```
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).
```

```
# !pip install mlxtend
```

```
#Library and Functions

import pandas as pd
import numpy as np
pd.set_option('display.max_columns', None)
pd.set_option('display.width', 500)
from mlxtend.frequent_patterns import apriori, association_rules


def outlier_thresholds(dataframe, variable):
    quartile1 = dataframe[variable].quantile(0.01)
    quartile3 = dataframe[variable].quantile(0.99)
    interquantile_range = quartile3 - quartile1
    up_limit = quartile3 + 1.5 * interquantile_range
    low_limit = quartile1 - 1.5 * interquantile_range
    return low_limit, up_limit

def replace_with_thresholds(dataframe, variable):
    low_limit, up_limit = outlier_thresholds(dataframe, variable)
    dataframe.loc[(dataframe[variable] < low_limit), variable] = low_limit
    dataframe.loc[(dataframe[variable] > up_limit), variable] = up_limit


def retail_data_prep(dataframe):
    dataframe.drop(dataframe[dataframe["StockCode"]=="POST"].index, inplace=True)
    dataframe.dropna(inplace=True)
    dataframe = dataframe[~dataframe["Invoice"].str.contains("C", na=False)]
    dataframe = dataframe[dataframe["Quantity"] > 0]
    dataframe = dataframe[dataframe["Price"] > 0]
    replace_with_thresholds(dataframe, "Quantity")
    replace_with_thresholds(dataframe, "Price")
    return dataframe
```

```
df_ = pd.read_excel(r"/content/drive/MyDrive/Online_Retail.xlsx")
df = df_.copy()
df = retail_data_prep(df)

df.head()
```

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6.0	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6.0	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8.0	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6.0	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6.0	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

New Section

```
df.tail(10)
```

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
541899	581587	22726	ALARM CLOCK BAKELIKE GREEN	4.0	2011-12-09 12:50:00	3.75	12680.0	France
541900	581587	22730	ALARM CLOCK BAKELIKE IVORY	4.0	2011-12-09 12:50:00	3.75	12680.0	France
541901	581587	22367	CHILDRENS APRON SPACEBOY DESIGN	8.0	2011-12-09 12:50:00	1.95	12680.0	France
541902	581587	22629	SPACEBOY LUNCH BOX	12.0	2011-12-09 12:50:00	1.95	12680.0	France
541903	581587	23256	CHILDRENS CUTLERY SPACEBOY	4.0	2011-12-09 12:50:00	4.15	12680.0	France
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12.0	2011-12-09 12:50:00	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6.0	2011-12-09 12:50:00	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4.0	2011-12-09 12:50:00	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4.0	2011-12-09 12:50:00	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3.0	2011-12-09 12:50:00	4.95	12680.0	France

```
#United Kingdom customers produce association rule
df = df[df['Country'] == "United Kingdom"]

df.tail(10)
# df.head()
```

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
541884	581585	84946	ANTIQUE SILVER T-LIGHT GLASS	12.0	2011-12-09 12:31:00	1.25	15804.0	United Kingdom
541885	581585	21684	SMALL MEDINA STAMPED METAL BOWL	12.0	2011-12-09 12:31:00	0.85	15804.0	United Kingdom
541886	581585	22398	MAGNETS PACK OF 4 SWALLOWS	12.0	2011-12-09 12:31:00	0.39	15804.0	United Kingdom
541887	581585	23328	SET 6 SCHOOL MILK BOTTLES IN CRATE	4.0	2011-12-09 12:31:00	3.75	15804.0	United Kingdom
541888	581585	23145	ZINC T-LIGHT HOLDER STAR LARGE	12.0	2011-12-09 12:31:00	0.95	15804.0	United Kingdom

```
# all possible products will be put in the columns
# If there is a product in the cart, the value will be 1, otherwise 0 will be assigned

df.groupby(['Invoice', 'Description']).agg({"Quantity": "sum"}).head(20)

df.groupby(['Invoice', 'Description']).agg({"Quantity": "sum"}).unstack()

df.groupby(['Invoice', 'Description']).agg({"Quantity": "sum"}).unstack().fillna(0)

invoice_product_df = df.groupby(['Invoice', 'Description'])['Quantity'].sum().unstack().fillna(0).applymap(lambda x: 1 if x > 0 else 0)


def create_invoice_product_df(dataframe, id=False):
    if id:
        return dataframe.groupby(['Invoice', "StockCode"])['Quantity'].sum().unstack().fillna(0). \
            applymap(lambda x: 1 if x > 0 else 0)
    else:
        return dataframe.groupby(['Invoice', 'Description'])['Quantity'].sum().unstack().fillna(0). \
            applymap(lambda x: 1 if x > 0 else 0)
```

```
invoice_product_df = create_invoice_product_df(df)
invoice_product_df.head()
```

Description	4 PURPLE FLOCK DINNER CANDLES	50'S CHRISTMAS GIFT BAG LARGE	DOLLY GIRL BEAKER	I LOVE LONDON MINI BACKPACK	NINE DRAWER OFFICE TIDY	OVAL WALL MIRROR DIAMANTE	RED SPOT GIFT BAG LARGE	SET 2 TEA TOWELS I LOVE LONDON	SPACEBOY BABY GIFT SET	TOADS BED L
Invoice										
536365	0	0	0	0	0	0	0	0	0	
536366	0	0	0	0	0	0	0	0	0	
536367	0	0	0	0	0	0	0	0	0	
536368	0	0	0	0	0	0	0	0	0	
536369	0	0	0	0	0	0	0	0	0	

```
frequent_itemsets = apriori(invoice_product_df, min_support=0.01, use_colnames=True)

frequent_itemsets.tail(20)
```

	support	itemsets
950	0.010945	(LUNCH BAG WOODLAND, LUNCH BAG CARS BLUE, LUNC...
951	0.012989	(LUNCH BAG CARS BLUE, LUNCH BAG RED RETROSPOT,...
952	0.012989	(LUNCH BAG SUKI DESIGN , LUNCH BAG CARS BLUE, ...
953	0.012448	(LUNCH BAG WOODLAND, LUNCH BAG CARS BLUE, LUNC...
954	0.011486	(LUNCH BAG SUKI DESIGN , LUNCH BAG CARS BLUE, ...
955	0.011727	(LUNCH BAG WOODLAND, LUNCH BAG CARS BLUE, LUNC...
956	0.010043	(LUNCH BAG SUKI DESIGN , LUNCH BAG WOODLAND, L...
957	0.011185	(LUNCH BAG DOLLY GIRL DESIGN, LUNCH BAG RED RE...
958	0.013771	(LUNCH BAG RED RETROSPOT, LUNCH BAG SPACEBOY D...
959	0.013952	(LUNCH BAG SUKI DESIGN , LUNCH BAG RED RETROSP...
960	0.012749	(LUNCH BAG WOODLAND, LUNCH BAG RED RETROSPOT, ...
961	0.010945	(LUNCH BAG WOODLAND, LUNCH BAG SPACEBOY DESIGN...
962	0.012629	(LUNCH BAG SUKI DESIGN , LUNCH BAG RED RETROSP...
963	0.013410	(LUNCH BAG WOODLAND, LUNCH BAG RED RETROSPOT, ...
964	0.011787	(LUNCH BAG SUKI DESIGN , LUNCH BAG WOODLAND, L...
965	0.013531	(REGENCY CAKESTAND 3 TIER, PINK REGENCY TEACUP...
966	0.010704	(WOODEN PICTURE FRAME WHITE FINISH, WOODEN FRA...
967	0.010223	(WOODEN STAR CHRISTMAS SCANDINAVIAN, WOODEN TR...
968	0.012147	(GREEN REGENCY TEACUP AND SAUCER, REGENCY CAKE...
969	0.011125	(LUNCH BAG BLACK SKULL., LUNCH BAG CARS BLUE,...

```
rules = association_rules(frequent_itemsets, metric="support", min_threshold=0.01)

rules.tail()
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
965	(LUNCH BAG RED RETROSPOT, LUNCH BAG PINK POLKA...	(LUNCH BAG CARS BLUE, LUNCH BAG BLACK SKULL.)	0.028324	0.024175	0.011125	0.392781	16.247663	0.010440
966	(LUNCH BAG BLACK SKULL.)	(LUNCH BAG CARS BLUE, LUNCH BAG RED RETROSPOT,...	0.059895	0.015275	0.011125	0.185743	12.160314	0.010210
967	(LUNCH BAG CARS BLUE)	(LUNCH BAG PINK POLKADOT, LUNCH BAG	0.053040	0.017620	0.011125	0.209751	11.904240	0.010191

```
def create_rules(dataframe, id=True, country="France"):
    dataframe = dataframe[dataframe['Country'] == country]
    dataframe = create_invoice_product_df(dataframe, id)
    frequent_itemsets = apriori(dataframe, min_support=0.01, use_colnames=True)
    rules = association_rules(frequent_itemsets, metric="support", min_threshold=0.01)
    return rules
```

```
rules_uk = create_rules(df, country="United Kingdom")
```

```
rules_uk
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	c
0	(20712)	(22386)	0.022070	0.048830	0.010043	0.455041	9.318811	0.008965	
1	(22386)	(20712)	0.048830	0.022070	0.010043	0.205665	9.318811	0.008965	
2	(20712)	(85099B)	0.022070	0.087017	0.012087	0.547684	6.294012	0.010167	
3	(85099B)	(20712)	0.087017	0.022070	0.012087	0.138908	6.294012	0.010167	
4	(85099B)	(20713)	0.087017	0.022310	0.010885	0.125086	5.606635	0.008943	
...	...	...	...	...	...	...	...	...	
1205	(22699, 22423)	(22697, 22698)	0.021168	0.024295	0.012147	0.573864	23.620739	0.011633	
1206	(22697)	(22698, 22699, 22423)	0.036803	0.013531	0.012147	0.330065	24.394031	0.011649	
1207	(22698)	(22697, 22699, 22423)	0.029647	0.015936	0.012147	0.409736	25.711340	0.011675	

```
def check_id(dataframe, stock_code):
    product_name = dataframe[dataframe["StockCode"] == stock_code][["Description"]].values[0].tolist()
    print(product_name)

df[df["StockCode"] == 71053][["Description"]].values[0].tolist()
```

```
['WHITE METAL LANTERN']
```

```
check_id(df,22632)
```

```
['HAND WARMER RED POLKA DOT']
```

```
df[df["StockCode"] == 22899][["Description"]].values[0].tolist()
```

```
["CHILDREN'S APRON DOLLY GIRL "]
```

```
df[df["StockCode"] == "84029G"][["Description"]].values[0].tolist()
```

```
['KNITTED UNION FLAG HOT WATER BOTTLE']
```

```
def arl_recommender(rules_df, product_id, rec_count=3):
    sorted_rules = rules_df.sort_values("lift", ascending=False)
    recommendation_list = []
    for i, product in sorted_rules["antecedents"].items():
        for j in list(product):
            if j == product_id:
                recommendation_list.append(list(sorted_rules.iloc[i]["consequents"]))
    recommendation_list = list({item for item_list in recommendation_list for item in item_list})
    return recommendation_list[:rec_count]


sorted_rules = rules_uk.sort_values("lift", ascending=False)

for i, product in sorted_rules["antecedents"].items():
    print(i,list(product))
    break
```

```
506 [22916]
```

```
arl_recommender(rules_uk, 20712, 1)
check_id(df, arl_recommender(rules_uk, 20712, 1)[0])
```

```
['REGENCY TEA PLATE GREEN ']
```

```
arl_recommender(rules_uk, 22916, 1)
check_id(df, arl_recommender(rules_uk, 22916, 1)[0])
```

```
['JUMBO STORAGE BAG SUKI']
```

```
arl_recommender(rules_uk, 22699, 1)
check_id(df, arl_recommender(rules_uk, 22699, 1)[0])
```

```
['JUMBO BAG DOILEY PATTERNS']
```

```
arl_recommender(rules_uk, 22699, 3)
check_id(df, arl_recommender(rules_uk, 22699, 3)[0])
check_id(df, arl_recommender(rules_uk, 22699, 3)[1])
check_id(df, arl_recommender(rules_uk, 22699, 3)[2])
```

```
['JUMBO BAG DOILEY PATTERNS']
['LUNCH BAG APPLE DESIGN']
['LUNCH BAG DOILEY PATTERN ']
```

```
arl_recommender(rules_uk, "85099B", 1)
check_id(df, arl_recommender(rules_uk, "85099B", 1)[0])
```

```
['SPOTTY BUNTING']
```

```
arl_recommender(rules_uk, "85123A", 1)
check_id(df, arl_recommender(rules_uk, "85123A", 1)[0])
```

```
['JUMBO SHOPPER VINTAGE RED PAISLEY']
```

✓ 0s completed at 11:24 AM