“KONSEP & APLIKASI DATA MAINING”

Oleh:

Agnes Martha Muwa
17.51.0007

PROGRAM STUDI S1- TEKNOLOGI INFORMASI
PROGRAM STUDI S1- SISTEM INFORMASI

KEMENTRIAN RISET DAN TEKNOLOGI PENDIDIKAN TINGGI
SEKOLAH TINGGI MANAJEMEN INFORMATIKA KOMPUTER
PRADNYA PARAMITA
MALANG
2020

1)

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [7]:
```
df=pd.read_csv('D:/agnes/dataset_soalno1.csv',delimiter=';')
```

In [8]:
```
df.head()
```

Out[8]:

	Age	Income	Student	Credit_rating	Class (buy_computer)
0	<=30	High	No	Fair	No
1	<=30	High	No	Excellent	No
2	31..40	High	No	Fair	Yes
3	> 40	Medium	No	Fair	Yes
4	> 40	Low	Yes	Fair	Yes

In [9]:
```
df.shape
```

Out[9]:
```
(51, 5)
```

In [10]:
```
#student
df['Student'].value_counts()
```

Out[10]:
```
Yes    27
No     24
Name: Student, dtype: int64
```

In [11]:
```
PYes = 27/51
PNo = 24/51
```

In [12]:
```
print(PYes)
```

```
0.5294117647058824
```

In [13]:
```
print(PNo)
```

```
0.47058823529411764
```

In [14]:
```
#income with student
pd.crosstab(df['Income'], df['Student'])
```

Out[14]:

Student	No	Yes
Income		
High	9	2
Low	1	20
Medium	14	5

In [15]:
```
PHighNo = 9/24
PLowNo = 1/24
PMediumNo= 14/24

PHighYes = 2/27
PLowYes = 20/27
PMediumYes = 5/27

PHigh = 11/51
PLow = 21/51
PMedium = 19/51

print(PHighNo)
```

```
0.375
```

In [16]:
```
print(PLowNo)
```

```
0.041666666666666664
```

In [17]:
```
print(PMediumNo)
```

```
0.5833333333333334
```

In [18]:
```
print(PHighYes)
```

```
0.07407407407407407
```

In [19]:
```
print(PLowYes)
```

```
0.7407407407407407
```

In [20]:
```
print(PMediumYes)
```

```
0.18518518518518517
```

In [21]:
```
print(PHigh)
```

```
0.21568627450980393
```

In [22]:
```
print(PLow)
```

```
0.4117647058823529
```

In [23]:
```
print(PMedium)
```

```
0.37254901960784315
```

In [24]:
```
#credit rating with student
pd.crosstab(df['Credit_rating'], df['Student'])
```

Out[24]:

Credit_rating	Student No	Yes
Excellent	8	12
Fair	16	15

In [25]:
```
PExcellentNo = 8/24
PFairNo = 16/24

PExcellentYes = 12/27
PFairYes = 15/27

PExcellent = 20/51
PFair = 31/51

print(PExcellentNo)
```

```
0.3333333333333333
```

In [26]:
```
print(PFairNo)
```

```
0.6666666666666666
```

In [27]:
```
print(PExcellentYes)
```

```
0.4444444444444444
```

In [28]:
```
print(PFairYes)
```

```
0.5555555555555556
```

In [29]:
```
print(PExcellent)
```

```
0.39215686274509803
```

In [30]:
```
print(PFair)
```

```
0.6078431372549019
```

In [31]:
```
#income with class(buy_computer)
pd.crosstab(df['Income'], df['Class (buy_computer)'])
```

Out[31]:

Income	Class (buy_computer) No	Yes
High	6	5
Low	11	10
Medium	5	14

In [32]:
```
PHighNo = 6/22
PLowNo = 11/22
PMediumNo= 5/22

PHighYes = 5/29
PLowYes = 10/29
PMediumYes = 24/29

PHigh = 11/51
PLow = 21/51
PMedium = 19/51

print(PHighNo)
```

```
0.2727272727272727
```

In [33]:
```
print(PLowNo)
```

```
0.5
```

In [34]:
```
print(PMediumNo)
```

```
0.22727272727272727
```

In [35]:
```
print(PHighYes)
```

```
0.1724137931034483
```

In [36]:
```
print(PLowYes)
```

```
0.3448275862068966
```

In [37]:
```
print(PMediumYes)
```

```
0.8275862068965517
```

In [38]:
```
#credit rating with class(buy_computer)
pd.crosstab(df['Credit_rating'], df['Class (buy_computer)'])
```

Out[38]:

Class (buy_computer)	No	Yes
Credit_rating		
Excellent	8	12
Fair	14	17

In [39]:
```
PExcellentNo = 8/22
PFairNo = 14/22

PExcellentYes = 12/29
PFairYes = 17/29

PExcellent = 20/51
PFair = 31/51
```

```
0.36363636363636365
```

In [40]:
```
print(PFairNo)
```

```
0.6363636363636364
```

In [41]:
```
print(PExcellentYes)
```

```
0.41379310344827586
```

In [42]:
```
print(PFairYes)
```

```
0.5862068965517241
```

jupyter **JawabanNo2** Last Checkpoint: Last Sunday at 7:29 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
pd.__version__
```

Out[2]: '1.0.1'

In [3]:
```
df = pd.read_excel('D:/agnes/dataset_soalno2.xls')
```

In [4]:
```
df
```

Out[4]:

	Category	Weather	Holiday	Game	Qty
0	NaN	V-1	V-2	V-3	NaN
1	A	5	1	0	250.0
2	B	3	1	1	200.0
3	C	1	1	0	75.0
4	D	4	1	1	400.0
5	E	4	0	0	150.0
6	F	2	0	0	50.0

3	C	1	1	0	75.0
4	D	4	1	1	400.0
5	E	4	0	0	150.0
6	F	2	0	0	50.0

In [38]:
```
#a.Apabila Cuaca buruk dengan nilai = 1, Weekday, dan Game = 0, maka berapa roti yang harus dibuat?
#  misalkan hari misterius = H-M (Weekday)
data = np.array([[5.,3.,'Weather V-1'],[1.,4.,'Weather V-1'],[4.,2.,'Weather V-1'],[1.,1.,'Holiday V-2'],[1.,1.,'Holiday V-2']
query = [1.,0.,'Weekday H-M']
```

In [39]:
```
df = pd.DataFrame(data)
df.columns = ['x','y','Qty']
df
```

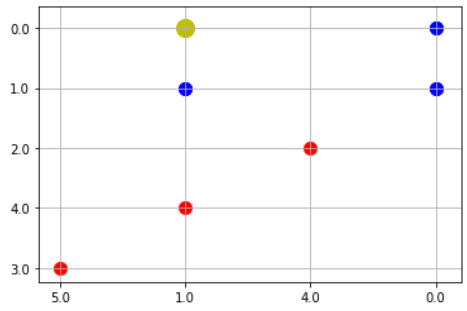
Out[39]:

	x	y	Qty
0	5.0	3.0	Weather V-1
1	1.0	4.0	Weather V-1
2	4.0	2.0	Weather V-1
3	1.0	1.0	Holiday V-2
4	1.0	1.0	Holiday V-2
5	0.0	0.0	Holiday V-2
6	0.0	1.0	Game V-3

5	0.0	0.0	Holiday V-2
6	0.0	1.0	Game V-3
7	0.0	1.0	Game V-3
8	0.0	0.0	Game V-3
9	1.0	0.0	Weekday H-M

[41]:
```
for i in range(10):
    if(df.iloc[i]['Qty'] == 'Weather V-1'):
        plt.scatter(df.iloc[i]['x'], df.iloc[i]['y'], s=100, c='r')
    elif(df.iloc[i]['Qty'] == 'Weekday H-M'):
        plt.scatter(df.iloc[i]['x'], df.iloc[i]['y'], s=200, c='y')
    else:
     plt.scatter(df.iloc[i]['x'], df.iloc[i]['y'], s=100, c='b')


plt.grid()
plt.show()
```



In [42]:
```
import math
dis = []
for i in range(10):
    dis.append(math.sqrt((float(df.iloc[i]['x']) - query[1]) **2 + (float(df.iloc[i]['y']) - query[0]) **2))
```

In [43]:
```
df['dis'] = dis
df
```

In [43]:
```
df['dis'] = dis
df
```

Out[43]:

	x	y	Qty	dis
0	5.0	3.0	Weather V-1	5.385165
1	1.0	4.0	Weather V-1	3.162278
2	4.0	2.0	Weather V-1	4.123106
3	1.0	1.0	Holiday V-2	1.000000
4	1.0	1.0	Holiday V-2	1.000000
5	0.0	0.0	Holiday V-2	1.000000
6	0.0	1.0	Game V-3	0.000000
7	0.0	1.0	Game V-3	0.000000
8	0.0	0.0	Game V-3	1.000000
9	1.0	0.0	Weekday H-M	1.414214

In [44]:
```
df.sort_values('dis')
```

In [43]:
```
df['dis'] = dis
df
```

Out[43]:

	x	y	Qty	dis
0	5.0	3.0	Weather V-1	5.385165
1	1.0	4.0	Weather V-1	3.162278
2	4.0	2.0	Weather V-1	4.123106
3	1.0	1.0	Holiday V-2	1.000000
4	1.0	1.0	Holiday V-2	1.000000
5	0.0	0.0	Holiday V-2	1.000000
6	0.0	1.0	Game V-3	0.000000
7	0.0	1.0	Game V-3	0.000000
8	0.0	0.0	Game V-3	1.000000
9	1.0	0.0	Weekday H-M	1.414214

In [44]:
```
df.sort_values('dis')
```

Out[44]:

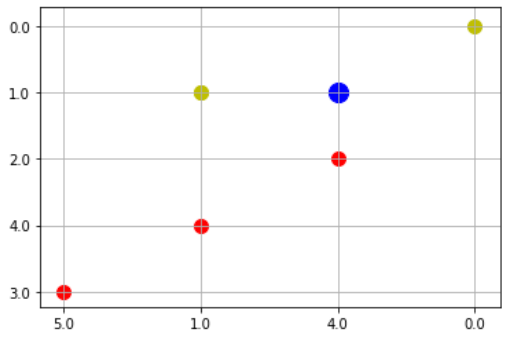
	x	y	Qty	dis
6	0.0	1.0	Game V-3	0.000000
7	0.0	1.0	Game V-3	0.000000
3	1.0	1.0	Holiday V-2	1.000000
4	1.0	1.0	Holiday V-2	1.000000
5	0.0	0.0	Holiday V-2	1.000000
8	0.0	0.0	Game V-3	1.000000
9	1.0	0.0	Weekday H-M	1.414214
1	1.0	4.0	Weather V-1	3.162278
2	4.0	2.0	Weather V-1	4.123106
0	5.0	3.0	Weather V-1	5.385165

In [45]:
```
df.to_excel('D:/agnes/outputNo2(a).xls')
```

In [46]:
```
#b.Apabila Cuaca baik dengan nilai 4, Weekend, dan Game =1, maka berapa roti yang harus dibuat?
##  misalkan hari misterius = "H-M" (Weekenda)
data = np.array([[5.,3.,'Weather V-1'],[1.,4.,'Weather V-1'],[4.,2.,'Weather V-1'],[1.,1.,'Holiday V-2'],[1.,1.,'Holiday V-2']
query = [4.,1.,'Weekend H-M']
```

In [26]:
```
for i in range(7):
    if(df.iloc[i]['Qty'] == 'V1'):
        plt.scatter(df.iloc[i]['x'], df.iloc[i]['y'], s=100, c='r')
    elif(df.iloc[i]['Qty'] == 'V2'):
        plt.scatter(df.iloc[i]['x'], df.iloc[i]['y'], s=100, c='y')
    else:
     plt.scatter(df.iloc[i]['x'], df.iloc[i]['y'], s=200, c='b')


plt.grid()
plt.show()
```



In [24]:
```
#b.Apabila Cuaca baik dengan nilai 4, Weekend, dan Game =1, maka berapa roti yang harus dibuat?
##  misalkan hari misterius = "NN"
data = np.array([[5.,3.,'V1'],[1.,4.,'V1'],[4.,2.,'V1'],[1.,1.,'V2'],[1.,1.,'V2'],[0.,0.,'V2'],[4.,1.,'NN']])
query = [4.,1.,'NN']
```

In [25]:
```
df = pd.DataFrame(data)
df.columns = ['x','y','Qty']
df
```

Out[25]:

	x	y	Qty
0	5.0	3.0	V1
1	1.0	4.0	V1
2	4.0	2.0	V1
3	1.0	1.0	V2
4	1.0	1.0	V2
5	0.0	0.0	V2
6	4.0	1.0	NN

In [37]:
```
import math
dis = []
for i in range(7):
    dis.append(math.sqrt((float(df.iloc[i]['x']) - query[0]) **2 + (float(df.iloc[i]['y']) - query[1]) **2))
```

In [38]:
```
df['dis'] = dis
df
```

Out[38]:

	x	y	Qty	dis
0	5.0	3.0	V1	2.236068
1	1.0	4.0	V1	4.242641
2	4.0	2.0	V1	1.000000
3	1.0	1.0	V2	3.000000
4	1.0	1.0	V2	3.000000
5	0.0	0.0	V2	4.123106
6	4.0	1.0	NN	0.000000

In [39]:
```
df.sort_values('dis')
```

In [39]:
```
df.sort_values('dis')
```

Out[39]:

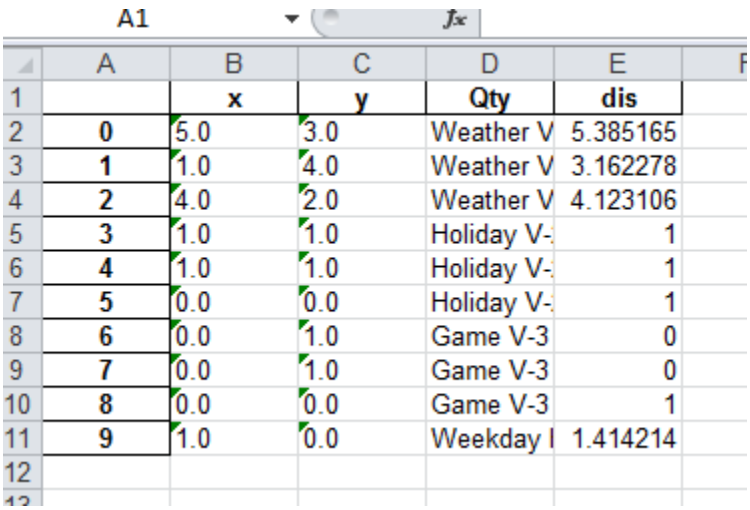
	x	y	Qty	dis
6	4.0	1.0	NN	0.000000
2	4.0	2.0	V1	1.000000
0	5.0	3.0	V1	2.236068
3	1.0	1.0	V2	3.000000
4	1.0	1.0	V2	3.000000
5	0.0	0.0	V2	4.123106
1	1.0	4.0	V1	4.242641

In [40]:
```
df.to_excel('D:/agnes/outputNo2(b).xls')
```

In [ ]:
```
```

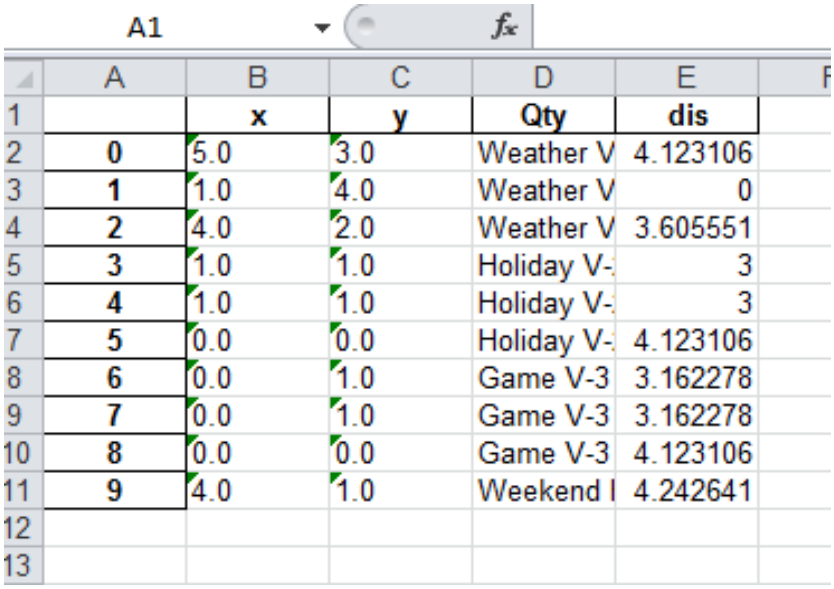
OUTPUT NO2

a.



	A	B	C	D	E
1		x	y	Qty	dis
2	0	5.0	3.0	Weather V	5.385165
3	1	1.0	4.0	Weather V	3.162278
4	2	4.0	2.0	Weather V	4.123106
5	3	1.0	1.0	Holiday V-	1
6	4	1.0	1.0	Holiday V-	1
7	5	0.0	0.0	Holiday V-	1
8	6	0.0	1.0	Game V-3	0
9	7	0.0	1.0	Game V-3	0
10	8	0.0	0.0	Game V-3	1
11	9	1.0	0.0	Weekday I	1.414214
12					

b.



	A	B	C	D	E
1		x	y	Qty	dis
2	0	5.0	3.0	Weather V	4.123106
3	1	1.0	4.0	Weather V	0
4	2	4.0	2.0	Weather V	3.605551
5	3	1.0	1.0	Holiday V-	3
6	4	1.0	1.0	Holiday V-	3
7	5	0.0	0.0	Holiday V-	4.123106
8	6	0.0	1.0	Game V-3	3.162278
9	7	0.0	1.0	Game V-3	3.162278
10	8	0.0	0.0	Game V-3	4.123106
11	9	4.0	1.0	Weekend I	4.242641
12					
13					

jupyter JawabanNo3 Last Checkpoint: 07/01/2020 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [2]:
```
import numpy as np
import pandas as pd
from apyori import apriori
```

In [6]:
```
store_data = pd.read_excel ('D:agnes/dataset_soalno3.xls')
```

In [8]:
```
store_data.head()
```

Out[8]:

	Item1	Item2	Item3	Item4	Item5	Item6	Item7	Item8	Item9	Item10
0	burgers	meatballs	eggs	low fat yogurt	NaN	mineral water	salmon	low fat yogurt	NaN	mineral water
1	chutney	low fat yogurt	NaN	whole wheat pasta	french fries	mineral water	salmon	whole wheat pasta	french fries	mineral water
2	turkey	whole wheat pasta	french fries	soup	light cream	shallot	NaN	soup	light cream	shallot
3	mineral water	soup	light cream	frozen vegetables	spaghetti	green tea	NaN	frozen vegetables	spaghetti	green tea
4	low fat yogurt	frozen vegetables	spaghetti	french fries	eggs	chocolate	frozen smoothie	french fries	eggs	chocolate

In [9]:
```
store_data.tail()
```

In [9]:
```
store_data.tail()
```

Out[9]:

	Item1	Item2	Item3	Item4	Item5	Item6	Item7	Item8	Item9	Item10
2049	burgers	eggs	french fries	fresh tuna	spaghetti	olive oil	clothes accessories	turkey	eggs	french fries
2050	burgers	eggs	frozen smoothie	french wine	eggs	french fries	energy drink	french fries	NaN	chocolate
2051	whole wheat pasta	cake	melons	champagne	pancakes	light mayo	soup	chocolate	milk	herb & pepper
2052	ground beef	tomato sauce	spaghetti	red wine	honey	hot dogs	turkey	herb & pepper	whole wheat pasta	mineral water
2053	burgers	eggs	frozen smoothie	milk	bacon	eggs	french fries	mineral water	avocado	cookies

In [10]:
```
store_data.shape
```

Out[10]: (2054, 10)

In [11]:
```
records = []
for i in range (0, 2054):
    records.append ([str(store_data.values[i,j])for j in range (0, 10)])
```

In [12]:
```
association_rules = apriori (records, min_support=0.2,min_confidence=0.2,min_lenght=2)
association_results = list (association_rules)
```

In [13]:
```
print(len(association_results))
```

```
61
```

In [14]:
```
print (association_results[0])
```

```
RelationRecord(items=frozenset({'avocado'}), support=0.314508276533593, ordered_statistics=[OrderedStatistic(items_base=fro
zenset(), items_add=frozenset({'avocado'}), confidence=0.314508276533593, lift=1.0)])
```

In [48]:
```
results =[]
for item in association_results:
    pair = item[0]
    items = [X for X in pair]

    value0 = str(items[0])
    value1 = str(item[1])
    value2 = str(item[1])[:10]
    value3 = str(item[2][0][2])[:10]
    value4 = str(item[2][0][3])[:10]

    rows = (value0,value1,value2,value3,value4)

    results.append(rows)

    label = ['title1'], ['title2'], ['support'], ['confidence'], ['lift']

    store_suggestion = pd.DataFrame.from_records(results,columns=label)

    print (store_suggestion)
```

```
  (title1,)          (title2,)  (support,) (confidence,) (lift,)
0   avocado  0.314508276533593  0.31450827    0.31450827     1.0
  (title1,)           (title2,)  (support,) (confidence,) (lift,)
0   avocado   0.314508276533593  0.31450827    0.31450827     1.0
1   burgers  0.24294060370009737  0.24294060    0.24294060     1.0
   (title1,)           (title2,)  (support,) (confidence,) (lift,)
0    avocado   0.314508276533593  0.31450827    0.31450827     1.0
1    burgers  0.24294060370009737  0.24294060    0.24294060     1.0
2  chocolate   0.4756572541382668  0.47565725    0.47565725     1.0
            (title1,)           (title2,)  (support,) (confidence,) (lift,)
0             avocado   0.314508276533593  0.31450827    0.31450827     1.0
1             burgers  0.24294060370009737  0.24294060    0.24294060     1.0
2           chocolate   0.4756572541382668  0.47565725    0.47565725     1.0
3  clothes accessories  0.33982473222979553  0.33982473    0.33982473     1.0
            (title1,)           (title2,)  (support,) (confidence,) (lift,)
0             avocado   0.314508276533593  0.31450827    0.31450827     1.0
1             burgers  0.24294060370009737  0.24294060    0.24294060     1.0
2           chocolate   0.4756572541382668  0.47565725    0.47565725     1.0
3  clothes accessories  0.33982473222979553  0.33982473    0.33982473     1.0
```

In [49]:
```
store_suggestion.describe()
```

```
59           chocolate  0.23661148977604674  0.23661148    0.23661148     1.0
             (title1,)           (title2,)  (support,) (confidence,) (lift,)
0              avocado   0.314508276533593  0.31450827    0.31450827     1.0
1              burgers  0.24294060370009737  0.24294060    0.24294060     1.0
2            chocolate   0.4756572541382668  0.47565725    0.47565725     1.0
3   clothes accessories  0.33982473222979553  0.33982473    0.33982473     1.0
4              cookies   0.3588120740019474  0.35881207    0.35881207     1.0
..                 ...                  ...         ...           ...     ...
56                milk  0.20837390457643623  0.20837390    0.20837390     1.0
57        mineral water   0.2249269717624148  0.22492697    0.22492697     1.0
58         french fries  0.22249269717624148  0.22249269    0.22249269     1.0
59            chocolate  0.23661148977604674  0.23661148    0.23661148     1.0
60  clothes accessories  0.24196689386562803  0.24196689    0.24196689     1.0

[61 rows x 5 columns]
```

In [49]:
```
store_suggestion.describe()
```

Out[49]:

	(title1,)	(title2,)	(support,)	(confidence,)	(lift,)
count	61	61	61	61	61
unique	15	53	53	53	1
top	french fries	0.24294060370009737	0.24294060	0.24294060	1.0
freq	12	4	4	4	61

In [50]:
```
store_suggestion.to_excel('D:/agnes/outputno3.xls')
```

OUTPUTNO3

	A	B	C	D	E	F
1		('title1',)	('title2',)	('support',	onfidence	('lif
2	0	avocado	0.3145082	0.3145082	0.3145082	1.0
3	1	burgers	0.2429406	0.2429406	0.2429406	1.0
4	2	chocolate	0.4756572	0.4756572	0.4756572	1.0
5	3	clothes ac	0.3398247	0.3398247	0.3398247	1.0
6	4	cookies	0.3588120	0.3588120	0.3588120	1.0
7	5	eggs	0.4099318	0.4099318	0.4099318	1.0
8	6	energy drir	0.3213242	0.3213242	0.3213242	1.0
9	7	french fries	0.6548198	0.6548198	0.6548198	1.0
10	8	herb & pep	0.3042843	0.3042843	0.3042843	1.0
11	9	milk	0.4079844	0.4079844	0.4079844	1.0
12	10	mineral wa	0.4527750	0.4527750	0.4527750	1.0
13	11	nan	0.6285296	0.6285296	0.6285296	1.0
14	12	shrimp	0.2151898	0.2151898	0.2151898	1.0
15	13	turkey	0.5272638	0.5272638	0.5272638	1.0
16	14	whole whe	0.2653359	0.2653359	0.2653359	1.0
17	15	french fries	0.2030185	0.2030185	0.2030185	1.0
18	16	mineral wa	0.3037974	0.3037974	0.3037974	1.0
19	17	turkey	0.2921129	0.2921129	0.2921129	1.0
20	18	chocolate	0.2521908	0.2521908	0.2521908	1.0
21	19	chocolate	0.3042843	0.3042843	0.3042843	1.0
22	20	milk	0.2711781	0.2711781	0.2711781	1.0
23	21	chocolate	0.3763388	0.3763388	0.3763388	1.0
24	22	clothes ac	0.2512171	0.2512171	0.2512171	1.0
25	23	clothes ac	0.3237585	0.3237585	0.3237585	1.0
26	24	clothes ac	0.2108081	0.2108081	0.2108081	1.0
27	25	clothes ac	0.3281402	0.3281402	0.3281402	1.0
28	26	french fries	0.3213242	0.3213242	0.3213242	1.0
29	27	mineral wa	0.2005842	0.2005842	0.2005842	1.0
30	28	nan	0.2429406	0.2429406	0.2429406	1.0
33	31	eggs	0.2409931	0.2409931	0.2409931	1.0
34	32	eggs	0.2848101	0.2848101	0.2848101	1.0
35	33	french fries	0.3033106	0.3033106	0.3033106	1.0
36	34	nan	0.3033106	0.3033106	0.3033106	1.0
37	35	milk	0.2804284	0.2804284	0.2804284	1.0
38	36	mineral wa	0.2702044	0.2702044	0.2702044	1.0
39	37	french fries	0.4819863	0.4819863	0.4819863	1.0
40	38	french fries	0.3709834	0.3709834	0.3709834	1.0
41	39	herb & pep	0.2478091	0.2478091	0.2478091	1.0
42	40	milk	0.2702044	0.2702044	0.2702044	1.0
43	41	mineral wa	0.3476144	0.3476144	0.3476144	1.0
44	42	nan	0.2502434	0.2502434	0.2502434	1.0
45	43	mineral wa	0.2891918	0.2891918	0.2891918	1.0
46	44	chocolate	0.2429406	0.2429406	0.2429406	1.0
47	45	chocolate	0.2453748	0.2453748	0.2453748	1.0
48	46	chocolate	0.2833495	0.2833495	0.2833495	1.0
49	47	clothes ac	0.2429406	0.2429406	0.2429406	1.0
50	48	clothes ac	0.2497565	0.2497565	0.2497565	1.0
51	49	clothes ac	0.2064264	0.2064264	0.2064264	1.0
52	50	clothes ac	0.3154819	0.3154819	0.3154819	1.0
53	51	clothes ac	0.2020447	0.2020447	0.2020447	1.0
54	52	french fries	0.2215189	0.2215189	0.2215189	1.0
55	53	french fries	0.2185978	0.2185978	0.2185978	1.0
56	54	french fries	0.2653359	0.2653359	0.2653359	1.0
57	55	french fries	0.2911392	0.2911392	0.2911392	1.0
58	56	milk	0.2083739	0.2083739	0.2083739	1.0
59	57	mineral wa	0.2249269	0.2249269	0.2249269	1.0
60	58	french fries	0.2224926	0.2224926	0.2224926	1.0
61	59	chocolate	0.2366114	0.2366114	0.2366114	1.0
62	60	clothes ac	0.2419668	0.2419668	0.2419668	1.0

Sheet1

jupyter **JawabanNo4** Last Checkpoint: 7 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
pd.__version__
```

Out[2]: '1.0.1'

In [10]:
```
df = pd.read_csv('D:/agnes/dataset_soalno4.txt',
                 delimiter=',')
```

In [11]:
```
df
```

Out[11]:

	Usia	Kelahiran_ke-	Waktu_Kelahiran	Tekanan_darah	Kelainan_jantung	Caesarian
0	22	1	0	2	0	0
1	26	2	0	1	0	1
2	26	2	1	1	0	0
3	28	1	0	2	0	0
4	22	2	0	1	0	1
...	...	...	...	...	...	...
...	...	...	...	...	...	...
75	27	2	1	1	0	0
76	33	4	0	1	0	1
77	29	2	1	2	0	1
78	25	1	2	0	0	1
79	24	2	2	1	0	0

80 rows × 6 columns

In [13]:
```
import math
dis = []
for i in range(80):
    dis.append(math.sqrt((float(df.iloc[i]['Usia'])-30)**2+
                         (float(df.iloc[i]['Kelahiran_ke-'])- 1)**2+
                         (float(df.iloc[i]['Waktu_Kelahiran'])-0)**2+
                         (float(df.iloc[i]['Tekanan_darah'])-1)**2))
```

In [14]:
```
df['dis'] = dis
df
```

Out[14]:

Out[14]:

	Usia	Kelahiran_ke-	Waktu_Kelahiran	Tekanan_darah	Kelainan_jantung	Caesarian	dis
0	22	1	0	2	0	0	8.062258
1	26	2	0	1	0	1	4.123106
2	26	2	1	1	0	0	4.242641
3	28	1	0	2	0	0	2.236068
4	22	2	0	1	0	1	8.062258
...	...	...	...	...	...	...	...
75	27	2	1	1	0	0	3.316625
76	33	4	0	1	0	1	4.242641
77	29	2	1	2	0	1	2.000000
78	25	1	2	0	0	1	5.477226
79	24	2	2	1	0	0	6.403124

80 rows × 7 columns

In [15]:
```
df.sort_values('dis')
```

In [15]:
```
df.sort_values('dis')
```

Out[15]:

	Usia	Kelahiran_ke-	Waktu_Kelahiran	Tekanan_darah	Kelainan_jantung	Caesarian	dis
27	30	1	0	1	0	0	0.000000
38	31	1	0	1	0	0	1.000000
67	29	2	0	1	1	0	1.414214
54	29	2	0	1	1	1	1.414214
59	30	2	1	2	1	1	1.732051
...	...	...	...	...	...	...	...
41	19	1	0	1	0	1	11.000000
61	19	1	0	1	0	1	11.000000
25	18	1	0	1	0	0	12.000000
26	18	1	1	2	1	1	12.083046
70	17	1	0	0	0	1	13.038405

80 rows × 7 columns

In [16]:
```
y = df.sort_values('dis').head(5)
y
```

Out[16]:

	Usia	Kelahiran_ke-	Waktu_Kelahiran	Tekanan_darah	Kelainan_jantung	Caesarian	dis
27	30	1	0	1	0	0	0.000000
38	31	1	0	1	0	0	1.000000
67	29	2	0	1	1	0	1.414214
54	29	2	0	1	1	1	1.414214
59	30	2	1	2	1	1	1.732051

In [17]:
```
z = y["Caesarian"]
z
```

Out[17]:
```
27    0
38    0
67    0
54    1
59    1
Name: Caesarian, dtype: int64
```

In [18]:
```
np.mean(z)
```

Out[18]: 0.4

In [19]:
```
df.to_excel('D:/agnes/outputNo4.xls')
```

In [ ]:

OUTPUTNO4

	A	B	C	D	E	F	G	H
1		Usia	Kelahiran_ke-	Waktu_Kelahiran	Tekanan_darah	Kelainan_jantung	Caesarian	dis
2	**0**	22	1	0	2	0	0	8.062257748
3	**1**	26	2	0	1	0	1	4.123105626
4	**2**	26	2	1	1	0	0	4.242640687
5	**3**	28	1	0	2	0	0	2.236067977
6	**4**	22	2	0	1	0	1	8.062257748
7	**5**	26	1	1	0	0	0	4.242640687
8	**6**	27	2	0	1	0	0	3.16227766
9	**7**	32	3	0	1	0	1	2.828427125
10	**8**	28	2	0	1	0	0	2.236067977
11	**9**	27	1	1	1	0	1	3.16227766
12	**10**	36	1	0	1	0	0	6
13	**11**	33	1	1	0	0	1	3.31662479
14	**12**	23	1	1	1	0	0	7.071067812
15	**13**	20	1	0	1	1	0	10
16	**14**	29	1	2	0	1	1	2.449489743
17	**15**	25	1	2	0	0	0	5.477225575
18	**16**	25	1	0	1	0	0	5
19	**17**	20	1	2	2	0	1	10.24695077
20	**18**	37	3	0	1	1	1	7.280109889
21	**19**	24	1	2	0	1	1	6.403124237
22	**20**	26	1	1	1	0	0	4.123105626
23	**21**	33	2	0	0	1	1	3.31662479
24	**22**	25	1	1	2	0	0	5.196152423
25	**23**	27	1	0	0	1	1	3.16227766
26	**24**	20	1	0	2	1	1	10.04987562
27	**25**	18	1	0	1	0	0	12
28	**26**	18	1	1	2	1	1	12.08304597
29	**27**	30	1	0	1	0	0	0
30	**28**	32	1	0	2	1	1	2.236067977

50	33	3	2	1	1	0	4.123105626
51	21	2	1	0	1	1	9.16515139
52	30	3	2	2	0	0	3
53	35	1	1	0	0	0	5.196152423
54	29	2	0	1	1	1	1.414213562
55	25	2	0	1	0	0	5.099019514
56	32	3	1	0	1	1	3.16227766
57	21	1	0	0	0	1	9.055385138
58	26	1	0	2	0	1	4.123105626
59	30	2	1	2	1	1	1.732050808
60	22	1	2	2	0	0	8.306623863
61	19	1	0	1	0	1	11
62	32	2	0	0	0	1	2.449489743
63	32	2	0	1	1	1	2.236067977
64	31	1	2	2	1	0	2.449489743
65	35	2	0	1	0	1	5.099019514
66	28	3	0	1	0	1	2.828427125
67	29	2	0	1	1	0	1.414213562
68	25	1	0	0	0	1	5.099019514
69	27	2	2	0	0	0	3.872983346
70	17	1	0	0	0	1	13.03840481
71	29	1	2	0	1	1	2.449489743
72	28	2	0	1	0	0	2.236067977
73	32	3	0	1	1	0	2.828427125
74	38	3	2	2	1	1	8.544003745
75	27	2	1	1	0	0	3.31662479
76	33	4	0	1	0	1	4.242640687
77	29	2	1	2	0	1	2
78	25	1	2	0	0	1	5.477225575
79	24	2	2	1	0	0	6.403124237