Abhishek Murthy
21BDS0064

Fall Sem 2024-2025

DA - 2

Data Mining Lab

20-08-2024

DA Part-1

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import pandas as pd
```

```
df = pd.read_csv('/content/drive/MyDrive/missing data.csv')
```

```
# a. Find out how many data is missing in each attribute
df.isnull().sum()
```

```
                 0
Suburb           0
Address          0
Rooms            0
Type             0
Price            0
Method           0
SellerG          0
Date             0
Distance         0
Postcode         0
Bedroom2         0
Bathroom         0
Car             62
Landsize         0
BuildingArea  6450
YearBuilt     5375
CouncilArea   1369
Lattitude        0
Longtitude       0
Regionname       0
Propertycount    0
```

```
# b. For all the missing values in ‘car’ attribute, fill the missing value with the mode.
df['Car'].fillna(df['Car'].mode())
```

	Car
0	1.0
1	0.0
2	0.0
3	1.0
4	2.0
...	...
13575	2.0
13576	2.0
13577	4.0
13578	5.0
13579	1.0

13580 rows × 1 columns

```
# c. For the ‘BuildingArea’ attribute, fill the missing value with the linear interpolation and quadratic interpolation.
df['BuildingArea'] = df['BuildingArea'].interpolate(method='linear')
df['BuildingArea'] = df['BuildingArea'].interpolate(method='quadratic')
```

```
# d. Fill the ‘yearbuilt’ attribute with forward fill approach
df['YearBuilt'] = df['YearBuilt'].fillna(method='ffill')
```

```
<ipython-input-7-d2271aad8fe2>:2: FutureWarning: Series.fillna with 'method' is deprecated and will raise in a future version. Use
  df['YearBuilt'] = df['YearBuilt'].fillna(method='ffill')
```

```
# e. Remove all the rows which doesn’t have a councilarea data.
df = df.dropna(subset=['CouncilArea'])
```

```
df.isnull().sum()
```

	0
Suburb	0
Address	0
Rooms	0
Type	0
Price	0
Method	0
SellerG	0
Date	0
Distance	0
Postcode	0
Bedroom2	0
Bathroom	0
Car	0
Landsize	0
BuildingArea	1
YearBuilt	1
CouncilArea	0
Lattitude	0
Longtitude	0
Regionname	0
Propertycount	0

DA-Part 2

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


transactions = [
    ['Milk', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
    ['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
    ['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
    ['Milk', 'Unicorn', 'Corn', 'Kidney Beans', 'Yogurt'],
    ['Corn', 'Onion', 'Onion', 'Kidney Beans', 'Ice cream', 'Eggs']
    ]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
```

```
# Find the frequent itemset from the dataset with support count = 4 and support
# count = 2.
# b. Generate the association rules for support count 2 using the two methods
# ‘confidence’ and ‘lift’.

# encoding transactions
t_encoder = TransactionEncoder()
t_encoder_final = t_encoder.fit(transactions).transform(transactions)
df = pd.DataFrame(t_encoder_final, columns=t_encoder.columns_)

# min support for frequent itemsets
frequent_itemsets_4 = apriori(df, min_support=4/6, use_colnames=True)
frequent_itemsets_2 = apriori(df, min_support=2/6, use_colnames=True)

# generate association rules
rules_confidence = association_rules(frequent_itemsets_2, metric="confidence", min_threshold=0.6)
rules_lift = association_rules(frequent_itemsets_2, metric="lift", min_threshold=1.0)

print("Frequent Itemsets with Support Count 4:")
print(frequent_itemsets_4[frequent_itemsets_4['support'] * len(transactions) == 4])

print("\nFrequent Itemsets with Support Count 2:")
print(frequent_itemsets_2[frequent_itemsets_2['support'] * len(transactions) == 2])
```

```
Frequent Itemsets with Support Count 4:
   support              itemsets
0      0.8                (Eggs)
2      0.8  (Kidney Beans, Eggs)

Frequent Itemsets with Support Count 2:
    support                                      itemsets
0       0.4                                        (Corn)
4       0.4                                      (Nutmeg)
7       0.4                          (Kidney Beans, Corn)
9       0.4                                  (Eggs, Milk)
10      0.4                                (Nutmeg, Eggs)
12      0.4                                (Eggs, Yogurt)
14      0.4                        (Kidney Beans, Nutmeg)
17      0.4                                (Milk, Yogurt)
18      0.4                               (Nutmeg, Onion)
19      0.4                              (Nutmeg, Yogurt)
20      0.4                               (Yogurt, Onion)
21      0.4                    (Kidney Beans, Eggs, Milk)
22      0.4                  (Kidney Beans, Eggs, Nutmeg)
24      0.4                  (Kidney Beans, Eggs, Yogurt)
25      0.4                         (Nutmeg, Eggs, Onion)
26      0.4                        (Nutmeg, Eggs, Yogurt)
27      0.4                         (Eggs, Yogurt, Onion)
28      0.4                  (Kidney Beans, Milk, Yogurt)
29      0.4                 (Kidney Beans, Nutmeg, Onion)
30      0.4                (Kidney Beans, Nutmeg, Yogurt)
31      0.4                 (Kidney Beans, Yogurt, Onion)
32      0.4                       (Nutmeg, Yogurt, Onion)
33      0.4           (Kidney Beans, Eggs, Nutmeg, Onion)
34      0.4          (Kidney Beans, Eggs, Nutmeg, Yogurt)
35      0.4           (Kidney Beans, Eggs, Yogurt, Onion)
36      0.4                 (Nutmeg, Eggs, Yogurt, Onion)
37      0.4         (Kidney Beans, Nutmeg, Yogurt, Onion)
38      0.4  (Nutmeg, Yogurt, Onion, Eggs, Kidney Beans)
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
```

```
print("\nAssociation Rules with Confidence:")
print(rules_confidence)

print("\nAssociation Rules with Lift:")
print(rules_lift)
```

```
0        0.00        inf      0.000000
1        0.00        1.0      0.000000
2        0.00        inf      0.000000
3       -0.08        0.6     -0.333333
4        0.08        inf      0.333333
..        ...        ...           ...
160      0.16        1.8      1.000000
161      0.16        1.8      1.000000
162      0.24        inf      1.000000
163      0.16        1.8      1.000000
164      0.16        1.8      1.000000

[165 rows x 10 columns]

Association Rules with Lift:
       antecedents                            consequents  \
0   (Kidney Beans)                                 (Corn)
1           (Corn)                         (Kidney Beans)
2   (Kidney Beans)                                 (Eggs)
3           (Eggs)                         (Kidney Beans)
4         (Nutmeg)                                 (Eggs)
..             ...                                    ...
183       (Nutmeg)    (Kidney Beans, Eggs, Yogurt, Onion)
184       (Yogurt)    (Nutmeg, Eggs, Kidney Beans, Onion)
185        (Onion)   (Nutmeg, Eggs, Kidney Beans, Yogurt)
186         (Eggs)  (Nutmeg, Kidney Beans, Yogurt, Onion)
187 (Kidney Beans)          (Nutmeg, Eggs, Yogurt, Onion)

     antecedent support  consequent support  support  confidence      lift  \
0                   1.0                 0.4      0.4    0.400000  1.000000
1                   0.4                 1.0      0.4    1.000000  1.000000
2                   1.0                 0.8      0.8    0.800000  1.000000
3                   0.8                 1.0      0.8    1.000000  1.000000
4                   0.4                 0.8      0.4    1.000000  1.250000
..                  ...                 ...      ...         ...       ...
183                 0.4                 0.4      0.4    1.000000  2.500000
184                 0.6                 0.4      0.4    0.666667  1.666667
185                 0.6                 0.4      0.4    0.666667  1.666667
186                 0.8                 0.4      0.4    0.500000  1.250000
187                 1.0                 0.4      0.4    0.400000  1.000000

     leverage  conviction  zhangs_metric
0        0.00        1.0       0.000000
1        0.00        inf       0.000000
2        0.00        1.0       0.000000
3        0.00        inf       0.000000
4        0.08        inf       0.333333
..        ...        ...            ...
183      0.24        inf       1.000000
184      0.16        1.8       1.000000
185      0.16        1.8       1.000000
186      0.08        1.2       1.000000
187      0.00        1.0       0.000000

[188 rows x 10 columns]
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transfo
  and should_run_async(code)
```

DA-Part 3

```
transactions = [
    ['Milk', 'Onion', np.nan, 'Kidney Beans', 'Eggs', 'Yogurt'],
    ['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
    ['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
    ['Milk', 'Unicorn', 'Corn', np.nan, 'Yogurt'],
    ['Corn', 'Onion', np.nan, 'Kidney Beans', 'Ice cream', 'Eggs']
  ]
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
```

```
dataset_cleaned = [[item if not pd.isna(item) else 'Mayo' for item in transaction] for transaction in transactions]

t_encoder = TransactionEncoder()
t_encoder_final = t_encoder.fit(dataset_cleaned).transform(dataset_cleaned)
df = pd.DataFrame(t_encoder_final, columns=t_encoder.columns_)

frequent_itemsets_2 = apriori(df, min_support=2/len(dataset_cleaned), use_colnames=True)

print("Frequent Itemsets with Support Count 2:")
print(frequent_itemsets_2)
```

```
Frequent Itemsets with Support Count 2:
    support                             itemsets
0       0.4                               (Corn)
1       0.8                               (Eggs)
2       0.8                       (Kidney Beans)
3       0.6                               (Mayo)
4       0.6                               (Milk)
5       0.6                              (Onion)
6       0.6                             (Yogurt)
7       0.4                         (Mayo, Corn)
8       0.8                 (Kidney Beans, Eggs)
9       0.4                         (Mayo, Eggs)
10      0.4                         (Eggs, Milk)
11      0.6                        (Eggs, Onion)
12      0.4                       (Eggs, Yogurt)
13      0.4                 (Kidney Beans, Mayo)
14      0.4                 (Kidney Beans, Milk)
15      0.6                (Kidney Beans, Onion)
16      0.4               (Kidney Beans, Yogurt)
17      0.4                         (Mayo, Milk)
18      0.4                        (Mayo, Onion)
19      0.4                       (Mayo, Yogurt)
20      0.4                       (Milk, Yogurt)
21      0.4                      (Yogurt, Onion)
22      0.4           (Kidney Beans, Eggs, Mayo)
23      0.4           (Kidney Beans, Eggs, Milk)
24      0.6          (Kidney Beans, Eggs, Onion)
25      0.4         (Kidney Beans, Eggs, Yogurt)
26      0.4                  (Mayo, Eggs, Onion)
27      0.4                (Eggs, Yogurt, Onion)
28      0.4          (Kidney Beans, Mayo, Onion)
29      0.4        (Kidney Beans, Yogurt, Onion)
30      0.4                 (Mayo, Milk, Yogurt)
31      0.4    (Kidney Beans, Eggs, Mayo, Onion)
32      0.4  (Kidney Beans, Eggs, Yogurt, Onion)
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_
  and should_run_async(code)
```