LAPORAN TUGAS 4 KELOMPOK 6

PENAMBANGAN DATA

Laporan ini dibuat untuk memenuhi tugas 4 mata kuliah Penambangan Data

Disusun oleh :

Edgar Vigo 1301180149

Muhammad Raihan Muhith 1301184245

M Wanda Wibisono 1301184339

M Tegar Zharfan H. S 1301184354

M Dwiantara Mahardika 1301184467

Telkom University

Bandung

2021

A. Dataset Abscisic Acid Signaling Network Data set

1. Memahami Dataset

Pada penugasan asosiasi ini diberikan dua dataset yang diperlukan untuk eksplorasi dan dicari rulenya menggunakan metode analisis Asosiasi, dataset yang pertama adalah data set Abscisic Acid Signaling Network Dataset (**http://archive.ics.uci.edu/ml/datasets/Abscisic+Acid+Signaling+Network**)

Abscisic Acid Signaling Network Data Set

Download: Data Folder, Data Set Description

Abstract: The objective is to determine the set of boolean rules that describe the interactions of the nodes within this plant sigr asynchronous update scheme.

Data Set Characteristics:	Multivariate	**Number of Instances:**	300	**Area:**	Life
Attribute Characteristics:	Integer	**Number of Attributes:**	43	**Date Donated**	2008-04-03
Associated Tasks:	Causal-Discovery	**Missing Values?**	N/A	**Number of Web Hits:**	62347

Dataset Abscisic Acid Signaling data ini berisi sebanyak 300 data dengan 43 atribut yang didalamnya bernilai nilai 1 atau 0, data ini data ini memiliki karakteristik disetiap atributnya bernilai integer dengan jenis multivariat dengan gambaran pada dataset akan bernilai seperti berikut ini :

```
1011101110101101101101001010001011000011001
1100001110111101101101111111011001011101011
1100011110111110101101100011010001110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
1100001110111110101101100011000011110101010
```

data yang dihasilkan dari signalling akan membentuk sebuah matriks dengan besar 21x43, dan pada dataset ini tidak memiliki nilai missing value, dan kemungkinan perlu dilakukan sampling dari dataset untuk digunakan pada proses asosiasi, sampling ini kemungkinan akan menggunakan 1 set data.

2. Preprocessing Data

Dataset ini tidak memiliki missing values atau wrong data dikarenakan isi datanya hanya berupa angka 0 dan 1.

```
df = pd.read_csv('plantCellSignaling.csv',sep=';',header=0,index_col=None)
df
```

	ABA	CLOSURE	Ca	CaATPase	CAIM	CIS	ABH1	GCR	ERA1	PEPC	...	GPA	ROS	AnionEM	ABI	DEPOLAR	HATPase	KOUT	KAP	Actin	NO
0	1.0	0.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0
1	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	0.0	...	1.0	0.0	1.0	1.0	1.0	0.0	1.0	0.0	1.0	1.0
2	1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
3	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
4	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5706	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0
5707	1.0	1.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0
5708	1.0	1.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0
5709	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0
5710	1.0	1.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0

5711 rows × 43 columns

```
df.isna().sum()

ABA         0
CLOSURE     0
Ca          0
CaATPase    0
CAIM        0
CIS         0
ABH1        0
GCR         0
ERA1        0
PEPC        0
OST         0
SPHK        0
PH          0
PLD         0
ROP2        0
KEV         0
AGB         0
RAC         0
RCN         0
NIA12       0
PLC         0
InsPK       0
IP6         0
ADPRc       0
GC          0
NOS         0
S1P         0
PA          0
IP3         0
CADPR       0
cGMP        0
MALATE      0
ATRBOH      0
GPA         0
ROS         0
AnionEM     0
ABI         0
DEPOLAR     0
HATPase     0
KOUT        0
KAP         0
Actin       0
NO          0
dtype: int64
```

Untuk mempermudah memahami isi datasetnya maka kami melakukan penambahan string pada setiap isi kolomnya sesuai dengan nama attribut' misal pada baris kolom Attribut ABA yang sebelumnya berisi '1.0' maka akan kami ubah menjadi 'ABA_1', begitu seterusnya untuk keseluruhan data.

```
for x in col:
    df[x] = df[x].replace([0,1],[x+'_0',x+'_1'])

df
```

	ABA	CLOSURE	Ca	CaATPase	CAIM	CIS	ABH1	GCR	ERA1	PEPC	...	GPA	ROS	AnionEM	ABI	DEPOLAR	HATPase	KOUT	KAP	Actin	NO
0	ABA_1	CLOSURE_0	Ca_1	CaATPase_1	CAIM_1	CIS_0	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_0	AnionEM_0	ABI_0	DEPOLAR_0	HATPase_1	KOUT_1	KAP_0	Actin_0	NO_1
1	ABA_1	CLOSURE_1	Ca_0	CaATPase_0	CAIM_0	CIS_0	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_0	AnionEM_1	ABI_1	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_1
2	ABA_1	CLOSURE_1	Ca_0	CaATPase_0	CAIM_0	CIS_1	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_0
3	ABA_1	CLOSURE_1	Ca_0	CaATPase_0	CAIM_0	CIS_0	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_0
4	ABA_1	CLOSURE_1	Ca_0	CaATPase_0	CAIM_0	CIS_0	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5706	ABA_1	CLOSURE_1	Ca_1	CaATPase_1	CAIM_0	CIS_1	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_1
5707	ABA_1	CLOSURE_1	Ca_0	CaATPase_1	CAIM_0	CIS_1	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_1
5708	ABA_1	CLOSURE_1	Ca_1	CaATPase_0	CAIM_0	CIS_1	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_1
5709	ABA_1	CLOSURE_1	Ca_1	CaATPase_1	CAIM_0	CIS_1	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_1
5710	ABA_1	CLOSURE_1	Ca_0	CaATPase_1	CAIM_0	CIS_1	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_1

5456 rows × 43 columns

3. Proses Asosiasi

Kami menggunakan **Algoritma Apriori**, untuk implementasi proses akan menggunakan bantuan library apyori pada python sehingga dapat terimplementasikan,

> **Algoritma apriori** merupakan salah satu algoritma klasik data mining**.** Algoritma apriori digunakan agar komputer dapat mempelajari aturan asosiasi, mencari pola hubungan antar satu atau lebih item dalam suatu dataset. Penting tidaknya suatu aturan asosiatif dapat diketahui dengan dua parameter, **support (nilai penunjang)** yaitu persentase kombinasi item tersebut dalam database dan **confidence (nilai kepastian)** yaitu kuatnya hubungan antar item dalam aturan asosiatif.
>
> aturan asosiatif biasanya dinyatakan dalam bentuk : {roti, mentega} -> {susu} (support = 40%, confidence = 50%)
>
> Yang artinya : "Seorang konsumen yang membeli roti dan mentega punya kemungkinan 50% untuk juga membeli susu. Aturan ini cukup signifikan karena mewakili 40% dari catatan transaksi selama ini."
>
> Analisis asosiasi didefinisikan suatu proses untuk menemukan semua aturan asosiatif yang memenuhi syarat minimum untuk support (minimum support) dan syarat minimum untuk confidence (minimum confidence).
>
> Tetapi di lain pihak Apriori memiliki kelemahan karena harus melakukan scan database setiap kali iterasi, sehingga waktu yang diperlukan bertambah dengan semakin banyak iterasi. Masalah ini yang dipecahkan oleh algoritma-algoritma baru seperti FP-growth**.**

Implementasi Python :

-Import Library yang dibutuhkan

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from apyori import apriori
```

- Load data untuk mengetahui sehingga bisa diketahui uniqueness, count data, top value, dan frequency dari setiap atribut yang ada pada dataset, kemudian kami buat rule menggunakan apriori dengan parameter yang perlu di perhatikan adalah min_support, min_confidence dan max_lengthnya.

```
[ ] df.describe(include='all')
```

	ABA	CLOSURE	Ca	CaATPase	CAIM	CIS	ABH1	GCR	ERA1	PEPC	...	GPA	ROS	AnionEM	ABI	DEPOLAR	HATPase	KOUT	KAP	Actin	NO
count	5456	5456	5456	5456	5456	5456	5456	5456	5456	5456	...	5456	5456	5456	5456	5456	5456	5456	5456	5456	5456
unique	1	2	2	2	2	2	2	2	2	2	...	2	2	2	2	2	2	2	2	2	2
top	ABA_1	CLOSURE_1	Ca_0	CaATPase_0	CAIM_0	CIS_0	ABH1_1	GCR_1	ERA1_1	PEPC_0	...	GPA_1	ROS_1	AnionEM_1	ABI_0	DEPOLAR_1	HATPase_0	KOUT_1	KAP_0	Actin_1	NO_0
freq	5456	4728	4425	4353	5291	3813	5315	5320	5322	5339	...	5204	4950	5108	5075	5303	5298	5234	5302	5289	4333

4 rows × 43 columns

- Kami menggunakan asosiasi dengan nilai minimum support 0.04 dan nilai minimum confidence sebanyak 0.8 karena dataset berisi biner dengan ,rata-rata nilai unique dibawah 2, dan max_length nya 2.

```
records = []
for i in range(0,df.shape[0]):
    records.append([str(df.values[i,j]) for j in range(0, 43)])
rules = apriori(records, min_support=0.04, min_confidence = 0.8, max_length=2, target='rules')
association_rules = list(rules)
```

- Dari proses tersebut dihasilkan rules sejumlah 1400

```
[ ] print('Terdapat sebanyak = ',i-1,' rules')

    Terdapat sebanyak =  1400  rules
```

- Berikut rules yang dihasilkan

```
Rule: ABH1_1 -> ABA_1
Support: 0.9741568914956011
Confidence: 0.9741568914956011
Lift: 1.0
=====================================
Rule: ABI_0 -> ABA_1
Support: 0.9301686217008798
Confidence: 0.9301686217008798
Lift: 1.0
=====================================
Rule: ABI_1 -> ABA_1
Support: 0.06983137829912023
Confidence: 1.0
Lift: 1.0
=====================================
Rule: ADPRc_0 -> ABA_1
Support: 0.781341642228739
Confidence: 1.0
Lift: 1.0
=====================================
Rule: ADPRc_1 -> ABA_1
Support: 0.218658357771261
Confidence: 1.0
Lift: 1.0
=====================================
…
Rule: cADPR_0 -> cGMP_0
Support: 0.7060117302052786
Confidence: 0.9149643705463184
Lift: 1.1837907530710725
=====================================
```

- Max Support = 1.0

```
support = []
for item in association_rules:
    support.append(item[1])
max_supp = max(support)
print('Max Support : ', max_supp)
```

```
Max Support :  1.0
```

- Max Confidence = 1.0

```
conf = []
for item in association_rules:
    conf.append(item[2][0][2])
max_conf = max(conf)
print('Max Confidence : ', max_conf)
```

```
Max Confidence :  1.0
```

- Max Lift = 6.676139952002021

```
lift = []
for item in association_rules:
    lift.append(item[2][0][3])
max_lift = max(lift)
print('Max Lift : ', max_lift)
```

```
Max Lift :  6.676139952002021
```

- Rules dengan Best Support

```
print("RULES DENGAN BEST SUPPORT \n")

for item in association_rules:

    # first index of the inner list
    # Contains base item and add item
    pair = item[0]
    items = [x for x in pair]
    if item[1] == max_supp:
        if len(items) > 1:
            print("Rule: " + items[0] + " -> " + items[1])

            #second index of the inner list
            print("Support: " + str(item[1]))

            #third index of the list located at 0th
            #of the third index of the inner list

            print("Confidence: " + str(item[2][0][2]))
            print("Lift: " + str(item[2][0][3]))
            print("=====================================")
```

```
RULES DENGAN BEST SUPPORT

Rule: AGB_1 -> ABA_1
Support: 1.0
Confidence: 1.0
Lift: 1.0
=====================================
```

- Rules dengan Best Confidence

```
for item in association_rules:

    # first index of the inner list
    # Contains base item and add item
    pair = item[0]
    items = [x for x in pair]
    if item[2][0][2] == max_conf:
        if len(items) > 1:
            print("Rule: " + items[0] + " -> " + items[1])

            #second index of the inner list
            print("Support: " + str(item[1]))

            #third index of the list located at 0th
            #of the third index of the inner list

            print("Confidence: " + str(item[2][0][2]))
            print("Lift: " + str(item[2][0][3]))
            print("=====================================")
```

```
RULES DENGAN BEST CONFIDENCE

Rule: ABI_1 -> ABA_1
Support: 0.06983137829912023
Confidence: 1.0
Lift: 1.0
=====================================
Rule: ADPRc_0 -> ABA_1
Support: 0.781341642228739
Confidence: 1.0
Lift: 1.0
=====================================
Rule: ADPRc_1 -> ABA_1
```

- Rules dengan Best Lift

```
print('RULES DENGAN BEST LIFT \n')

for item in association_rules:

    # first index of the inner list
    # Contains base item and add item
    pair = item[0]
    items = [x for x in pair]
    if item[2][0][3] == max_lift:
        if len(items) > 1:
            print("Rule: " + items[0] + " -> " + items[1])

            #second index of the inner list
            print("Support: " + str(item[1]))

            #third index of the list located at 0th
            #of the third index of the inner list

            print("Confidence: " + str(item[2][0][2]))
            print("Lift: " + str(item[2][0][3]))
            print("=====================================")
```

```
RULES DENGAN BEST LIFT

Rule: AnionEM_0 -> CLOSURE_0
Support: 0.05681818181818816
Confidence: 0.8908045977011494
Lift: 6.676139952002021
=====================================
```

B. Dataset Labor Relation

1. Memahami Dataset

Pada penugasan asosiasi ini diberikan dua dataset yang diperlukan untuk eksplorasi dan dicari rulenya menggunakan metode analisis Asosiasi, dataset yang kedua adalah data set Dataset Labor Relations.

Data Set Characteristics:	Multivariate	Number of Instances:	57	Area:	Social
Attribute Characteristics:	Categorical, Integer, Real	Number of Attributes:	16	Date Donated	1988-11-01
Associated Tasks:	N/A	Missing Values?	No	Number of Web Hits:	91189

Dataset Labor Relations ini merupakan dataset yang berisi 16 atribut dengan 1 class, dataset ini memiliki jumlah data sebanyak 57 data sehingga untuk penanganannya perlu memikirkan lagi jika perlu menghapusnya karena data jumlahnya sudah sedikit kemudian data ini berkaitan dengan sosial, dan memiliki atribut yang bersifat kategorik dan numerik, untuk atributnya sendiri terdiri dari berikut ini.

```
1. dur: duration of agreement[1..7]
2 wage1.wage : wage increase in first year of contract[2.0 .. 7.0]
3 wage2.wage : wage increase in second year of contract[2.0 .. 7.0]
4 wage3.wage : wage increase in third year of contract[2.0 .. 7.0]
5 cola : cost of living allowance[none, tcf, tc]
6 hours.hrs : number of working hours during week[35 .. 40]
7 pension : employer contributions to pension plan[none, ret_allw, empl_contr]
8 stby_pay : standby pay[2 .. 25]
9 shift_diff : shift differencial : supplement for work on II and III shift[1 .. 25]
10 educ_allw.boolean : education allowance[true false]
11 holidays : number of statutory holidays[9 .. 15]
12 vacation : number of paid vacation days[ba, avg, gnr]
13 lngtrm_disabil.boolean : employer's help during employee longterm disability[true , false]
14 dntl_ins : employers contribution towards the dental plan[none, half, full]
15 bereavement.boolean : employer's financial contribution towards the covering the costs of bereavement[true , fal
16 empl_hplan : employer's contribution towards the health plan[none, half, full]
```

2. Preprocessing Data

pada keterangan data tidak mengandung missing value, namun setelah diimport datanya memang tidak terkandung missing value namun mengandung banyak wrong value dan “NaN” sehingga perlu dilakukan penanganan,

	dur	wage1	wage2	wage3	cola	hours	pension	stby_pay	shift_diff	educ_allw	holidays	vacation	lngtrm_disabil	dntl_ins	bereavement	empl_plan	good/bad
0	1.0	5.0	NaN	NaN	NaN	40.0	NaN	NaN	2.0	NaN	11.0	average	NaN	NaN	yes	NaN	good
1	2.0	4.5	5.8	NaN	NaN	35.0	ret_allw	NaN	NaN	yes	11.0	below average	NaN	full	NaN	full	good
2	NaN	NaN	NaN	NaN	NaN	38.0	empl_contr	NaN	5.0	NaN	11.0	generous	yes	half	yes	half	good
3	3.0	3.7	4.0	5.0	tc	NaN	NaN	NaN	NaN	yes	NaN	NaN	NaN	NaN	yes	NaN	good
4	3.0	4.5	4.5	5.0	NaN	40.0	NaN	NaN	NaN	NaN	12.0	average	NaN	half	yes	half	good
5	2.0	2.0	2.5	NaN	NaN	35.0	NaN	NaN	6.0	yes	12.0	average	NaN	NaN	NaN	NaN	good
6	3.0	4.0	5.0	5.0	tc	NaN	empl_contr	NaN	NaN	NaN	12.0	generous	yes	none	yes	half	good
7	3.0	6.9	4.8	2.3	NaN	40.0	NaN	NaN	3.0	NaN	12.0	below average	NaN	NaN	NaN	NaN	good
8	2.0	3.0	7.0	NaN	NaN	38.0	NaN	12.0	25.0	yes	11.0	below average	yes	half	yes	NaN	good
9	1.0	5.7	NaN	NaN	none	40.0	empl_contr	NaN	4.0	NaN	11.0	generous	yes	full	NaN	NaN	good
10	3.0	3.5	4.0	4.6	none	36.0	NaN	NaN	3.0	NaN	13.0	generous	NaN	NaN	yes	full	good

Seperti yang sudah di tahapan sebelumnya diketahui dataset ini memiliki jumlah data yang bisa dikategorikan sangat minim, oleh karena itu untuk penanganan yang dilakukan ada baiknya untuk melakukan imputasi dibandingkan melakukan penghapusan variabel, dari hasil eksplorasi ditemukan data berikut ini.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 17 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   dur             39 non-null     float64
 1   wage1           39 non-null     float64
 2   wage2           30 non-null     float64
 3   wage3           12 non-null     float64
 4   cola            24 non-null     object 
 5   hours           37 non-null     float64
 6   pension         18 non-null     object 
 7   stby_pay        7 non-null      float64
 8   shift_diff      24 non-null     float64
 9   educ_allw       18 non-null     object 
 10  holidays        38 non-null     float64
 11  vacation        37 non-null     object 
 12  lngtrm_disabil  16 non-null     object 
 13  dntl_ins        25 non-null     object 
 14  bereavement     20 non-null     object 
 15  empl_plan       24 non-null     object 
 16  good/bad        40 non-null     object 
dtypes: float64(8), object(9)
memory usage: 5.4+ KB
```

Perlu digaris bawahi bahwa nilai null ini merupakan perubahan dari “?” menjadi NaN, karena NaN memiliki makna yang ambigu sehingga bisa dikategorikan sebagai nilai yang missing atau hilang, sehingga perlu dilakukan penanganan Imputasi nilai Mean dan Mode untuk atribut yang bernilai numerik dan object.

```
by_mean = ['wage1','wage2','wage3','hours','stby_pay','shift_diff','holidays']
by_mode = ['dur','cola','pension','educ_allw','vacation','lngtrm_disabil',
           'dntl_ins','bereavement','empl_plan']
```

```
for x in by_mean:
    df[x].fillna(round(df[x].mean()), inplace=True)

for x in by_mode:
    df[x].fillna(df[x].mode()[0], inplace=True)
```

Selanjutnya pada dataset tidak ditemukan nilai duplicate sehingga tidak perlu dilakukan perubahan atau penanganan apapun,

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 17 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   dur             40 non-null     float64
 1   wage1           40 non-null     float64
 2   wage2           40 non-null     float64
 3   wage3           40 non-null     float64
 4   cola            40 non-null     object 
 5   hours           40 non-null     float64
 6   pension         40 non-null     object 
 7   stby_pay        40 non-null     float64
 8   shift_diff      40 non-null     float64
 9   educ_allw       40 non-null     object 
 10  holidays        40 non-null     float64
 11  vacation        40 non-null     object 
 12  lngtrm_disabil  40 non-null     object 
 13  dntl_ins        40 non-null     object 
 14  bereavement     40 non-null     object 
 15  empl_plan       40 non-null     object 
 16  good/bad        40 non-null     object 
dtypes: float64(8), object(9)
memory usage: 5.4+ KB
```

Kemudian data disimpan dan akan di *load* untuk proses Asosiasi.

```
df.to_csv('labor-neg.csv')
```

```
df.head()
```

	dur	wage1	wage2	wage3	cola	hours	pension	stby_pay	shift_diff	educ_allw	holidays	vacation	lngtrm_disabil	dntl_ins	bereavement	empl_plan	good/bad
0	1.0	5.0	4.0	4.0	none	40.0	none	6.0	2.0	no	11.0	average	yes	half	yes	full	good
1	2.0	4.5	5.8	4.0	none	35.0	ret_allw	6.0	5.0	yes	11.0	below average	yes	full	yes	full	good
2	2.0	4.0	4.0	4.0	none	38.0	empl_contr	6.0	5.0	no	11.0	generous	yes	half	yes	half	good
3	3.0	3.7	4.0	5.0	tc	38.0	none	6.0	5.0	yes	11.0	below average	yes	half	yes	full	good
4	3.0	4.5	4.5	5.0	none	40.0	none	6.0	5.0	no	12.0	average	yes	half	yes	half	good

Encode Data dari setiap atribut

```
df['dur'] = df['dur'].replace([1.0,2.0,3.0], ['duration1','duration2','duration3'])
df['dur'].value_counts()
```

```
duration2    18
duration3    13
duration1     9
Name: dur, dtype: int64
```

```
df['wage1'] = df['wage1'].replace([2.0, 4.5, 3.5, 3.0, 4.0, 5.0, 2.5, 2.8, 5.7, 3.7, 6.0, 6.4, 2.1, 4.3, 6.9],
['wageFirst2.0', 'wageFirst4.5', 'wageFirst3.5', 'wageFirst3.0', 'wageFirst4.0', 'wageFirst5.0', 'wageFirst2.5', 'wageFirst2.8',
 'wageFirst5.7', 'wageFirst3.7', 'wageFirst6.0', 'wageFirst6.4', 'wageFirst2.1', 'wageFirst4.3', 'wageFirst6.9'])
df["wage1"].value_counts()
```

```
wageFirst2.0    8
wageFirst4.5    7
wageFirst3.5    4
wageFirst4.0    4
wageFirst3.0    4
wageFirst5.0    2
wageFirst2.5    2
wageFirst2.8    2
wageFirst5.7    1
wageFirst6.9    1
wageFirst2.1    1
wageFirst6.4    1
wageFirst6.0    1
wageFirst3.7    1
wageFirst4.3    1
Name: wage1, dtype: int64
```

```
df['wage2'] = df['wage2'].replace([4.0, 2.5, 4.5, 2.0, 5.0, 3.0, 7.0, 5.8, 4.4, 4.8, 6.4],
['wageSecond4.0', 'wageSecond2.5', 'wageSecond4.5', 'wageSecond2.0', 'wageSecond5.0', 'wageSecond3.0', 'wageSecond7.0',
'wageSecond5.8', 'wageSecond4.4', 'wageSecond4.8', 'wageSecond6.4'])
df["wage2"].value_counts()
```

```
wageSecond4.0    19
wageSecond2.5     5
wageSecond2.0     3
wageSecond5.0     3
wageSecond4.5     3
wageSecond3.0     2
wageSecond5.8     1
wageSecond4.4     1
wageSecond7.0     1
wageSecond4.8     1
wageSecond6.4     1
Name: wage2, dtype: int64
```

```
df['wage3'] = df['wage3'].replace([4.0, 5.0, 2.0, 4.6, 2.5, 5.1, 2.3, 2.1],
['wageThird4.0', 'wageThird5.0', 'wageThird2.0', 'wageThird4.6', 'wageThird2.5', 'wageThird5.1', 'wageThird2.3', 'wageThird2.1'])
df["wage3"].value_counts()
```

```
wageThird4.0    28
wageThird5.0     4
wageThird2.0     2
wageThird4.6     2
wageThird5.1     1
wageThird2.5     1
wageThird2.1     1
wageThird2.3     1
Name: wage3, dtype: int64
```

```
df['cola'] = df['cola'].replace(["tc","none","tcf"],['cola_tc','cola_none','cola_tcf'])
df['cola'].value_counts()
```

```
cola_none     30
cola_tc        6
cola_tcf       4
Name: cola, dtype: int64
```

```
df['hours'] = df['hours'].replace([40.0, 38.0, 35.0, 37.0, 36.0, 27.0, 33.0],
                                  ['hours40','hours38','hours35','hours37','hours36','hours27','hours33'])
df['hours'].value_counts()
```

```
hours40    16
hours38    11
hours35     5
hours37     4
hours36     2
hours33     1
hours27     1
Name: hours, dtype: int64
```

```
df['pension'] = df['pension'].replace(["empl_contr","none","ret_allw"],["pension_empl","pension_none","pension_ret"])
df['pension'].value_counts()
```

```
pension_none    30
pension_empl     7
pension_ret      3
Name: pension, dtype: int64
```

```
df['stby_pay'] = df['stby_pay'].replace([6.0,2.0,4.0,12.0,13.0,8.0],
                                        ["pay6","pay2","pay4","pay12","pay13","pay8"])
df['stby_pay'].value_counts()
```

```
pay6     33
pay2      3
pay13     1
pay8      1
pay12     1
pay4      1
Name: stby_pay, dtype: int64
```

```
df['shift_diff'] = df['shift_diff'].replace([5.0,4.0,3.0,2.0,0.0,25.0,10.0,6.0,1.0],
                     ["shift5","shift4","shift3","shift2","shift0","shift25","shift10","shift6","shift1"])
df['shift_diff'].value_counts()
```

```
shift5     20
shift3      6
shift4      6
shift2      3
shift25     1
shift1      1
shift6      1
shift10     1
shift0      1
Name: shift_diff, dtype: int64
```

```
df['educ_allw'] = df['educ_allw'].replace(['yes','no'],['educ_yes','educ_no'])
df['educ_allw'].value_counts()
```

```
educ_no     33
educ_yes     7
Name: educ_allw, dtype: int64
```

```
df['holidays'] = df['holidays'].replace([11.0,10.0,12.0,9.0,15.0,13.0],
                 ["holidays11","holidays10","holidays12","holidays9","holidays15","holidays13"])
df['holidays'].value_counts()
```

```
holidays11    15
holidays10    10
holidays12     8
holidays9      3
holidays13     2
holidays15     2
Name: holidays, dtype: int64
```

```
df['vacation'] = df['vacation'].replace(["below average","average","generous"],['vac_below','vac_avg','vac_gen'])
df['vacation'].value_counts()
```

```
vac_below    17
vac_gen      12
vac_avg      11
Name: vacation, dtype: int64
```

```
df['lngtrm_disabil'] = df['lngtrm_disabil'].replace(["yes","no"],['lngtrm_yes',"lngtrm_no"])
df['lngtrm_disabil'].value_counts()
```

```
lngtrm_yes    35
lngtrm_no      5
Name: lngtrm_disabil, dtype: int64
```

```
df['dntl_ins'] = df['dntl_ins'].replace(["full","half","none","no"],['dntl_full',"dntl_half","dntl_none","dntl_none"])
df['dntl_ins'].value_counts()
```

```
dntl_half    26
dntl_full     8
dntl_none     6
Name: dntl_ins, dtype: int64
```

```
df['empl_plan'] = df['empl_plan'].replace(['full','half','none'],['empl_full','empl_half','empl_none'])
df['empl_plan'].value_counts()
```

```
empl_full    28
empl_none     6
empl_half     6
Name: empl_plan, dtype: int64
```

```
df['bereavement'] = df['bereavement'].replace(['yes','no'],['bereavement_yes','bereveament_no'])
df['bereavement'].value_counts()
```

```
bereavement_yes    38
bereveament_no      2
Name: bereavement, dtype: int64
```

3. Proses Asosiasi

Untuk Proses Asosiasi menggunakan Algoritma Apriori juga sama seperti dataset sebelumnya, untuk implementasi proses akan menggunakan bantuan library apyori pada python sehingga dapat terimplementasikan.

Implementasi Pada Python

Pada bahasa pemrograman python algoritma ini dapat menggunakan library berikut ini untuk melakukan atau implementasinya.

```
from apyori import apriori
```

Selanjutnya load dataset yang sudah di preproses sebelumnya dan mendrop "good/bad"dengan perintah berikut ini.

```
df_apr = df.drop(['good/bad'],1)
df_apr.head()
```

	dur	wage1	wage2	wage3	cola	hours	pension	stby_pay	shift_diff	educ_allw	holidays	vacation	lngtrm_disabil	dntl_ins	bereavement	empl
0	duration1	wageFirst5.0	wageSecond4.0	wageThird4.0	cola_none	hours40	pension_none	pay6	shift2	educ_no	holidays11	vac_avg	lngtrm_yes	dntl_half	bereavement_yes	en
1	duration2	wageFirst4.5	wageSecond5.8	wageThird4.0	cola_none	hours35	pension_ret	pay6	shift5	educ_yes	holidays11	vac_below	lngtrm_yes	dntl_full	bereavement_yes	en
2	duration2	wageFirst4.0	wageSecond4.0	wageThird4.0	cola_none	hours38	pension_empl	pay6	shift5	educ_no	holidays11	vac_gen	lngtrm_yes	dntl_half	bereavement_yes	em
3	duration3	wageFirst3.7	wageSecond4.0	wageThird5.0	cola_tc	hours38	pension_none	pay6	shift5	educ_yes	holidays11	vac_below	lngtrm_yes	dntl_half	bereavement_yes	en
4	duration3	wageFirst4.5	wageSecond4.5	wageThird5.0	cola_none	hours40	pension_none	pay6	shift5	educ_no	holidays12	vac_avg	lngtrm_yes	dntl_half	bereavement_yes	em

kemudian kita buat rule menggunakan apriori dengan parameter yang perlu di perhatikan adalah min_support, min_confidence dan max_lengthnya

```
records = []
for i in range(0, 39):
    records.append([str(df.values[i,j]) for j in range(0, 16)])
rules = apriori(records, min_support = 0.04, min_confidence = 0.5, min_lift = 3, max_length = 2, target = "rules")
association_results = list(rules)

print(len(association_results))
```

sebelum diproses perlu dilakukan convert dari data frame menjadi list sehingga dapat diproses, pada proses kali ini akan menggunakan parameter **min_support = 0.04 (4%), min_confidence = 0.5 (50%), min_lift = 3, max_length = 2,** dan hasilnya akan seperti berikut ini.

```
Rule :  bereveament_no  -> dntl_none
Support :  0.05128205128205128
Confidence :  1.0
Lift :  6.5
=============================
Rule :  bereveament_no  -> duration1
Support :  0.05128205128205128
Confidence :  1.0
Lift :  4.333333333333333
=============================
Rule :  bereveament_no  -> empl_none
Support :  0.05128205128205128
Confidence :  1.0
Lift :  6.5
=============================
Rule :  bereveament_no  -> lngtrm_no
Support :  0.05128205128205128
Confidence :  1.0
Lift :  7.800000000000001
=============================
Rule :  bereveament_no  -> wageFirst2.0
Support :  0.05128205128205128
Confidence :  1.0
Lift :  4.875
=============================
Rule :  cola_tc  -> pay2
Support :  0.05128205128205128
Confidence :  0.6666666666666666
Lift :  4.333333333333333
=============================
```

```
=============================
Rule :  pension_ret  -> cola_tc
Support :  0.05128205128205128
Confidence :  0.6666666666666666
Lift :  4.333333333333333
=============================
Rule :  wageThird5.0  -> cola_tc
Support :  0.05128205128205128
Confidence :  0.5
Lift :  3.25
=============================
Rule :  wageFirst3.5  -> cola_tcf
Support :  0.05128205128205128
Confidence :  0.5
Lift :  4.875
=============================
Rule :  hours37  -> dntl_full
Support :  0.07692307692307693
Confidence :  0.7500000000000001
Lift :  4.178571428571429
=============================
Rule :  dntl_full  -> shift4
Support :  0.10256410256410256
Confidence :  0.5714285714285714
Lift :  4.457142857142857
=============================
Rule :  empl_none  -> dntl_none
Support :  0.07692307692307693
Confidence :  0.5
Lift :  3.25
=============================
```

dari pengamatan yang dilakukan jika melakukan pengaturan terhadap min_support, min_confidence ini dapat merubah hasilnya karena seperti yang kita ketahui apriori akan terus melakukan perulangan dan tidak menggunakan nilai support yang dibawah min_support, sehingga bila semakin kecil nilai min_support maka kemungkinan rule akan muncul lebih banyak, namun akan lebih spesifik ke rule tertentu, dan pada asosiasi sendiri penentuan ini ditentukan oleh keperluan penguna.

```
Maximum dan Minimum Value dengan Contoh Rulenya

Maximum Value
============================
Rule :  bereveament_no  -> dntl_none
Best Support :  0.10256410256410256
============================
Rule :  bereveament_no  -> dntl_none
Best Confidence :  1.0
============================

Minimum Value
============================
Rule :  bereveament_no  -> dntl_none
Minimum Support :  0.05128205128205128
============================
Rule :  bereveament_no  -> dntl_none
Minimum Confidence :  0.5
============================
```

```
Maximum Value
=============================
Rule  0 :  bereveament_no  -> dntl_none
Rule  1 :  bereveament_no  -> duration1
Rule  2 :  bereveament_no  -> empl_none
Rule  3 :  bereveament_no  -> lngtrm_no
Rule  4 :  bereveament_no  -> wageFirst2.0
Rule  5 :  cola_tc  -> pay2
Rule  6 :  pension_ret  -> cola_tc
Rule  7 :  wageThird5.0  -> cola_tc
Rule  8 :  wageFirst3.5  -> cola_tcf
Rule  9 :  hours37  -> dntl_full
Rule  10 :  dntl_full  -> shift4
Rule  11 :  duration3  -> wageThird5.0
Rule  12 :  wageSecond2.5  -> holidays10
Rule  13 :  vac_avg  -> wageSecond2.5
Rule  14 :  wageFirst2.0  -> wageSecond2.5
=============================
Best Support :  0.10256410256410256
```

```
=============================
Rule  0 :  bereveament_no  -> dntl_none
Rule  1 :  bereveament_no  -> duration1
Rule  2 :  bereveament_no  -> empl_none
Rule  3 :  bereveament_no  -> lngtrm_no
Rule  4 :  bereveament_no  -> wageFirst2.0
Rule  5 :  duration1  -> holidays9
Rule  6 :  wageFirst2.8  -> duration1
Rule  7 :  duration3  -> holidays13
Rule  8 :  wageThird2.0  -> duration3
Rule  9 :  duration3  -> wageThird4.6
Rule  10 :  duration3  -> wageThird5.0
Rule  11 :  wageThird2.0  -> holidays10
Rule  12 :  holidays13  -> vac_gen
Rule  13 :  wageFirst3.5  -> holidays13
Rule  14 :  shift3  -> holidays9
Rule  15 :  hours36  -> vac_gen
Rule  16 :  wageFirst2.0  -> wageThird2.0
=============================
Best Confidence :  1.0
```

```
Minimum Value
============================
Rule  0 :  bereveament_no  -> dntl_none
Rule  1 :  bereveament_no  -> duration1
Rule  2 :  bereveament_no  -> empl_none
Rule  3 :  bereveament_no  -> lngtrm_no
Rule  4 :  bereveament_no  -> wageFirst2.0
Rule  5 :  cola_tc  -> pay2
Rule  6 :  pension_ret  -> cola_tc
Rule  7 :  wageThird5.0  -> cola_tc
Rule  8 :  wageFirst3.5  -> cola_tcf
Rule  9 :  dntl_none  -> wageFirst4.0
Rule  10 :  dntl_none  -> wageSecond5.0
Rule  11 :  wageFirst2.8  -> duration1
Rule  12 :  duration3  -> holidays13
Rule  13 :  wageThird2.0  -> duration3
Rule  14 :  duration3  -> wageThird4.6
Rule  15 :  wageFirst4.0  -> empl_half
Rule  16 :  empl_half  -> wageSecond4.5
Rule  17 :  wageThird5.0  -> empl_half
Rule  18 :  empl_none  -> holidays9
Rule  19 :  empl_none  -> pay2
Rule  20 :  empl_none  -> pension_ret
Rule  21 :  wageThird2.0  -> holidays10
Rule  22 :  holidays13  -> vac_gen
Rule  23 :  wageFirst3.5  -> holidays13
Rule  24 :  holidays9  -> pay2
Rule  25 :  hours36  -> vac_gen
Rule  26 :  shift3  -> pay2
Rule  27 :  wageFirst2.0  -> wageSecond2.0
Rule  28 :  wageFirst2.0  -> wageThird2.0
Rule  29 :  wageFirst3.5  -> wageThird4.6
Rule  30 :  wageFirst4.5  -> wageThird5.0
Rule  31 :  wageThird5.0  -> wageSecond4.5
============================
minimum Support :  0.05128205128205128
```

```
============================
Rule  0 :  bereveament_no  -> dntl_none
Rule  1 :  bereveament_no  -> duration1
Rule  2 :  bereveament_no  -> empl_none
Rule  3 :  bereveament_no  -> lngtrm_no
Rule  4 :  bereveament_no  -> wageFirst2.0
Rule  5 :  cola_tc  -> pay2
Rule  6 :  pension_ret  -> cola_tc
Rule  7 :  wageThird5.0  -> cola_tc
Rule  8 :  wageFirst3.5  -> cola_tcf
Rule  9 :  empl_none  -> dntl_none
Rule  10 :  lngtrm_no  -> dntl_none
Rule  11 :  dntl_none  -> wageFirst4.0
Rule  12 :  wageFirst4.0  -> empl_half
Rule  13 :  wageThird5.0  -> empl_half
Rule  14 :  empl_none  -> lngtrm_no
Rule  15 :  empl_none  -> shift3
Rule  16 :  wageFirst2.0  -> wageSecond2.5
Rule  17 :  wageFirst3.5  -> wageThird4.6
Rule  18 :  wageFirst4.5  -> wageSecond4.5
Rule  19 :  wageFirst4.5  -> wageThird5.0
============================
minimum Confidence :  0.5
```

Pada dataset Labor Relation menghasilkan Best Support 0.102 dan Best Confidence 1.0 sedangkan untuk Minimum Support 0.512 dan Minimum Confidence 0.5