Market Basket Analysis of Hospital Medication Prescriptions

Ashley Traore

Business Question

Which ten medications are most frequently prescribed alongside other medications, based on antecedent, consequent relationships in patient prescription patterns?

Goal

The goal of this analysis is to identify the most frequently co-prescribed medications, enabling the hospital to make informed decisions related to inventory management, clinical safety (including the detection of potentially dangerous drug combinations), and the monitoring of prescribing behaviors.

Market Basket Analysis and Expected Outcomes

Market Basket Analysis (MBA) is a data mining technique used to discover patterns or associations between items that frequently co-occur in transactional datasets. It is commonly applied to understand purchasing behavior, whether by customers in retail settings or patients in healthcare contexts. For example, in this analysis, an MBA can reveal which medications are often prescribed together during a patient's stay at the hospital.

The analysis begins by scanning the entire dataset to identify frequent item sets, or combinations of items that appear together in a significant number of transactions. The prevalence of these item sets is measured by a metric called support, which represents the proportion of records that contain a given combination (Mehta, 2024). For instance, a support value of 0.03 means the combination is present in 3% of all transactions.

Once frequent item sets are identified, the algorithm generates association rules in the form "If A occurs, then B is likely to occur." These rules are evaluated using three key metrics:

- Support – The overall frequency of the rule across the dataset.
- Confidence – The likelihood that item B appears in transactions that contain item A. For example, a confidence of 0.8 indicates that B occurs 80% of the time when A is present.
- Lift – The strength of the association between A and B, compared to what would be expected if they were independent. A lift greater than 1 implies a positive association; the items co-occur more often than chance would predict.

By evaluating and interpreting these rules, businesses and healthcare organizations can make informed decisions about product placement, inventory management, prescribing patterns, risk alerts, and more.

The expected outcome is a prioritized list of association rules that highlight the most frequent and meaningful relationships within the dataset. For instance, a rule like "If Medication A is prescribed, then Medication B is also prescribed" when supported by high values of support, confidence, and lift, can point to a consistent and clinically relevant prescribing pattern. These insights can directly inform business decisions, including inventory optimization, safety protocols, and treatment standardization. Meanwhile, rules with lower support but elevated lift and confidence may represent less common but potentially significant trends, such as emerging clinical practices or off-label prescribing behaviors. By carefully analyzing these rules, organizations can unlock data-driven insights that enhance operational efficiency, promote safer prescribing, and ultimately improve patient care outcomes

Examples of Transactions

```
# Print 10 transactions
for i, transaction in enumerate(list_of_lists[0:10], start=1):
    print(f"Transaction {i}: {transaction}")
```

```
Transaction 1: ['amlodipine', 'albuterol aerosol', 'allopurinol', 'pantoprazole', 'lorazepam', 'omeprazole', 'mometasone', 'fluconozole', 'gabapentin', 'pravastatin', 'cialis', 'losartan', 'metoprolol succinate XL', 'sulfamethoxazole', 'abilify', 'spironolactone', 'albuterol HFA', 'levofloxacin', 'prome
thazine', 'glipizide']
Transaction 2: ['citalopram', 'benicar', 'amphetamine salt combo xr']
Transaction 3: ['enalapril']
Transaction 4: ['paroxetine', 'allopurinol']
Transaction 5: ['abilify', 'atorvastatin', 'folic acid', 'naproxen', 'losartan']
Transaction 6: ['cialis']
Transaction 7: ['hydrochlorothiazide', 'glyburide']
Transaction 8: ['metformin', 'salmeterol inhaler', 'sertraline HCI']
Transaction 9: ['metoprolol', 'carvedilol', 'losartan']
Transaction 10: ['glyburide']
```

One Assumption of Market Basket Analysis

One assumption in Market Basket Analysis is that each transaction, such as a patient encounter, is treated as an independent event. This means the items selected in one transaction are assumed to have no influence on those in another, and the analysis does not account for sequence, timing, or causal relationships between transactions. (Deniran, 2023)

Data Transformation

- Import Libraries
    - Importing all necessary Python libraries to perform Market Basket Analysis, including tools for data manipulation, transformation, and mining association rules. Then, load the medication dataset into a Pandas Data Frame using the read_csv() function.

```
[1]: !python --version
```

```
Python 3.11.7
```

```
[2]: # Import Libraries
import pandas as pd
import numpy as np
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
import seaborn as sns
import matplotlib.pyplot as plt


# Change setting in pandas to display all columns
pd.options.display.max_columns = None
```

```
[3]: # Import medical dataset
medicineDF =pd.read_csv(r"C:\Users\ashle\Desktop\MSDA WGU\Data Mining 2 -D212\dataset\medical_market_basket.csv")
```

```
[4]: medicineDF
```

[4]:

	Presc01	Presc02	Presc03	Presc04	Presc05	Presc06	Presc07	Presc08	Presc09	Presc10	Presc11	Presc12	Presc13
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	amlodipine	albuterol aerosol	allopurinol	pantoprazole	lorazepam	omeprazole	mometasone	fluconozole	gabapentin	pravastatin	cialis	losartan	metoprolol succinate XL
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	citalopram	benicar	amphetamine salt combo xr	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...
14997	clopidogrel	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

- Check for Null-Only Columns and Rows
  - Inspect the Data Frame for any columns that contain only null values. In this dataset, all columns return False, confirming that none are entirely null. Next, count and display the number of rows where all values are missing. Remove these fully null rows to clean the dataset, then print the updated shape to confirm the final number of records and fields.

```
# Check for columns that only contain null values.
medicineDF.isna().all()
```

```
Presc01    False
Presc02    False
Presc03    False
Presc04    False
Presc05    False
Presc06    False
Presc07    False
Presc08    False
Presc09    False
Presc10    False
Presc11    False
Presc12    False
Presc13    False
Presc14    False
Presc15    False
Presc16    False
Presc17    False
Presc18    False
Presc19    False
Presc20    False
dtype: bool
```

```
# Count rows that only contain null values
print(f"Rows with all null values: {medicineDF.isna().all(axis=1).sum()}")
# Drop rows that contain only null values
medicine_NotNA = medicineDF.dropna(axis=0, how='all')
print(f"Shape of Data Frame after null rows have been removed; {medicine_NotNA.shape}")
```

```
Rows with all null values: 7501
Shape of Data Frame after null rows have been removed; (7501, 20)
```

- Convert Data Frame to Transaction Format
    - Transform the cleaned Data Frame into a list of lists, where each inner list represents a single transaction (for example, a list of prescriptions). While converting, exclude any null values to ensure only valid items are retained. Then iterate through the resulting list to print and review the first ten transactions individually.

```
# Convert the dataframe into a list of lists without null values.
list_of_lists = []
for i in range(len(medicine_NotNA)):
    transaction = [str(medicine_NotNA.values[i, j]) for j in range(medicine_NotNA.shape[1]) if pd.notnull(medicine_NotNA.values[i, j])]
    list_of_lists.append(transaction)
```

```
# Print 10 transactions
for i, transaction in enumerate(list_of_lists[0:10], start=1):
    print(f"Transaction {i}: {transaction}")
```

```
Transaction 1: ['amlodipine', 'albuterol aerosol', 'allopurinol', 'pantoprazole', 'lorazepam', 'omeprazole', 'mometasone', 'fluconozole', 'gabapentin', 'pravastatin', 'cialis', 'losartan', 'metoprolol succinate XL', 'sulfamethoxazole', 'abilify', 'spironolactone', 'albuterol HFA', 'levofloxacin', 'prome
thazine', 'glipizide']
Transaction 2: ['citalopram', 'benicar', 'amphetamine salt combo xr']
Transaction 3: ['enalapril']
Transaction 4: ['paroxetine', 'allopurinol']
Transaction 5: ['abilify', 'atorvastatin', 'folic acid', 'naproxen', 'losartan']
Transaction 6: ['cialis']
Transaction 7: ['hydrochlorothiazide', 'glyburide']
Transaction 8: ['metformin', 'salmeterol inhaler', 'sertraline HCI']
Transaction 9: ['metoprolol', 'carvedilol', 'losartan']
Transaction 10: ['glyburide']
```

- Encode Transactions for Analysis
    - o Use TransactionEncoder to convert the list of transactions into a binary-encoded format, where each column represents an item and each row indicates its presence or absence in the transaction. Print the resulting encoded Data Frame to verify the transformation.

```
# Encode the data using TransactionEncoder
te = TransactionEncoder()
te_array = te.fit(list_of_lists).transform(list_of_lists)
encodedDF = pd.DataFrame(te_array, columns=te.columns_)
```

```
encodedDF
```

	Duloxetine	Premarin	Yaz	abilify	acetaminophen	actonel	albuterol HFA	albuterol aerosol	alendronate	allopurinol	alprazolam	amitriptyline	amlodipine	amoxicillin	a
0	False	False	False	True	False	False	True	True	False	True	False	False	True	False	
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	True	False	False	False	False	
4	False	False	False	True	False	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
7496	False	False	False	False	False	False	False	False	False	False	False	False	False	False	
7497	False	False	False	False	False	False	False	False	False	False	False	False	False	False	
7498	False	False	False	False	False	False	False	False	False	False	False	False	False	False	
7499	False	False	False	False	False	False	False	False	False	False	True	False	False	False	
7500	False	False	False	False	False	False	False	False	False	False	False	False	False	False	

7501 rows × 119 columns

- Review Final Dataset
    - o Display all column names in the encoded Data Frame to examine the full item set. Print the shape of the dataset to report the number of transactions (rows) and distinct items (columns). Finally, export the cleaned and encoded dataset.

```
: # Display all columns in the encoded dataframe.
  list(encodedDF.columns)
```

```
: ['Duloxetine',
   'Premarin',
   'Yaz',
   'abilify',
   'acetaminophen',
   'actonel',
   'albuterol HFA',
   'albuterol aerosol',
   'alendronate',
   'allopurinol',
   'alprazolam',
   'amitriptyline',
   'amlodipine',
   'amoxicillin',
   'amphetamine',
   'amphetamine salt combo',
   'amphetamine salt combo xr',
   'atenolol',
```

```
: encodedDF.shape
```

```
: (7501, 119)
```

```
: # Export cleaned data
  cleanedData = encodedDF
  cleanedData.to_csv(r"C:\Users\ashle\Desktop\MSDA WGU\Data Mining 2 -D212\task 3\cleaned data\cleanedData.csv", index=False)
```

Association Rules with the Apriori Algorithm

Applying the Apriori algorithm to identify frequently co-prescribed medications. Then, printing the top item sets.

```
5]: # Applying Apriori to the dataset to identify frequently prescribed medications.
    rules = apriori(cleanedData, min_support=0.02, use_colnames=True)
    rules.head()
```

5]:

	support	itemsets
0	0.046794	(Premarin)
1	0.238368	(abilify)
2	0.020397	(albuterol aerosol)
3	0.033329	(allopurinol)
4	0.079323	(alprazolam)

```
7]: rules.shape
```

```
7]: (103, 2)
```

Creating association rules from the frequent item sets with a filter to include only those with a lift of 1 or higher, meaning the items are positively associated. The result is stored in ruleTable and then printing the first few rules for review.

```python
# Creating association rules from the item sets with a filter to include only those with a lift above 1.
ruleTable = association_rules(rules, metric='lift', min_threshold =1)
ruleTable.head()
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	kul
0	(abilify)	(amlodipine)	0.238368	0.071457	0.023597	0.098993	1.385352	1.0	0.006564	1.030562	0.365218	0.082441	0.029655	0
1	(amlodipine)	(abilify)	0.071457	0.238368	0.023597	0.330224	1.385352	1.0	0.006564	1.137144	0.299568	0.082441	0.120604	0
2	(abilify)	(amphetamine salt combo)	0.238368	0.068391	0.024397	0.102349	1.496530	1.0	0.008095	1.037830	0.435627	0.086402	0.036451	0
3	(amphetamine salt combo)	(abilify)	0.068391	0.238368	0.024397	0.356725	1.496530	1.0	0.008095	1.183991	0.356144	0.086402	0.155399	0
4	(abilify)	(amphetamine salt combo xr)	0.238368	0.179709	0.050927	0.213647	1.188845	1.0	0.008090	1.043158	0.208562	0.138707	0.041372	0

Support, Lift, Confidence Values

Displaying the support, confidence, and lift metrics associated with each association rule. Please refer to the "association_rules.csv" file attached for the full table.

```python
# Print rules with their support, lift, and confidence values
ruleTable[['antecedents', 'consequents', 'support', 'lift', 'confidence']]
```

	antecedents	consequents	support	lift	confidence
0	(amlodipine)	(abilify)	0.023597	1.385352	0.330224
1	(abilify)	(amlodipine)	0.023597	1.385352	0.098993
2	(amphetamine salt combo)	(abilify)	0.024397	1.496530	0.356725
3	(abilify)	(amphetamine salt combo)	0.024397	1.496530	0.102349
4	(abilify)	(amphetamine salt combo xr)	0.050927	1.188845	0.213647
...	...	...	...	...	...
89	(metoprolol)	(diazepam)	0.022930	1.468215	0.240559
90	(doxycycline hyclate)	(glyburide)	0.020131	1.239135	0.211781
91	(glyburide)	(doxycycline hyclate)	0.020131	1.239135	0.117785
92	(glyburide)	(losartan)	0.028530	1.263488	0.166927
93	(losartan)	(glyburide)	0.028530	1.263488	0.215943

94 rows × 5 columns

assocation_rules.csv

Top Three Relevant Rules

- Support – The overall frequency of the rule across the dataset. For example, a support value of 0.05 means the combination is present in 5% of all transactions.

```
# Printing the three rules with the highest support value
topThreeRulesSupport = ruleTable.sort_values(by='support', ascending=False).head(3)
topThreeRulesSupport
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	kulc
8	(carvedilol)	(abilify)	0.174110	0.238368	0.059725	0.343032	1.439085	1.0	0.018223	1.159314	0.369437	0.169312	0.137421	0.2
9	(abilify)	(carvedilol)	0.238368	0.174110	0.059725	0.250559	1.439085	1.0	0.018223	1.102008	0.400606	0.169312	0.092566	0.2
19	(abilify)	(diazepam)	0.238368	0.163845	0.052660	0.220917	1.348332	1.0	0.013604	1.073256	0.339197	0.150648	0.068256	0.2

- Confidence – The likelihood that item B appears in transactions that contain item A. For example, a confidence of 0.45 indicates that B occurs 45% of the time when A is present.

```
# Printing the three rules with the highest confidence value
topThreeRulesConfidence = ruleTable.sort_values(by='confidence', ascending=False).head(3)
topThreeRulesConfidence
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	kulc
30	(metformin)	(abilify)	0.050527	0.238368	0.023064	0.456464	1.914955	1.0	0.011020	1.401255	0.503221	0.086760	0.286354	0.2
24	(glipizide)	(abilify)	0.065858	0.238368	0.027596	0.419028	1.757904	1.0	0.011898	1.310962	0.461536	0.099759	0.237201	0.2
28	(lisinopril)	(abilify)	0.098254	0.238368	0.040928	0.416554	1.747522	1.0	0.017507	1.305401	0.474369	0.138413	0.233952	0.2

- Lift – The strength of the association between A and B, compared to what would be expected if they were independent. A lift greater than 1 implies a positive association; the items co-occur more often than chance would predict.

```
# Printing the three rules with the highest lift value
topThreeRulesLift = ruleTable.sort_values(by='lift', ascending=False).head(3)
topThreeRulesLift
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	kulc
75	(carvedilol)	(lisinopril)	0.174110	0.098254	0.039195	0.225115	2.291162	1.0	0.022088	1.163716	0.682343	0.168096	0.140684	0.3
74	(lisinopril)	(carvedilol)	0.098254	0.174110	0.039195	0.398915	2.291162	1.0	0.022088	1.373997	0.624943	0.168096	0.272197	0.3
72	(glipizide)	(carvedilol)	0.065858	0.174110	0.022930	0.348178	1.999758	1.0	0.011464	1.267048	0.535186	0.105651	0.210764	0.2

Results

The screenshot below displays the top ten medication association rules identified from the dataset, each representing a pair of drugs that frequently co-occur in patient prescriptions. Every rule shown has a support value of at least 0.040, meaning these medication combinations appear in at least 4% of all transactions. This indicates they are not rare, isolated pairings but represent moderately frequent prescribing patterns worth further investigation.

Each rule also demonstrates a confidence level of 0.20 or higher, suggesting that when Medication A is prescribed, Medication B is co-prescribed in 20% or more of those cases. While 20% may not indicate a highly deterministic relationship, it does suggest a moderate conditional dependence, especially relevant in settings where treatment variability is common.

Finally, all association rules in this set exhibit a lift value greater than 1.1, implying that these co-prescriptions occur more frequently than random chance would predict. A lift above 1 confirms a positive association, reinforcing the idea that these medications are not just independently common, but meaningfully related in a clinical context.

Overall, the combination of meaningful support, moderate-to-high confidence, and positive lift underscores the relevance of these medication pairings in real-world prescribing behavior. These insights can inform inventory management, highlight common treatment pathways, promote safer prescribing, and ultimately improve patient care outcomes.

```python
# Printing the top 10 most common rules
top10Rules = ruleTable.sort_values(by=['support', 'confidence', 'lift'], ascending=False).head(10)
top10Rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	representativity	leverage	conviction	zhangs_metric	jaccard	certainty	ku
9	(carvedilol)	(abilify)	0.174110	0.238368	0.059725	0.343032	1.439085	1.0	0.018223	1.159314	0.369437	0.169312	0.137421	
8	(abilify)	(carvedilol)	0.238368	0.174110	0.059725	0.250559	1.439085	1.0	0.018223	1.102008	0.400606	0.169312	0.092566	
19	(diazepam)	(abilify)	0.163845	0.238368	0.052660	0.321400	1.348332	1.0	0.013604	1.122357	0.308965	0.150648	0.109018	
18	(abilify)	(diazepam)	0.238368	0.163845	0.052660	0.220917	1.348332	1.0	0.013604	1.073256	0.339197	0.150648	0.068256	
5	(amphetamine salt combo xr)	(abilify)	0.179709	0.238368	0.050927	0.283383	1.188845	1.0	0.008090	1.062815	0.193648	0.138707	0.059103	
4	(abilify)	(amphetamine salt combo xr)	0.238368	0.179709	0.050927	0.213647	1.188845	1.0	0.008090	1.043158	0.208562	0.138707	0.041372	
7	(atorvastatin)	(abilify)	0.129583	0.238368	0.047994	0.370370	1.553774	1.0	0.017105	1.209650	0.409465	0.150000	0.173315	
6	(abilify)	(atorvastatin)	0.238368	0.129583	0.047994	0.201342	1.553774	1.0	0.017105	1.089850	0.467950	0.150000	0.082443	
29	(lisinopril)	(abilify)	0.098254	0.238368	0.040928	0.416554	1.747522	1.0	0.017507	1.305401	0.474369	0.138413	0.233952	
28	(abilify)	(lisinopril)	0.238368	0.098254	0.040928	0.171700	1.747522	1.0	0.017507	1.088672	0.561638	0.138413	0.081449	

Practical Significance of Results

The results of this analysis offer meaningful, real-world insights with direct applications in a medical setting. By identifying the most frequently co-prescribed medications, healthcare administrators and clinical teams can make more data-informed decisions across several operational areas. These include streamlining pharmacy inventory management by ensuring that commonly paired medications are stocked together, evaluating the safety and appropriateness of prescribing trends, and enhancing overall patient care by anticipating therapeutic combinations that align with typical treatment pathways.

Notably, the analysis uncovered medication pairs with a confidence level of 0.45, which means that when one medication is prescribed, there is a 45% likelihood that the other will also be prescribed. In a healthcare environment, where treatment variability is high, this level of confidence is quite strong. It suggests an almost 50-50 chance of co-prescription, representing a reliable and actionable pattern. For hospital systems, this kind of insight is powerful. it supports smarter forecasting, reduces the risk of medication shortages, and empowers care teams to make more consistent, coordinated treatment decisions.

Recommendations

Based on the findings from the analysis, the hospital should consider implementing two strategic initiatives:

First, optimize pharmacy inventory management by aligning stock levels with medication pairs that exhibit high support and confidence levels of being prescribed together. These combinations represent strong co-prescription patterns, and anticipating them can help prevent shortages, reduce waste, and streamline the fulfillment of treatment plans.

Second, establish a review committee, including clinicians, pharmacists, and clinical governance or medical safety personnel, to evaluate the safety and clinical appropriateness of frequently co-prescribed medications. Special attention should be given to pairs with high support values, as their frequent occurrence across the dataset suggests they play a central role in patient care and may benefit from standardization or further safety profiling.

Together, these actions promote safer, more efficient prescribing practices while ensuring that operational decisions are grounded in real-world data.

Sources

Mehta, K. (2024, December 31). *Market basket analysis*. What Is It, Examples, Types, Data Mining. https://www.wallstreetmojo.com/market-basket-analysis/

Deniran, O. H. (2023, November 27). *Boosting sales with data: The Power of Market Basket Analysis in retail*. Medium. https://medium.com/@chemistry8526/boosting-sales-with-data-the-power-of-market-basket-analysis-in-retail-c79cc10a14df