Data Mining and Web algorithm

Lab Assignment 4

[07 Feb – 12 Mar, 2022]

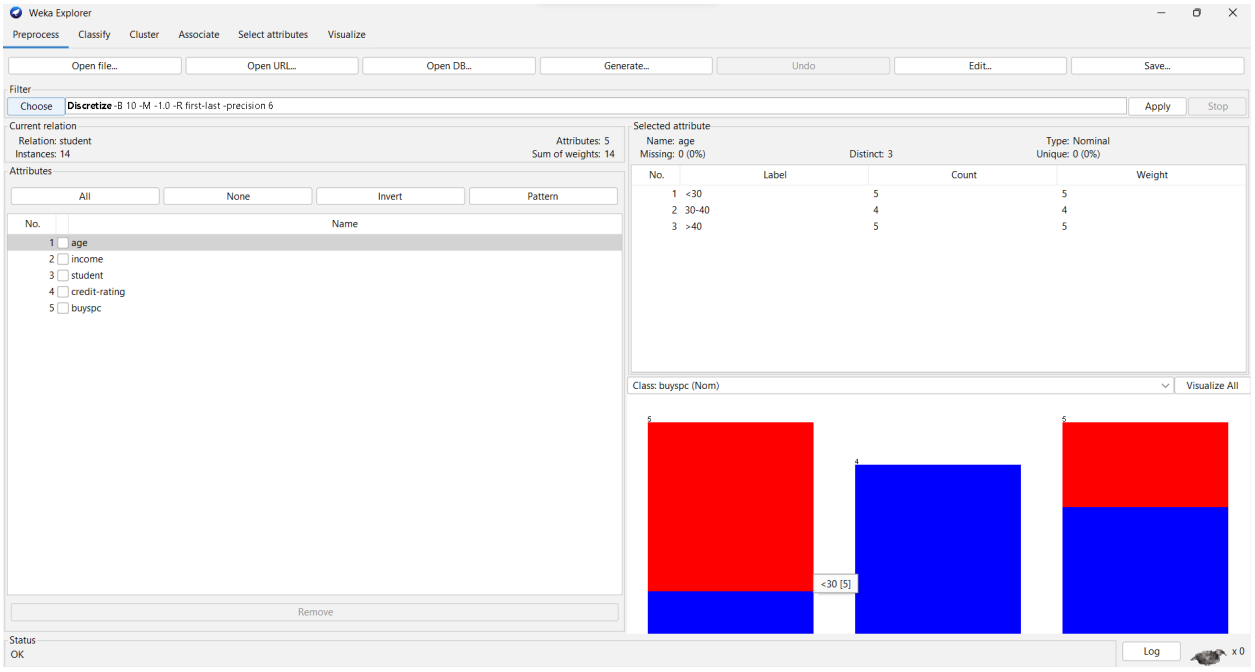
Patil Amit Gurusidhappa
19104004
B11

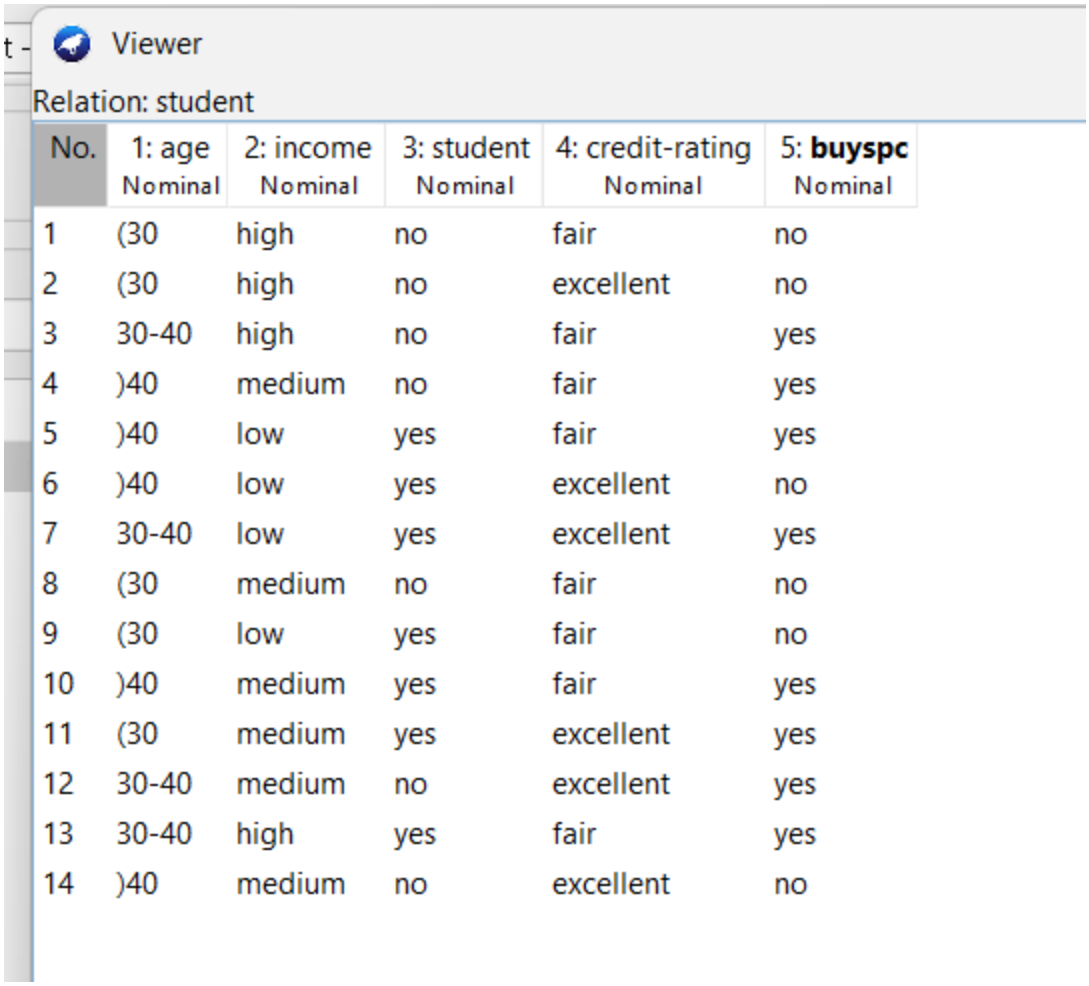
Association Rule Mining

Q1: Apply discretization filters as illustrated above on numerical attributes and run the Apriori association rule algorithm.

Steps for run Apriori algorithm in WEKA

a. Load student.arff dataset





Viewer

Relation: student

No.	1: age Nominal	2: income Nominal	3: student Nominal	4: credit-rating Nominal	5: **buyspc** Nominal
1	(30	high	no	fair	no
2	(30	high	no	excellent	no
3	30-40	high	no	fair	yes
4	)40	medium	no	fair	yes
5	)40	low	yes	fair	yes
6	)40	low	yes	excellent	no
7	30-40	low	yes	excellent	yes
8	(30	medium	no	fair	no
9	(30	low	yes	fair	no
10	)40	medium	yes	fair	yes
11	(30	medium	yes	excellent	yes
12	30-40	medium	no	excellent	yes
13	30-40	high	yes	fair	yes
14	)40	medium	no	excellent	no

b. Choose filter button and select the Unsupervised-Discretize option and apply
c. Click on Associate tab and Choose Apriori algorithm
d. Optional: In order to change the parameters for the run (example support, confidence etc.) we click on the text box immediately to the right of the choose button.
e. Click on start button
f. Rules are generated. Derive interesting insights and observe the effect of discretization in the rule generation process.

```
Best rules found:

 1. age=30-40 4 ==> buyspc=yes 4    <conf:(1)> lift:(1.75) lev:(0.12) [1] conv:(1.71)
 2. income=low 4 ==> student=yes 4    <conf:(1)> lift:(2) lev:(0.14) [2] conv:(2)
 3. age=<30 student=no 3 ==> buyspc=no 3    <conf:(1)> lift:(2.33) lev:(0.12) [1] conv:(1.71)
 4. credit-rating=fair buyspc=no 3 ==> age=<30 3    <conf:(1)> lift:(2.8) lev:(0.14) [1] conv:(1.93)
 5. age=<30 credit-rating=fair 3 ==> buyspc=no 3    <conf:(1)> lift:(2.33) lev:(0.12) [1] conv:(1.71)
 6. age=>40 buyspc=yes 3 ==> credit-rating=fair 3    <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 7. age=>40 credit-rating=fair 3 ==> buyspc=yes 3    <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 8. age=<30 income=high 2 ==> student=no 2    <conf:(1)> lift:(2) lev:(0.07) [1] conv:(1)
 9. income=high buyspc=no 2 ==> age=<30 2    <conf:(1)> lift:(2.8) lev:(0.09) [1] conv:(1.29)
10. age=<30 income=high 2 ==> buyspc=no 2    <conf:(1)> lift:(2.33) lev:(0.08) [1] conv:(1.14)
```

Q2: Write an Apriori algorithm in python. Use this link for help: https://towardsdatascience.com/apriori-association-rule-mining-explanation-and-Python-implementation-290b42afdfc6

```
import numpy as np
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules ,fpgrowth
```

```
store_data=pd.read_excel('E:\Work\JIIT\sem_6\JIIT-SEM-6\DataMining&WebAlgorithms\LabTest1_Practice\Online Retail.xlsx')
store_data
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	Uni
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	
2	536365	84406B	CREAM CUPID HEARTS COAT	8	2010-12-01 08:26:00	

```
store_data.columns
```

```
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity',
'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

```
store_data.isnull().sum()
```

```
InvoiceNo            0
StockCode            0
Description       1455
Quantity        331072
InvoiceDate          0
UnitPrice            0
CustomerID      134697
Country              0
dtype: int64
```

```
# stripping extra space in description
store_data['Description']=store_data['Description'].str.strip()

# dropping the rows without any invoice number
store_data.dropna(axis=0,subset=['InvoiceNo'],inplace=True)
store_data['InvoiceNo']=store_data['InvoiceNo'].astype('str')

# Drop all transactions which are done on credit
store_data=store_data[~store_data['InvoiceNo'].str.contains('C')]
```

```
store_data.Country.unique()
```

```
array(['United Kingdom', 'France', 'Australia', 'Netherlands',
'Germany',
       'Norway', 'EIRE', 'Switzerland', 'Spain', 'Poland',
'Portugal',
       'Italy', 'Belgium', 'Lithuania', 'Japan', 'Iceland',
       'Channel Islands', 'Denmark', 'Cyprus', 'Sweden', 'Finland',
       'Austria', 'Bahrain', 'Israel', 'Greece', 'Hong Kong',
'Singapore',
       'Lebanon', 'United Arab Emirates', 'Saudi Arabia',
       'Czech Republic', 'Canada', 'Unspecified', 'Brazil', 'USA',
       'European Community', 'Malta', 'RSA'], dtype=object)
```

```python
# Defination of 1 hot coding
def hot_encode(x):
    if(x<=0):
        return 0
    if(x>=1):
        return 1
```

```python
# Splitting the data according to the region of transaction
# Transactions done in France
basket_France = (store_data[store_data['Country'] =="France"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
basket_France
```

Description	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 EGG HOUSE PAINTED WOOD	12 MESSAGE CARDS WITH ENVELOPES	12 PENCII SMALL TUBI WOODLANI
InvoiceNo					
536370	0.0	0.0	0.0	0.0	0.(
536852	0.0	0.0	0.0	0.0	0.(
536974	0.0	0.0	0.0	0.0	0.(
537065	0.0	0.0	0.0	0.0	0.(
537463	0.0	0.0	0.0	0.0	0.(
...	...	...	...	...	..
580986	0.0	0.0	0.0	0.0	0.(

```
encoded_data=basket_France.applymap(hot_encode)
basket_France=encoded_data
basket_France.head()
```

Description	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 EGG HOUSE PAINTED WOOD	12 MESSAGE CARDS WITH ENVELOPES	12 PENCII SMALL TUBI WOODLANI
InvoiceNo					
536370	0	0	0	0	(
536852	0	0	0	0	(
536974	0	0	0	0	(
537065	0	0	0	0	(
537463	0	0	0	0	(

```
# Building the model
```

```
frq_items = apriori(basket_France, min_support = 0.1, use_colnames = True)

# Collecting the inferred rules in a dataframe
rules = association_rules(frq_items, metric ="lift", min_threshold = 1)
rules = rules.sort_values(['confidence', 'lift'], ascending =[False,
False])
print(rules.head())
```

```
                                antecedents                       consequents  \
41          (SET/6 RED SPOTTY PAPER PLATES)    (SET/6 RED SPOTTY PAPER CUPS)
44  (SET/6 RED SPOTTY PAPER PLATES, POSTAGE)    (SET/6 RED SPOTTY PAPER CUPS)
34       (STRAWBERRY LUNCH BOX WITH CUTLERY)                         (POSTAGE)
26      (ROUND SNACK BOXES SET OF4 WOODLAND)                         (POSTAGE)
40            (SET/6 RED SPOTTY PAPER CUPS)  (SET/6 RED SPOTTY PAPER PLATES)

    antecedent support  consequent support   support  confidence      lift  \
41            0.127551            0.137755  0.122449    0.960000  6.968889
44            0.107143            0.137755  0.102041    0.952381  6.913580
34            0.122449            0.765306  0.114796    0.937500  1.225000
26            0.158163            0.765306  0.147959    0.935484  1.222366
```

Q3: Perform Discretization by binning with following approach:
a. Smoothing by bin means(in python)
b. Smoothing by bin median(in python)

```
import numpy as np
import math
from sklearn.datasets import load_iris
from sklearn import datasets, linear_model, metrics


dataset = load_iris()
a = dataset.data
b = np.zeros(150)


for i in range (150):
    b[i]=a[i,1]

b=np.sort(b)
```

```
bin1=np.zeros((30,5))
bin2=np.zeros((30,5))
bin3=np.zeros((30,5))


for i in range (0,150,5):
    k=int(i/5)
    mean=(b[i] + b[i+1] + b[i+2] + b[i+3] + b[i+4])/5
    for j in range(5):
        bin1[k,j]=mean
print("Bin Mean: \n",bin1)


for i in range (0,150,5):
    k=int(i/5)
    for j in range (5):
        if (b[i+j]-b[i]) < (b[i+4]-b[i+j]):
            bin2[k,j]=b[i]
        else:
            bin2[k,j]=b[i+4]
print("Bin Boundaries: \n",bin2)


for i in range (0,150,5):
    k=int(i/5)
    for j in range (5):
        bin3[k,j]=b[i+2]
print("Bin Median: \n",bin3)
```

c. Smoothing by bin boundary(in weka)

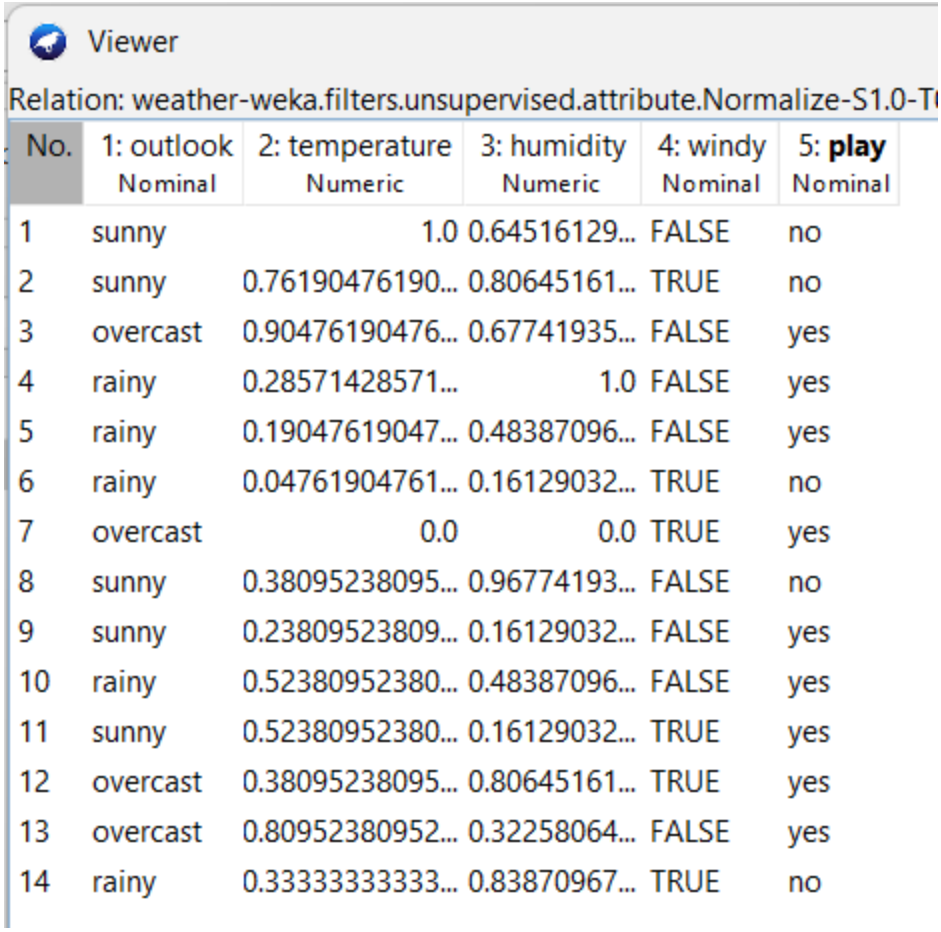
Before Normalization

Viewer

Relation: weather

No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: **play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

After Normalization



Viewer

Relation: weather-weka.filters.unsupervised.attribute.Normalize-S1.0-T

No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: **play** Nominal
1	sunny	1.0	0.64516129...	FALSE	no
2	sunny	0.76190476190...	0.80645161...	TRUE	no
3	overcast	0.90476190476...	0.67741935...	FALSE	yes
4	rainy	0.28571428571...	1.0	FALSE	yes
5	rainy	0.19047619047...	0.48387096...	FALSE	yes
6	rainy	0.04761904761...	0.16129032...	TRUE	no
7	overcast	0.0	0.0	TRUE	yes
8	sunny	0.38095238095...	0.96774193...	FALSE	no
9	sunny	0.23809523809...	0.16129032...	FALSE	yes
10	rainy	0.52380952380...	0.48387096...	FALSE	yes
11	sunny	0.52380952380...	0.16129032...	TRUE	yes
12	overcast	0.38095238095...	0.80645161...	TRUE	yes
13	overcast	0.80952380952...	0.32258064...	FALSE	yes
14	rainy	0.33333333333...	0.83870967...	TRUE	no

Q4: Run the FP Growth association rule algorithm in weka.
You may refer below link:
https://medium.com/@easpex/pitfalls-of-using-fp-growth-algorithm-in-weka-dbb34090 123a