Creado por:

Isabel Maniega

Market Basket Analysis

Teoría

Las **reglas de asociación** normalmente se escriben así: {Pañales} -> {Cerveza}, lo que significa que existe una fuerte relación entre los clientes que compraron pañales y también compraron cerveza en la misma transacción.

En el ejemplo anterior, {Pañal} es el antecedente y {Cerveza} es el consecuente. Tanto los antecedentes como los consecuentes pueden tener varios elementos. En otras palabras, {pañal, chicle} -> {cerveza, papas fritas} es una regla válida.

El **soporte** (support) es la frecuencia relativa con la que aparecen las reglas. En muchos casos, es posible que desee buscar un alto apoyo para asegurarse de que sea una relación útil. Sin embargo, puede haber casos en los que un soporte bajo sea útil si está tratando de encontrar relaciones "ocultas".

La **confianza** (confidence) es una medida de la fiabilidad de la regla. Una confianza de .5 en el ejemplo anterior significaría que en el 50 % de los casos en los que se compraron pañales y chicles, la compra también incluyó cerveza y papas fritas. Para la recomendación de productos, una confianza del 50 % puede ser perfectamente aceptable, pero en una situación médica, este nivel puede no ser lo suficientemente alto.

Elevación (Lift) es la relación entre el soporte observado y el esperado si las dos reglas fueran independientes (ver wikipedia). La regla general básica es que un valor de elevación cercano a 1 significa que las reglas son completamente independientes. Los valores de elevación > 1 son generalmente más "interesantes" y podrían ser indicativos de un patrón de regla útil.

In [1]:
```
# pip install xlrd
```

In [2]:
```
# pip install openpyxl
```

In [3]:
```
# pip install mlxtend
```

In [4]:
```
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

df = pd.read_excel('http://archive.ics.uci.edu/ml/machine-learning-databases/00352/Online%20Retail.xlsx')
df.head()
```

Out[4]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

In [10]:
```
df['Description'] = df['Description'].str.strip()
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')
df = df[~df['InvoiceNo'].str.contains('C')]
```

- Realizaremos un análisis para la compra en **Francia**

In [11]:
```
basket = (df[df['Country'] == "France"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
basket
```

Out[11]:

Description	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 EGG HOUSE PAINTED WOOD	12 MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	12 PENCILS SMALL TUBE RED RETROSPOT	12 PENCILS SMALL TUBE SKULL	12 PENCILS TALL TUBE POSY	12 PENCILS TALL TUBE RED RETROSPOT	12 PENCILS TALL TUBE WOODLAND	...
InvoiceNo											
536370	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
536852	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
536974	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
537065	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
537463	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
...	...	...	...	...	...	...	...	...	...	...	...
580986	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
581001	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
581171	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
581279	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
581587	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...

392 rows × 1563 columns

In [12]:
```
# convertir los menores de 0 en 0 y mayores de 1 en 1

def encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1

basket_sets = basket.applymap(encode_units)
basket_sets.drop('POSTAGE', inplace=True, axis=1)
```

In [13]:
```
# Realizamos el modelo:
frequent_itemsets = apriori(basket_sets, min_support=0.07, use_colnames=True)
```

```
/home/isabelmaniega/FEI_projects/venv/lib/python3.8/site-packages/mlxtend/frequent_patterns/fpcommon.py:111: DeprecationWarning: DataFrames with non-bool types result in worse computationalperformance and their support might be discontinued in the future.Please use a DataFrame with bool type
  warnings.warn(
```

In [14]:
```
# Realizarmos la regla de asociación:
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
rules.head()
```

Out[14]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE PINK)	0.096939	0.102041	0.073980	0.763158	7.478947	0.064088	3.791383
1	(ALARM CLOCK BAKELIKE PINK)	(ALARM CLOCK BAKELIKE GREEN)	0.102041	0.096939	0.073980	0.725000	7.478947	0.064088	3.283859
2	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.094388	0.079082	0.815789	8.642959	0.069932	4.916181
3	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.096939	0.079082	0.837838	8.642959	0.069932	5.568878
4	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE PINK)	0.094388	0.102041	0.073980	0.783784	7.681081	0.064348	4.153061

In [15]:
```
# Filtrar por los valores de lift (elevación) mayores o iguales a 6 y
# de confianza (confidence) mayores o iguales a 0.8

rules[(rules['lift'] >= 6) & (rules['confidence'] >= 0.8)]
```

Out[15]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
2	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.094388	0.079082	0.815789	8.642959	0.069932	4.916181
3	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.096939	0.079082	0.837838	8.642959	0.069932	5.568878
16	(SET/6 RED SPOTTY PAPER PLATES)	(SET/20 RED RETROSPOT PAPER NAPKINS)	0.127551	0.132653	0.102041	0.800000	6.030769	0.085121	4.336735
18	(SET/6 RED SPOTTY PAPER CUPS)	(SET/6 RED SPOTTY PAPER PLATES)	0.137755	0.127551	0.122449	0.888889	6.968889	0.104878	7.852041
19	(SET/6 RED SPOTTY PAPER PLATES)	(SET/6 RED SPOTTY PAPER CUPS)	0.127551	0.137755	0.122449	0.960000	6.968889	0.104878	21.556122
20	(SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO...	(SET/6 RED SPOTTY PAPER PLATES)	0.102041	0.127551	0.099490	0.975000	7.644000	0.086474	34.897959
21	(SET/6 RED SPOTTY PAPER CUPS, SET/6 RED SPOTTY...	(SET/20 RED RETROSPOT PAPER NAPKINS)	0.122449	0.132653	0.099490	0.812500	6.125000	0.083247	4.625850
22	(SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET...	(SET/6 RED SPOTTY PAPER CUPS)	0.102041	0.137755	0.099490	0.975000	7.077778	0.085433	34.489796

In [16]:
```
# Número de datos que contiene la regla usada: "ALARM CLOCK BAKELIKE GREEN"

basket["ALARM CLOCK BAKELIKE GREEN"].sum()
```

Out[16]:
```
340.0
```

In [17]:
```
# Número de datos que contiene la segunda regla más usada: "ALARM CLOCK BAKELIKE RED"
basket["ALARM CLOCK BAKELIKE RED"].sum()
```

Out[17]:
```
316.0
```

- Realizaremos un análisis para la compra en **Alemania**

In [19]:
```
basket2 = (df[df['Country'] =="Germany"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))

basket_sets2 = basket2.applymap(encode_units)
basket_sets2.drop('POSTAGE', inplace=True, axis=1)
frequent_itemsets2 = apriori(basket_sets2, min_support=0.05, use_colnames=True)
rules2 = association_rules(frequent_itemsets2, metric="lift", min_threshold=1)

rules2[ (rules2['lift'] >= 4) &
        (rules2['confidence'] >= 0.5)]
```

```
/home/isabelmaniega/FEI_projects/venv/lib/python3.8/site-packages/mlxtend/frequent_patterns/fpcommon.py:111: DeprecationWarning: DataFrames with non-bool types result in worse computationalperformance and their support might be discontinued in the future.Please use a DataFrame with bool type
  warnings.warn(
```

Out[19]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(PLASTERS IN TIN CIRCUS PARADE)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.115974	0.137856	0.067834	0.584906	4.242887	0.051846	2.076984
7	(PLASTERS IN TIN SPACEBOY)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.107221	0.137856	0.061269	0.571429	4.145125	0.046488	2.011670
10	(RED RETROSPOT CHARLOTTE BAG)	(WOODLAND CHARLOTTE BAG)	0.070022	0.126915	0.059081	0.843750	6.648168	0.050194	5.587746

In [20]:
```
basket2["PLASTERS IN TIN CIRCUS PARADE"].sum()
```

Out[20]:
```
774.0
```

Creado por:

Isabel Maniega