20231115_assignment6

November 15, 2023

[ ]:
```python
!pip install pandas numpy matplotlib seaborn scikit-learn==1.2.2 mlxtend


# This is to disable annoying warning messages from sklearn 1.2.2
def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn
```

```
WARNING: Ignoring invalid distribution -cikit-learn

(/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-

packages)

Requirement already satisfied: pandas in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (2.1.0)
Requirement already satisfied: numpy in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (1.26.0)
Requirement already satisfied: matplotlib in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (3.8.0)
Requirement already satisfied: seaborn in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (0.12.2)
Requirement already satisfied: scikit-learn==1.2.2 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (1.2.2)
Requirement already satisfied: mlxtend in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (0.23.0)
Requirement already satisfied: scipy>=1.3.2 in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (from scikit-learn==1.2.2) (1.11.2)
Requirement already satisfied: joblib>=1.1.1 in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (from scikit-learn==1.2.2) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
scikit-learn==1.2.2) (3.2.0)
Requirement already satisfied: python-dateutil>=2.8.2 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in /home/ugrad10/zarafat/src/data-
```

```
mining/.venv/lib/python3.10/site-packages (from pandas) (2023.3)
Requirement already satisfied: contourpy>=1.0.1 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
matplotlib) (1.1.1)
Requirement already satisfied: cycler>=0.10 in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
matplotlib) (4.42.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
matplotlib) (23.1)
Requirement already satisfied: pillow>=6.2.0 in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (from matplotlib) (10.0.1)
Requirement already satisfied: pyparsing>=2.3.1 in
/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-packages (from
matplotlib) (3.1.1)
Requirement already satisfied: six>=1.5 in /home/ugrad10/zarafat/src/data-
mining/.venv/lib/python3.10/site-packages (from python-dateutil>=2.8.2->pandas)
(1.16.0)
WARNING: Ignoring invalid distribution -cikit-learn
(/home/ugrad10/zarafat/src/data-mining/.venv/lib/python3.10/site-
packages)

[notice] A new release of pip is
available: 23.2.1 -> 23.3.1
[notice] To update, run:
pip install --upgrade pip
```

[ ]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1 Assignment 6

1.1 Ziad Arafat

1.1.1 Reading in the data

1. We read in the CSV using the pandas library and store it in a dataframe.
2. We print the data in the first two rows using the `head()` method

[ ]:
```
"""
We will use a dataset called Online Retail Data Set from the UCI repository
(https://archive.ics.uci.edu/ml/datasets/Online+Retail). Its csv file (Online␣
↪Retail.csv) can be
downloaded from Canvas. This is a transactional data set that contains all the␣
↪transactions
occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered␣
↪non-store online
retail. Note that the InvoiceNo column contains the id of the transaction and␣
↪the column StockCode
contains the ids of the items.
Based on association rules generation, what are the top 10 association rules␣
↪(please use
confidence as the rule evaluation metric) that you can find based on this␣
↪dataset?
For Association Rules Mining, you can consider using frequent_patterns package␣
↪in MLxtend.
Please check http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/
↪association_rules/ for how
to generate the association rules using MLxtend.
"""
```

[ ]:
```
'  \nWe will use a dataset called Online Retail Data Set from the UCI
repository\n(https://archive.ics.uci.edu/ml/datasets/Online+Retail). Its csv
file (Online Retail.csv) can be\ndownloaded from Canvas. This is a transactional
data set that contains all the transactions\noccurring between 01/12/2010 and
09/12/2011 for a UK-based and registered non-store online\nretail. Note that the
InvoiceNo column contains the id of the transaction and the column
StockCode\ncontains the ids of the items.\nBased on association rules
generation, what are the top 10 association rules (please use\nconfidence as the
rule evaluation metric) that you can find based on this dataset?\nFor
Association Rules Mining, you can consider using frequent_patterns package in
MLxtend.\nPlease check
http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/
for how\nto generate the association rules using MLxtend.\n'
```

[ ]:
```
df_online_retail = pd.read_csv("Online Retail.csv")
print(df_online_retail.head(n=5))

# columns: InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice,␣
↪CustomerID, Country
```

```
  InvoiceNo StockCode                          Description  Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
1    536365     71053                  WHITE METAL LANTERN         6
2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
```

```
4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6

      InvoiceDate  UnitPrice  CustomerID         Country
0  12/1/2010 8:26       2.55     17850.0  United Kingdom
1  12/1/2010 8:26       3.39     17850.0  United Kingdom
2  12/1/2010 8:26       2.75     17850.0  United Kingdom
3  12/1/2010 8:26       3.39     17850.0  United Kingdom
4  12/1/2010 8:26       3.39     17850.0  United Kingdom
```

[ ]:
```
# Step 1: Verify the format of StockCode values
print("Sample StockCode values from the DataFrame:")
print(df_online_retail['StockCode'].head())
```

```
Sample StockCode values from the DataFrame:
0    85123A
1     71053
2    84406B
3    84029G
4    84029E
Name: StockCode, dtype: object
```

1.1.2 Preprocessing

1. I created a dict that maps the stock codes to descriptions so that we can see the products in the end
2. We setup the transactions by grouping the dataset by invoice number and stockcode.

[ ]:
```
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import fpgrowth, association_rules

# Create the stockcode to description mapping dictionary
# This is to map the StockCode values to the actual description at the end
stockcode_description_mapping = pd.Series(
        df_online_retail.Description.values,
        index=df_online_retail.StockCode
).to_dict()

# Prepare the transactions data
transactions = df_online_retail.groupby('InvoiceNo')['StockCode'].apply(list)
transactions.head()
```

[ ]:
```
InvoiceNo
536365    [85123A, 71053, 84406B, 84029G, 84029E, 22752,…
536366                                    [22633, 22632]
536367    [84879, 22745, 22748, 22749, 22310, 84969, 226…
536368                      [22960, 22913, 22912, 22914]
536369                                           [21756]
Name: StockCode, dtype: object
```

Transaction encoder

1. We create a transaction encoder using our prepared transactions dataset so that it will use stockcodes as encoding

[ ]:
```
# Transaction Encoder using the actual StockCode values
te = TransactionEncoder()
te_ary = te.fit(transactions).transform(transactions)
df_transaction_encoder = pd.DataFrame(te_ary, columns=te.columns_)

df_transaction_encoder.head()
```

[ ]:
```
   10002  10080  10120  10123C  10123G  10124A  10124G  10125  10133  10134  \
0  False  False  False   False   False   False   False  False  False  False
1  False  False  False   False   False   False   False  False  False  False
2  False  False  False   False   False   False   False  False  False  False
3  False  False  False   False   False   False   False  False  False  False
4  False  False  False   False   False   False   False  False  False  False

   …      M   PADS   POST      S  gift_0001_10  gift_0001_20  gift_0001_30  \
0  …  False  False  False  False         False         False         False
1  …  False  False  False  False         False         False         False
2  …  False  False  False  False         False         False         False
3  …  False  False  False  False         False         False         False
4  …  False  False  False  False         False         False         False

   gift_0001_40  gift_0001_50      m
0         False         False  False
1         False         False  False
2         False         False  False
3         False         False  False
4         False         False  False

[5 rows x 4070 columns]
```

1.1.3 Generate Frequent Itemsets

[ ]:
```
# Generate frequent itemsets and association rules
frequent_itemsets = fpgrowth(
        df_transaction_encoder,
        min_support=0.01,
        use_colnames=True
)

frequent_itemsets.head()
```

[ ]:
```
    support  itemsets
0  0.086718  (85123A)
```

```
1  0.017915  (84029G)
2  0.016911  (84029E)
3  0.014865   (22752)
4  0.013205   (71053)
```

1.1.4 Generate association rules

1. We selected confidence as the metric and used a threshold of 0.7

[ ]:
```
rules = association_rules(
        frequent_itemsets,
        metric="confidence",
        min_threshold=0.7
)

rules.head()
```

[ ]:
```
        antecedents consequents  antecedent support  consequent support  \
0           (22745)     (22748)            0.016448            0.016988
1           (22748)     (22745)            0.016988            0.016448
2    (22726, 22728)     (22727)            0.016178            0.041737
3    (21931, 22386)    (85099B)            0.020000            0.082432
4   (21931, 85099C)    (85099B)            0.016332            0.082432

    support  confidence       lift  leverage  conviction  zhangs_metric
0  0.012124    0.737089  43.387751  0.011844    3.738955       0.993290
1  0.012124    0.713636  43.387751  0.011844    3.434626       0.993836
2  0.012124    0.749403  17.955177  0.011448    3.823924       0.959834
3  0.016062    0.803089   9.742389  0.014413    4.659804       0.915669
4  0.012317    0.754137   9.148549  0.010970    3.732030       0.905481
```

1.1.5 Display the top 10 rules sorted by confidence

1. First we transform the antecedents and consequents into the actual item descriptions
    1. This makes the data more informational than just stock codes
2. Then we can display them in a table neatly

[ ]:
```
#  Map antecedents and consequents to descriptions
rules['antecedents'] = rules['antecedents'].apply(
        lambda x: [stockcode_description_mapping[item] for item in x]
)
rules['consequents'] = rules['consequents'].apply(
        lambda x: [stockcode_description_mapping[item] for item in x]
)

# sort the rules by confidence
rules = rules.sort_values(by='confidence', ascending=False)
```

```
# Print the top 10 association rules
# only the antecedents, consequents and confidence
# Without truncating the columns
with pd.option_context('display.max_colwidth', None):
    display(rules[['antecedents', 'consequents', 'confidence']].head(n=10))
```

```
                                                                              ␣
↪    antecedents  \
71  [ROSES REGENCY TEACUP AND SAUCER , REGENCY CAKESTAND 3 TIER, PINK REGENCY␣
↪TEACUP AND SAUCER]
82                                                                   [REGENCY␣
↪TEA PLATE PINK]
25                             [SET/20 RED RETROSPOT PAPER NAPKINS , SET/6 RED␣
↪SPOTTY PAPER CUPS]
75                             [ROSES REGENCY TEACUP AND SAUCER , PINK REGENCY␣
↪TEACUP AND SAUCER]
70   [GREEN REGENCY TEACUP AND SAUCER, REGENCY CAKESTAND 3 TIER, PINK REGENCY␣
↪TEACUP AND SAUCER]
22          [JUMBO SHOPPER VINTAGE RED PAISLEY, JUMBO STORAGE BAG SUKI, JUMBO␣
↪BAG PINK POLKADOT]
83                                                                   [REGENCY␣
↪TEA PLATE PINK]
67                                     [REGENCY CAKESTAND 3 TIER, PINK REGENCY␣
↪TEACUP AND SAUCER]
10                                        [CHARLOTTE BAG PINK POLKADOT,␣
↪STRAWBERRY CHARLOTTE BAG]
17                    [WOODLAND CHARLOTTE BAG, CHARLOTTE BAG SUKI DESIGN,␣
↪STRAWBERRY CHARLOTTE BAG]

                           consequents  confidence
71   [GREEN REGENCY TEACUP AND SAUCER]    0.899110
82          [REGENCY TEA PLATE GREEN ]    0.898089
25      [SET/6 RED SPOTTY PAPER PLATES]    0.895270
75   [GREEN REGENCY TEACUP AND SAUCER]    0.894137
70  [ROSES REGENCY TEACUP AND SAUCER ]    0.875723
22           [JUMBO BAG RED RETROSPOT]    0.867742
83          [REGENCY TEA PLATE ROSES ]    0.866242
67   [GREEN REGENCY TEACUP AND SAUCER]    0.860697
10       [RED RETROSPOT CHARLOTTE BAG]    0.858639
17       [RED RETROSPOT CHARLOTTE BAG]    0.858553
```

[ ]: