**D212 Data Mining II**

**Task 3: Market Basket Analysis**

Petra I Bier

Western Governors University

D212 Data Mining II

December 16, 2024

Dr. Kesselly Kamara PhD

**D212 Data Mining II**

**Task 3: Market Basket Analysis**

**Part I: Question**

A :

A1: Question

Which medications should be located in the automatic medication dispenser (Pyxis) based on their historical incidence of being prescribed together to improve medication availability and inventory management?

A2: Goal

As medication costs rise for patients and insurance companies, thorough data analysis is needed to discover patterns and trends in prescription and inventory control of medications to reduce costs and prevent waste. Based on the analysis results, workflows could be developed to reduce the time hospital staff need to obtain frequently prescribed medications and reduce the number of expired medicines by decreasing the inventory for specific medications. The goal of this Market Basket Analysis (MBA) is to identify medication combinations that are prescribed together with a support of 0.03 or greater and a lift metric of greater than one to improve Pyxis medication stocking to decrease staff retrieval time and medication expiration rates.

**Part II Justification**

B :

B1: Explanation

Market Basket Analysis is a type of data mining that looks for associations between transaction items in a data set. MBA can be used in multiple industries to reveal customer purchasing habits, detect fraud in the financial sector, and track the most common prescriptions of specific medical diagnoses.

An MBA often uses the Apriori algorithm to uncover relationships between items. Item sets, which consist of items that appear together within transactions, are found first. Item set examples would be transactions that may include things such as Christmas ornaments, tinsel, tree lights, and gingerbread. Any combination of these items in a transaction can be an item set.

*(Ornament, Tinsel)*

*(Lights, Tinsel)*

*(Ornament, Gingerbread, Lights)*

Then, association rules, which show relationships between these items, are discovered. An association rule is when one item, the antecedent, is seen occurring with another item, the consequent in any transaction.

*(Antecedent) 🡪 (Consequent)*

An example of an association rule would be if tinsel (the consequent) is purchased when Christmas ornaments are bought (the antecedent). This rule is considered a logical *statement of if (A), then (C)* (Chaudhary, 2024).

*(Ornament) 🡪 (Tinsel)*

This algorithm finds frequent item sets based on three key metrics predefined by the user: support, confidence, and lift. Frequency is “an item set that occurs at least a certain minimum number of times” (Larouse et al., 2014), which the user determines. These discovered relationships show how often the antecedent and consequent appear together. The association rules will show correlation but not causation of the relationships.

Support looks at how frequently an item or item set occurs. It is calculated as a proportion of how often the item set occurs within all transactions, where N is the total number of transactions. Values closer to one indicate that the item set occurs more frequently. This metric applies only to the item set(s), not the association rules.

*Support(A🡪 C) = Count(A and C) / N*

*Support(Ornament🡪 Tinsel) = Count(Ornament and Tinsel) / Total number of transactions*

Confidence is how frequently C occurs when A occurs. The calculation for confidence is the ratio of how often both items appear together over how often the antecedent item occurs in all transactions (Savyakhosla, 2024). It is important to note that confidence is not symmetrical, meaning that Confidence (Ornament🡪 Tinsel) may be different from Confidence (Tinsel 🡪 Ornament) (Raschka, 2018).

*Confidence(A🡪 C) = Count(A and C) / Count(A)*

*Confidence(Ornament🡪 Tinsel) = Count(Ornament and Tinsel) / Count(Ornament)*

Lift measures how likely C is to occur when A occurs, compared to how often C occurs independently. It determines the strength of the association between the antecedent and the consequent occurring (McColl, 2024). Values greater than one mean an increased likelihood that if A (Ornaments) occurs, C (Tinsel) will occur. Values less than one indicate that the presence of A decreases the probability of C occurring compared to the regular occurrence of C. A value equal to one means the items are entirely independent (Sivek, 2024).

*Lift(A🡪 C) = Support(A and C) / (Support(A) \* Support(C))*

*Lift(Ornament🡪 Tinsel) = Support(Ornament and Tinsel) / (Support(Ornament) \* Support(Tinsel))*

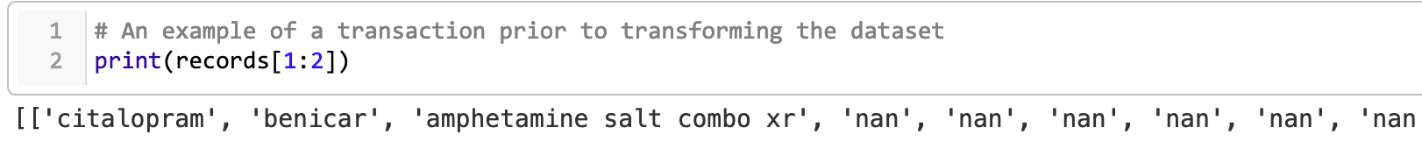
Summary of lift values:

* Values greater than one mean that tinsel is more likely to be purchased if ornaments are purchased.
* Values less than one mean that tinsel is less likely to be purchased when ornaments are purchased.
* Values equal to one mean ornament and tinsel purchases are entirely independent.

This MBA analysis had several goals, including finding item sets of prescribed medication with support of 0.03 or greater and a lift of greater than one with the intent of finding which frequently co-prescribed medications should be stocked in the Pyxis for ease of access.

B2: Transaction

The medical data set had 7501 rows of data after cleaning, with the first row being a list of all the medications prescribed, leaving 7000 transactions to be analyzed. An example of a transaction (prescription) would be (citalopram, benicar, amphetamine salt combo xr). The three medications appear together in a single prescription.



B3: MBA Assumption

For any statistical analysis to work, the assumptions of that model must be met. For an MBA, each entry is assumed to be a single transaction. If this criterion is not met, the analysis will not be a reliable model on which decisions can be based (Deniran, 2023).

**Part III Preparation**

C :

C1: Transformation

The data set for this analysis had to be cleaned and transformed to run an MBA on the transactions. The initial data was found to be 15002 rows and 20 columns. It was also noted that there were a large number of duplicates.

A close-up of a website

Description automatically generated A screenshot of a computer code

Description automatically generated

Investigating the duplicates revealed a large number of rows with only NaN values. These were removed (Kumar, 2018), which gave a final count of 7501 rows and 20 columns.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The next step in preprocessing the data was to transform each transaction into a list of medications prescribed together. These individual lists of prescriptions were combined into a more extensive list containing all the transactions (a "list of lists"). This can be thought of as a folder containing multiple sheets of paper, each with a list of medications for each prescription.

records = []

for i in range (0, 7501):

records.append([str(basket.values[i,j]) for j in range(0, 20)])

A screenshot of a computer

Description automatically generated

The data can then be one hot encoded into True and False values using the mlxtend package (Raschka, 2018), which gives any instance of a value as True, and NaNs as False. An array is created with the value names used for the columns, and each row is a single transaction. This array is then turned into a database to run the Apriori algorithm. This resulted in the database having 7501 rows and 120 columns.

TE = TransactionEncoder()

array = TE.fit(records).transform(records)

transf\_df = pd.DataFrame(array, columns = TE.columns\_)

pd.set\_option('display.max\_columns', None)

transf\_df

A screenshot of a computer

Description automatically generated

After the database was created, the columns were reviewed to check for any empty rows, and a single column titled "nan" was found, which was removed, leaving 7501 rows and 119 columns. The data was cleaned and ready to be analyzed using the Apriori algorithm.

TE.columns\_

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer code

Description automatically generated



analysis = transf\_df.drop(['nan'], axis = 1)

analysis

A screenshot of a computer

Description automatically generated

The cleaned data set is included in the project submission, labeled MBA\_data.csv .

C2: Apriori Code

The Apriori algorithm finds frequent item sets where the user inputs a support threshold value pre-determined to define "frequent". The algorithm functions on the principle that adding another item to the set will not make it more frequent if it is infrequent. The algorithm is iterative, searching for the frequent items first and then for the frequent item sets (Raschka, 2018). Based on this principle, the algorithm does not continue to search any item sets containing infrequent items. This can be simplified by the idea that if no one likes ice cream, adding chocolate chips will not make more people like it.

The Apriori model was fit on the cleaned data set. The total of frequent items and items set were found was 54.

#Defined 0.03 minimum support

a\_rules = apriori(analysis, min\_support=0.03, use\_colnames=True)

a\_rules.sort\_values(by=['support'], ascending=False)

A screenshot of a computer

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A close-up of a person's face

Description automatically generated

A screenshot of a computer code

Description automatically generated

C3: Rules Table

An association rules table was created using the lift metric as the selection criteria. The lift value was set to one. The association table lists the consequent and antecedent support values for each item or item set, the support value for the combined item set, the confidence value of the association rule, and the lift value of the rule. The total number of rules created based on lift was 32.

rule\_table = association\_rules(a\_rules, metric = 'lift', min\_threshold = 1)

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

C4: Top Rules

The association rules were sorted according to lift, and the top three were analyzed.

top\_rules = rule\_table.sort\_values("lift", ascending = False)

top\_rules.head(3)

A screenshot of a computer

Description automatically generated

Rule 1: (Carvedilol) 🡪 (Lisinopril)

* Antecedent support (0.174): 17.4% of all prescriptions contain Carvedilol
* Consequent support (0.098): 9.8% of all prescriptions contain Lisinopril
* Support (0.039): 3.9% of prescriptions contain both Carvedilol and Lisinopril
* Confidence (0.225): If Carvedilol is prescribed, Lisinopril is prescribed 22.5% of the time
* Lift (2.291): The two medications appear together 2.29 times more than if prescribed individually

Rule 2: (Lisinopril) 🡪 (Carvedilol)

* Antecedent support (0.098): 9.8% of all prescriptions contain Lisinopril
* Consequent support (0.174): 17.4% of all prescriptions contain Carvedilol
* Support (0.039): 3.9% of prescriptions contain both medications
* Confidence (0.399): If Lisinopril is prescribed, Carvedilol is prescribed 39.9% of the time
* Lift (2.291): The two medications appear together 2.29 times more than if prescribed individually

Rule 3: (Abilify) → (Lisinopril)

* Antecedent support (0.238): 23.8% of prescriptions contain Abilify
* Consequent support (0.098): 9.8% of prescriptions contain Lisinopril
* Support (0.041): 4.1% of prescriptions contain both medications
* Confidence (0.172): When Abilify is prescribed, Lisinopril is prescribed 17.2% of the time
* Lift (1.748): The two medications appear together 1.75 times more than if prescribed individually

**Part IV Analysis**

D

D1: Significance of Evaluation Metrics

This analysis is based on the lift metric value being greater than one. The metrics used for this analysis reveal several points regarding the positive associations between the items. The support metric between the three medications that appear in the top three rules ranges from 9.8% to 23.8%, showing how often each medication occurs in all the prescriptions evaluated. Abilify is prescribed the most, followed by Carvedilol and Lisinopril.

The confidence values show the strength of the relationship between the item sets. It can be noted that the relationships between the same items are not symmetrical. The second rule of Lisinopril and Carvedilol has a higher value (39.9%) than the inverse, rule one, Carvedilol and Lisinopril (22.5%). Rule three also has a high confidence value of 17.2%. These rules show how much more likely the second medication is to be prescribed in the presence of the first medication.

The lift metric has the highest values for rules one and two at 2.29 each. This shows a strong relationship between Carvedilol and Lisinopril and a moderate strength in the relationship between Abilify and Lisinopril (1.75). These lift values imply that the medication item sets appear together at a higher rate than just random chance.

D2: Practical Significance

The results from this analysis can improve inventory control and accessibility of medications through stocking decisions based on the data. By understanding which medications are commonly prescribed together, Pyxis can be stocked for efficiency, leading to improved workflow for medical staff and nurses.

D3: Recommendations

Several actions can be recommended based on this analysis of the data. The initial recommendation would be to ensure the Pyxis machine is stocked with commonly prescribed medications and the less prescribed medicines, which have a high incidence of being prescribed in conjunction with other medicines. In this case, specifically Carvedilol and Lisinopril. Since Abilify is already the most commonly prescribed medication, it would be sensible to extend the analysis of the rules to see which medications are most often co-prescribed with Abilify.

Future analysis recommended would be to look at the medications that have both low support and low lift, as this may reveal rarely prescribed medications. Ordering these medications from the pharmacy would have a low impact on staff workflow due to the infrequent prescription rate. These medications may also be at higher risk of expiring and should be kept in the pharmacy in smaller quantities to prevent waste due to expiration.

**Part V**

E : Panopto

<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=22df12cc-4c13-4591-91e4-b2480177af72>

F: Web sources

Chaudhary, S. (2024). *Market Basket Analysis: Anticipating Customer Behavior*. Retrieved December 5, 2024, from <https://www.turing.com/kb/market-basket-analysis>

Deniran, O. H. (2023, November 27). *Boosting Sales with Data: The Power of Market Basket Analysis in Retail*. Retrieved November 10, 2024, from <https://medium.com/@chemistry8526/boosting-sales-with-data-the-power-of-market-basket-analysis-in-retail-c79cc10a14df>

Kumar, D. (2018, December 25). Introduction to Data Preprocessing in Machine Learning. Towards Data Science. Retrieved November 15, 2024, from <https://futuremachinelearning.org/understanding-silhouette-score-a-key-metric-for-clustering/>

McColl, L. (2024). *Market Basket Analysis: Understanding Customer Behaviour*. Select Statistical Services. Retrieved November 10, 2024, from <https://select-statistics.co.uk/blog/market-basket-analysis-understanding-customer-behaviour/>

Raschka, (2018). MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. Journal of Open Source Software, 3(24), 638, https://doi.org/10.21105/joss.00638

Savyakhosla (2024, August 16). *Market Basket Analysis in Data Mining*. Geeks for Geeks. Retrieved December 5, 2024, from <https://www.geeksforgeeks.org/market-basket-analysis-in-data-mining/>

Sivek, S. C., Ph.D (2020, November 16). *Market Basket Analysis 101: Key Concepts*. Towards Data Science. Retrieved December 5, 2024, from <https://towardsdatascience.com/market-basket-analysis-101-key-concepts-1ddc6876cd00>

G: Sources

Larose, Daniel T., and Chantal D. Larose. *Discovering Knowledge in Data : An Introduction to Data Mining*, John Wiley & Sons, Incorporated, 2014.

References

Chaudhary, S. (2024). *Market Basket Analysis: Anticipating Customer Behavior*. Retrieved December 5, 2024, from <https://www.turing.com/kb/market-basket-analysis>

Deniran, O. H. (2023, November 27). *Boosting Sales with Data: The Power of Market Basket Analysis in Retail*. Retrieved November 10, 2024, from <https://medium.com/@chemistry8526/boosting-sales-with-data-the-power-of-market-basket-analysis-in-retail-c79cc10a14df>

Kumar, D. (2018, December 25). *Introduction to Data Preprocessing in Machine Learning*. Towards Data Science. Retrieved April 15, 2024, from <https://towardsdatascience.com/introduction-to-data-preprocessing-in-machine-learning-a9fa83a5dc9d>

McColl, L. (2024). *Market Basket Analysis: Understanding Customer Behaviour*. Select Statistical Services. Retrieved November 10, 2024, from <https://select-statistics.co.uk/blog/market-basket-analysis-understanding-customer-behaviour/>

Raschka, (2018). MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. Journal of Open Source Software, 3(24), 638, https://doi.org/10.21105/joss.00638

Savyakhosla (2024, August 16). *Market Basket Analysis in Data Mining*. Geeks for Geeks. Retrieved December 5, 2024, from <https://www.geeksforgeeks.org/market-basket-analysis-in-data-mining/>

Sivek, S. C., Ph.D (2020, November 16). *Market Basket Analysis 101: Key Concepts*. Towards Data Science. Retrieved December 5, 2024, from https://towardsdatascience.com/market-basket-analysis-101-key-concepts-1ddc6876cd00