = P(A \cap B) = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Total transaksi}}$$

b. Confidence

Confidence dari association rule adalah ukuran ketepatan suatu rule, yaitu presentasi transaksi yang mengandung A dan mengandung B. Dengan adanya confidence dapat diukur kuatnya hubungan antar-item dalam association rule. Rumus untuk menghitung nilai confidence dari dua item tersebut adalah sebagai berikut:

$$\text{Confidence(A} \rightarrow \text{B)} = P(A|B) = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Jumlah transaksi yang mengandung A}}$$

REFERENSI

- GeeksforGeeks. (2025, February 2). *Frequent pattern growth algorithm*.
<https://www.geeksforgeeks.org/frequent-pattern-growth-algorithm/> (diakses pada 15
Maret 2025).
- Rihastuti, S., & Rosyidi, A. (2023). Penerapan Algoritma FP-growth UNTUK ANALISA POLA
Belanja Konsumen Toko Daffamart. *Jurnal Teknologi Informasi*, 9(2), 126–131.
<https://doi.org/10.52643/jti.v9i2.3661>