Rudra_Ronit_HW2_Practicum_Q3

October 5, 2016

1 Question

Load the Extended Bakery dataset (75000-out2-binary.csv) into Python using a Pandas dataframe. Calculate the binary correlation coefficient Φ for the Chocolate Coffee and Chocolate Cake items. Show whether the two items are symmetric binary variables via their co-presence and co-absence. Would an association rule between these items as antecedent and consequent have a high confidence level? Why or why not?

2 Answer

2.1 1. Loading the Dataset

First we import all the required libraries and then use read_csv() function to read in the data.

```
In [1]: import numpy as np
        import pandas as pd
```

```
In [2]: data = pd.read_csv("75000-out2-binary.csv")
```

2.2 2. Printing out some information about the data

```
In [3]: print("Size of Dataset:",data.shape,'\n')
        print("Column Names:",data.columns,'\n')
        print("Column Types:",np.unique(data.dtypes),'\n')
        data[['Chocolate Cake','Chocolate Coffee']].head(10)
```

```
Size of Dataset: (75000, 51)

Column Names: Index(['Transaction Number', 'Chocolate Cake', 'Lemon Cake', 'Casino Cake',
       'Opera Cake', 'Strawberry Cake', 'Truffle Cake', 'Chocolate Eclair',
       'Coffee Eclair', 'Vanilla Eclair', 'Napoleon Cake', 'Almond Tart',
       'Apple Pie', 'Apple Tart', 'Apricot Tart', 'Berry Tart',
       'Blackberry Tart', 'Blueberry Tart', 'Chocolate Tart', 'Cherry Tart',
       'Lemon Tart', 'Pecan Tart', 'Ganache Cookie', 'Gongolais Cookie',
       'Raspberry Cookie', 'Lemon Cookie', 'Chocolate Meringue',
       'Vanilla Meringue', 'Marzipan Cookie', 'Tuile Cookie', 'Walnut Cookie',
       'Almond Croissant', 'Apple Croissant', 'Apricot Croissant',
       'Cheese Croissant', 'Chocolate Croissant', 'Apricot Danish',
       'Apple Danish', 'Almond Twist', 'Almond Bear', 'Blueberry Danish',
       'Lemon Lemonade', 'Raspberry Lemonade', 'Orange Juice', 'Green Tea',
       'Bottled Water', 'Hot Coffee', 'Chocolate Coffee',
       'Vanilla Frappuccino', 'Cherry Soda', 'Single Espresso'],
      dtype='object')

Column Types: [dtype('int64')]
```

```
Out[3]:    Chocolate Cake  Chocolate Coffee
        0               0                 0
        1               0                 0
        2               0                 0
        3               0                 0
        4               0                 0
        5               0                 0
        6               0                 0
        7               0                 0
        8               0                 0
        9               0                 0
```

Calculating the Contingency table is straightforward with the crosstab() function.

```
In [4]: contingency = pd.crosstab(data['Chocolate Cake']>0,data['Chocolate Coffee']>0)
        contingency

Out[4]: Chocolate Coffee  False  True
        Chocolate Cake
        False             65802   2933
        True               2962   3303
```

Correlation between these two can be easily computed using the corr attribute.

```
In [5]: data[['Chocolate Cake','Chocolate Coffee']].corr()

Out[5]:                   Chocolate Cake  Chocolate Coffee
        Chocolate Cake          1.000000          0.485566
        Chocolate Coffee        0.485566          1.000000

In [6]: data[['Chocolate Coffee','Chocolate Cake']].corr()

Out[6]:                   Chocolate Coffee  Chocolate Cake
        Chocolate Coffee          1.000000        0.485566
        Chocolate Cake            0.485566        1.000000
```

Binary Correlation is Symmetric as can be seen from tables above. They provide both auto-correlation as well as cross-correlation values. This can be checked by manually applying the correlation expression:

```
In [7]: f1_,f0_,f_1,f_0=[np.sum(contingency.iloc[1,:]),
                         np.sum(contingency.iloc[0,:]),
                         np.sum(contingency.iloc[:,1]),
                         np.sum(contingency.iloc[:,0])]

In [8]: print("Correlation is:",(75000*contingency.iloc[1,1]-f_1*f1_)/(np.sqrt(f1_*f0_*f_1*f_0)))

Correlation is: 0.485566492528
```

This symmetry is because correlation puts equal emphasis on copresence as well as coabsence.

Assuming Chocolate Cake to be X and Chocolate Coffee to be Y, we calculate the confidence of an association rule both being antecedents and subsequents in turn i.e X->Y and Y->X

```
In [9]: print("Confidence of x->y = ",contingency.iloc[1,1]/(f1_))
        print("Confidence of y->x = ",contingency.iloc[1,1]/(f_1))

Confidence of x->y =  0.527214684757
Confidence of y->x =  0.529666452854
```

Thus, both the association rules have almost the same confidence level. I could be said that if a customer buys Chocolate Cake, it is 50% probable that he will buy Chocolate Coffee too or vice versa. But, the confidence level being around halfway does not support it too much. Think of it this way, suppose a customer has bought these two items. Speaking intuitively, Chocolate Coffee and Chocolate Cake would not be anywhere near each other in a grocery store. The chocolate cake would be in the bakery/fresh baked goods section while the chocolate coffee would be on the beverages aisle. There seems to be no relationship between the two except the word "chocolate". When has someone, buying chocolate cake, thought of getting chocolate coffee too? The probability is low. If both the items are kept close to each other then it might go higher. Furthermore, there is no attribute for customer ID. This association was made purely on the amount of transactions. One inference could be that a single customer buys both these products regularly. This would mean that a unique customer is confounding and skewing the association.