In [2]:
```
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
dataset = pd.read_csv('Bakery.csv', header=None)
print(dataset.head())
```

```
              0             1                    2        3        4
0  TransactionNo         Items             DateTime  Daypart  DayType
1              1         Bread  2016-10-30 09:58:11  Morning  Weekend
2              2   Scandinavian  2016-10-30 10:05:34  Morning  Weekend
3              2   Scandinavian  2016-10-30 10:05:34  Morning  Weekend
4              3  Hot chocolate  2016-10-30 10:07:57  Morning  Weekend
```

In [4]:
```
transactions = []
for i in range(0, dataset.shape[0]):
    transactions.append([str(dataset.values[i, j]) for j in range(0, dataset.shape[1]) if str(dataset.values[i, j]) != 'nan'])
```

In [5]:
```
print(f"Total transactions: {len(transactions)}")
```

```
Total transactions: 20508
```

In [6]:
```
print("Sample transaction:", transactions[0])
```

```
Sample transaction: ['TransactionNo', 'Items', 'DateTime', 'Daypart', 'DayType']
```

In [8]:
```
te = TransactionEncoder()
te_ary = te.fit(transactions).transform(transactions)
df = pd.DataFrame(te_ary, columns=te.columns_)
print(df.head())
```

```
      1     10    100   1000   1001   1002   1003   1004   1005   1006  ...  \
0  False  False  False  False  False  False  False  False  False  False  ...
1   True  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  ...

   Toast  TransactionNo  Truffles  Tshirt  Valentine's card  Vegan Feast  \
0  False           True     False   False             False        False
1  False          False     False   False             False        False
2  False          False     False   False             False        False
3  False          False     False   False             False        False
4  False          False     False   False             False        False

   Vegan mincepie  Victorian Sponge  Weekday  Weekend
0           False             False    False    False
1           False             False    False     True
2           False             False    False     True
3           False             False    False     True
4           False             False    False     True

[5 rows x 19035 columns]
```

In [9]:
```
frequent_itemsets = apriori(df, min_support=0.01, use_colnames=True)
frequent_itemsets.sort_values(by='support', ascending=False, inplace=True)
```

In [10]:
```
print("\nTop Frequent Itemsets:")
print(frequent_itemsets.head(10))
```

```
Top Frequent Itemsets:
     support              itemsets
21  0.624488             (Weekday)
0   0.564121           (Afternoon)
12  0.409791             (Morning)
22  0.375463             (Weekend)
37  0.354642  (Weekday, Afternoon)
5   0.266774              (Coffee)
64  0.252292    (Weekday, Morning)
38  0.209479  (Weekend, Afternoon)
48  0.172762     (Weekday, Coffee)
2   0.162132               (Bread)
```

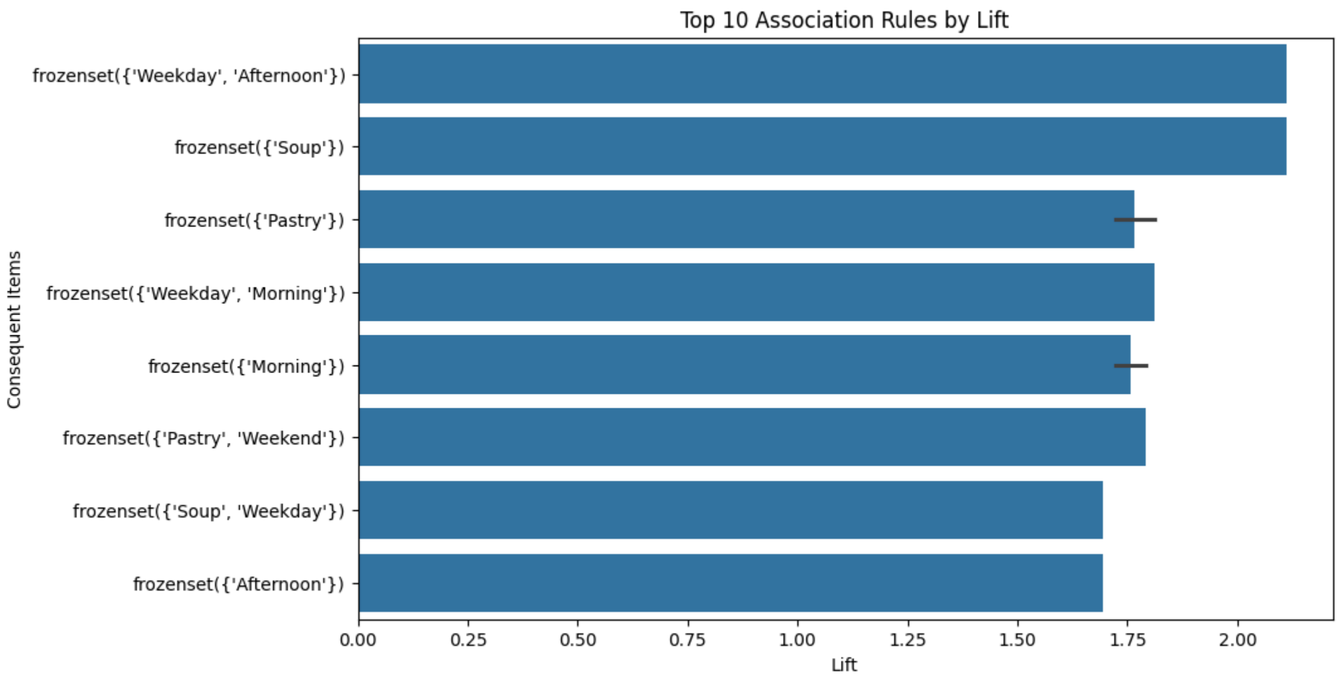
In [11]:
```
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
print("\nSample Rules:")
print(rules.head())
```

```
Sample Rules:
   antecedents  consequents  antecedent support  consequent support   support  \
0    (Weekday)  (Afternoon)            0.624488            0.564121  0.354642
1  (Afternoon)    (Weekday)            0.564121            0.624488  0.354642
2    (Weekday)     (Coffee)            0.624488            0.266774  0.172762
3     (Coffee)    (Weekday)            0.266774            0.624488  0.172762
4    (Weekend)    (Morning)            0.375463            0.409791  0.157500

   confidence      lift  representativity  leverage  conviction  \
0    0.567893  1.006685               1.0  0.002355    1.008728
1    0.628663  1.006685               1.0  0.002355    1.011243
2    0.276646  1.037004               1.0  0.006165    1.013647
3    0.647596  1.037004               1.0  0.006165    1.065574
4    0.419481  1.023644               1.0  0.003638    1.016691

   zhangs_metric   jaccard  certainty  kulczynski
0       0.017685  0.425247   0.008652    0.598278
1       0.015235  0.425247   0.011118    0.598278
2       0.095026  0.240448   0.013463    0.462121
3       0.048666  0.240448   0.061538    0.462121
4       0.036984  0.250893   0.016417    0.401911
```

In [12]:
```
plt.figure(figsize=(10,6))
sns.barplot(x='lift', y='consequents', data=rules.nlargest(10, 'lift'))
plt.title('Top 10 Association Rules by Lift')
plt.xlabel('Lift')
plt.ylabel('Consequent Items')
plt.show()
```



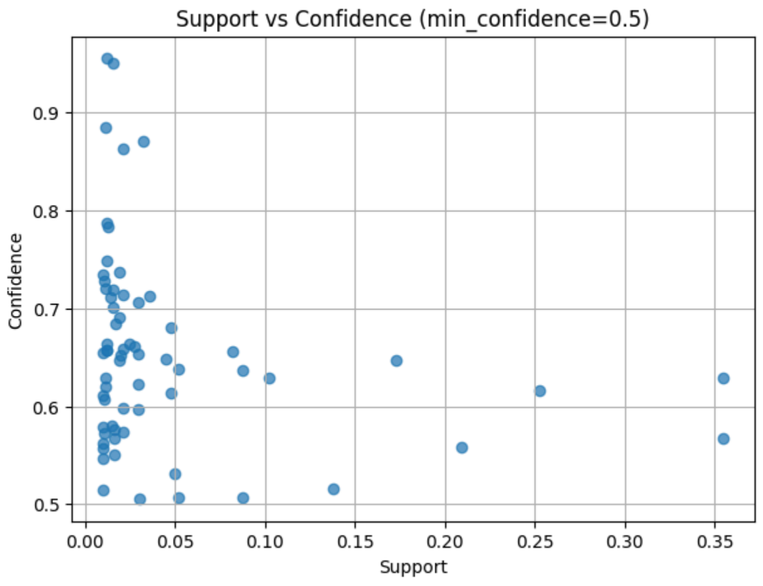
In [13]:
```
rules_high_conf = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.5)
print("\nRules with Minimum Confidence = 0.5")
print(rules_high_conf.head())
```

```
Rules with Minimum Confidence = 0.5
   antecedents  consequents  antecedent support  consequent support   support  \
0    (Weekday)  (Afternoon)            0.624488            0.564121  0.354642
1  (Afternoon)    (Weekday)            0.564121            0.624488  0.354642
2    (Morning)    (Weekday)            0.409791            0.624488  0.252292
3    (Weekend)  (Afternoon)            0.375463            0.564121  0.209479
4     (Coffee)    (Weekday)            0.266774            0.624488  0.172762

   confidence      lift  representativity  leverage  conviction  \
0    0.567893  1.006685               1.0  0.002355    1.008728
1    0.628663  1.006685               1.0  0.002355    1.011243
2    0.615659  0.985862               1.0 -0.003618    0.977029
3    0.557922  0.989011               1.0 -0.002328    0.985977
4    0.647596  1.037004               1.0  0.006165    1.065574

   zhangs_metric   jaccard  certainty  kulczynski
0       0.017685  0.425247   0.008652    0.598278
1       0.015235  0.425247   0.011118    0.598278
2      -0.023721  0.322629  -0.023511    0.509829
3      -0.017480  0.286916  -0.014222    0.464630
4       0.048666  0.240448   0.061538    0.462121
```

In [14]:
```
plt.figure(figsize=(7,5))
plt.scatter(rules_high_conf['support'], rules_high_conf['confidence'], alpha=0.7)
plt.title('Support vs Confidence (min_confidence=0.5)')
plt.xlabel('Support')
plt.ylabel('Confidence')
plt.grid(True)
plt.show()
```


Support vs Confidence (min_confidence=0.5)
Confidence
Support
0.9
0.8
0.7
0.6
0.5
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35

In [ ]: