Association Rules Analysis to Identify SKUs to Relocate at Dillard's in Biloxi, MS

Northwestern University IEMS 308
Jake Atlas

Executive Summary

In determining where to position a particular stock keeping unit (SKU), Dillard's should consider the value that can be added by strategic positioning of items. By analyzing point of sales (POS) data, Dillard's can obtain sets of items that are associated with the purchase of other items. Subsequently, by adjusting the planogram of a store such that these associated items are located near each other, sales can be driven upward, resulting in increased revenues and profits.

Association rules analysis is an analytical technique that serves to identify relationships in data. Often called market basket analysis, association rules examination is particularly useful for stores, which can adjust item locations to drive sales of other goods that are determined to be purchased commonly with the relocated item. Using the POS data from Dillard's stores, which contains information regarding each transaction for a given time period, Dillard's management can take advantage of association rules techniques to identify and exploit patterns that will drive in-store sales.

After performing exploratory analysis on the data and determining which attributes of the data would be most relevant in establishing associative relationships between purchased items, a subset of the dataset was taken such that all transactions came from the POS system at the Biloxi, MS Dillard's store. Association rules analysis was then conducted to identify patterns.

The results proved to be favorable, in that there are clear associative relationships between items. Dillard's can use these relationships to change the planogram of the Biloxi store, rearranging such that associated items are nearer to each other. By positioning these associated items close, Dillard's can drive sales of goods that are associated with the moved items.

Problem Statement

The Dillard's point of sales dataset contains a wealth of information about transactions conducted onsite at Dillard's store locations across the United States. My job was to identify a set of 100 SKUs that are good candidates for relocation, as Dillard's wishes to modify the planograms in its stores.

Methodology

The first step in conducting an association rules analysis of the Dillard's POS data is performing exploratory data analysis to determine ways in which to subset the data without reducing the value of the subsequent analysis. Due to the large size of the dataset and the limited computing power of available computers, it was necessary to examine only one store's transaction history. Biloxi, MS was chosen for this analysis, and a table containing the relevant information for Biloxi transactions was added to the database containing the original data.

In order to prepare for association rules analysis, three data preprocessing techniques were applied to the data. First, duplicate records were removed; if, in a particular transaction, an SKU was purchased multiple times, the duplicates were removed so that the purchase of this SKU was represented only once for the given transaction. Subsequently, SKUs that were represented under 5 times across all transactions were eliminated, as they did not appear enough to warrant moving in the store. Finally, the remaining SKUs were one-hot encoded. At this point, all transactions were listed in a single row, with binary indicators to denote whether each of the SKUs was purchased in that particular transaction.

Association Rule Examples

Antecedent		Consequent
$\{A\}$	$\rightarrow$	$\{C\}$
$\{A, B\}$	$\rightarrow$	$\{C, D, E\}$

Figure 1

Association rules analysis was conducted next. Above, labeled Figure 1, are examples of generic association rules which complement the discussion of the methodology applied. In order to determine sets of association rules, the Apriori algorithm was used. The Apriori algorithm first examines the relative frequency of each item's appearance, referred to as the "support" of each item. If support does not meet a minimum threshold, all subsequent baskets containing that SKU are eliminated from the analysis. The algorithm then moves to considering baskets of two items at a time, and applies the same logic. This is repeated for increasing basket size until all possible baskets of SKUs are either considered or eliminated.

To output a list of candidate SKUs for moving, the algorithm was repeated using varying minimum support values until as close as possible to 100 unique SKUs could be produced as antecedents. Appearing at the top of the next page, Figure 2 details the total number of unique SKUs appearing as antecedents across the outputted rules as a function of the minimum support level. As the threshold is lowered, more rules are generated and more unique SKUs appear, but the time to run the algorithm is significantly increased as minimum support decreases. Since the minimum support level necessary to output 100 values is somewhat low and also because the runtime requires many days, I've selected to compile the list of recommended SKUs to relocate using two methods. 85% of the SKUs come from association rules analysis. The remaining 15% are simply items that are purchased very frequently, which may be items that Dillard's wishes to place at the forefront of the store. For the entire list of candidate SKUs, see Appendix I on page 7.

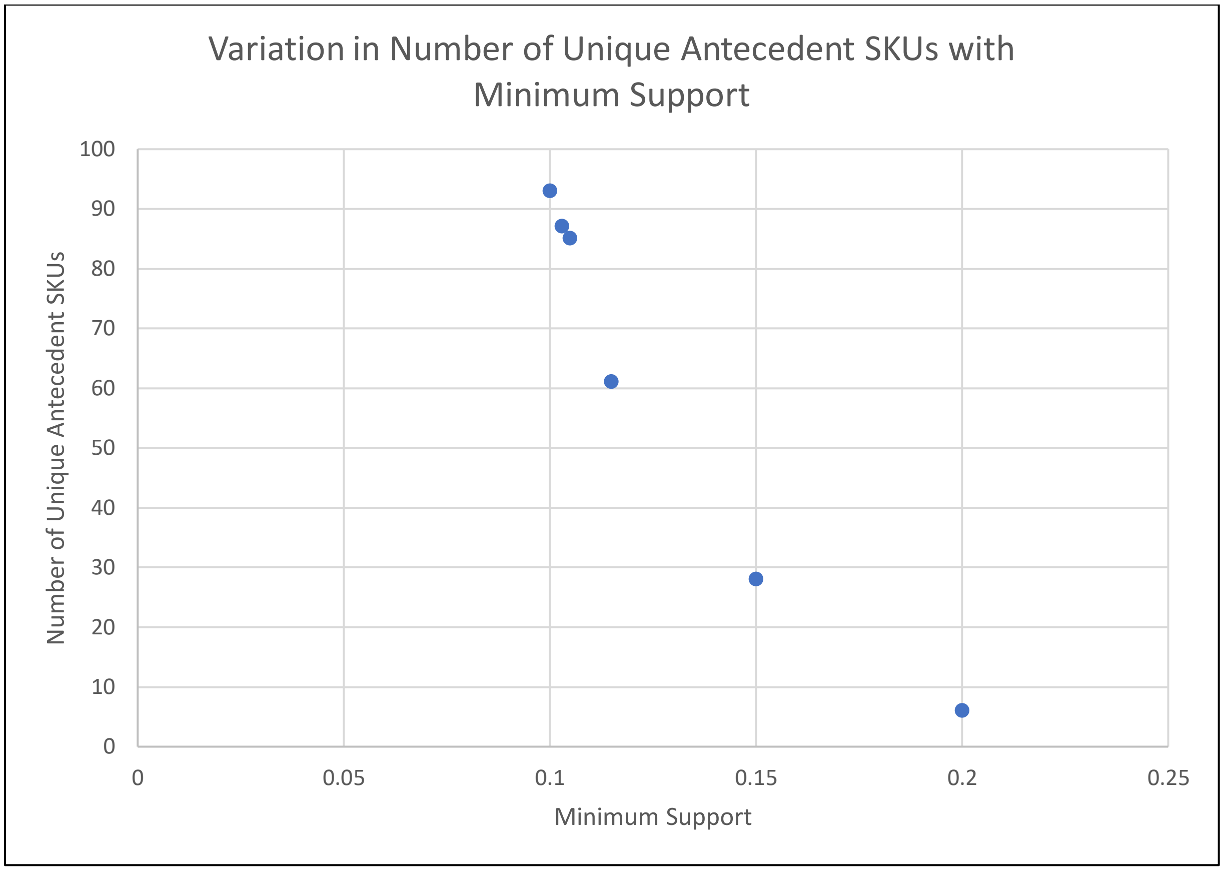

Figure 2

Analysis

The entire analysis is focused on identifying strong patterns in the data. In order to distinguish given association rules from each other, three primary measures were used. These are support, confidence, and lift.

Support is simply the probability that a basket contains all the SKUs in both the antecedent and consequent. To ensure that support was sufficiently high for all rules, three measures were taken. First, all SKUs that appeared fewer than 5 times were removed. If a SKU appeared infrequently, then support would be low, so these were removed before the Apriori algorithm was applied. Second, during the application of the Apriori algorithm, a minimum support value was used, which filters out potential association rules regarding set of items that appear infrequently. Lastly, after applying the Apriori algorithm, a third filter was imposed on the data that further pared down the rules based on support, except in the cases where confidence or lift were particularly high.

Confidence is the probability that a basket contains the consequent given that it contains the antecedent. Like support, an association rule is better if it has high confidence. In order to ensure the association rules chosen had high confidence, a filter was imposed on data that eliminated rules that didn't meet a baseline confidence, except in the cases where support or lift were particularly high.

Lift is the confidence of a rule divided by the probability that the consequent is in a transaction. As with support and confidence, high lift signifies a potentially meaningful association rule. In order to be sure that the selected association rules had high lift values, rules were removed if lift values were low, except in the cases where support and confidence were particularly high.

This entire process was repeated at varying values of minimum support during the actual application of the Apriori algorithm. Due to the numerous checks on support, confidence, and lift, the suggested SKUs obtained using association rules analysis are all part of meaningful association rules. Since it was not feasible to obtain 100 unique SKUs using this method alone, the list of candidate SKUs was supplemented by SKUs that appeared quite frequently across transactions. It makes sense to provide the remaining SKUs by examining the frequency with which each SKU is purchased; items purchased very frequently have merit for relocation to a focal point in the store. It is therefore logical to add these SKUs to the list of candidates instead of further decreasing the minimum support threshold to allow SKUs with weaker association rules in the list.

Conclusions

The key insights from the association rules analysis applied to the Dillard's point of sales data are:

Dillard's should absolutely consider reevaluating its planograms and moving SKUs. Many of the association rules generated had very confidence and lift values, which indicates that moving some items to be nearer to each other will almost definitely drive sales of associated items. Dillard's should take advantage of these natural patterns.

Dillard's' goal of moving no more than 20 items in a store is not particularly restrictive. The association rules generated involve many of the same SKUs, repeated in different patterns. Due to the nature of association rules and the Apriori algorithm, this makes sense. A large number of the rules with shorter antecedents and consequents are just subsets of rules with larger antecedents and consequents. Due to the fact that it requires a low minimum support value to generate rules with 100 unique antecedent SKUs, it is likely the case that making only 20 moves will capture a significant amount of the untapped value in the layout of the planograms.

These key insights are validated both by the association rules generated as well as by the broader analysis conducted to identify patterns within the rules. I am therefore confident in my conclusion that Dillard's should take advantage of its ability to rearrange its planograms by making 20 SKU relocations.

Next Steps

While the Apriori algorithm is very effective at identifying associations in baskets, it fails to take monetary implications into account. Although it may be the case that {A}$\rightarrow${B} has higher support, confidence and lift than {C}$\rightarrow${D}, if the profit margin associated with selling D is significantly higher than the profit margin associated with selling B, it may be more effective to focus on {C}$\rightarrow${D} to drive sales of D, despite the fact that the Apriori algorithm and related metrics label it as a weaker rule. In order to remedy this, support values can be assigned to individual items, which can boost or lower the "value" of a rule. Incorporating this technique into the analysis would be a very useful next step for Dillard's.

It is also important to consider other stores besides Biloxi, MS. It is not necessarily the case that the suggested planogram changes in Biloxi would be identical to the suggested changes in other stores, as different stores carry different items and see different customers. Therefore, it is more effective to carry out this broader analysis on a store-by-store basis, essentially repeating this analysis, which regards only one of Dillard's' many locations. Exploring further on a store by store basis would enable Dillard's to make pinpointed changes to the planograms in accordance with the items in stock and the customers' tendencies.

Appendix 1 – Candidate SKUs

29633	3524026	4306623	9257426
39633	3559555	4498011	9277426
173088	3589483	4628597	9288109
250896	3631365	4668011	9297426
264715	3690654	4686413	9308109
270896	3864099	4976322	9370294
726718	3868338	4992993	9402188
776350	3871688	4999530	9526376
803921	3874099	5079809	9549158
1184024	3894099	5108107	9552306
1310252	3898011	5509179	9667426
1588107	3908011	5528349	9708505
1832285	3949538	5957568	7029904
2072671	3968011	5978084	6949904
2258366	3968356	6318344	7039904
2288366	3978011	6367618	5309905
2571221	3988011	6618353	6939904
2688353	3998011	6656135	5749904
2698353	4062567	7808101	5189905
2708353	4072567	8391343	6349904
2716578	4108011	8401343	7049904
2726578	4112626	8718362	3348362
2783996	4138348	8791356	5129905
2938210	4208011	8798636	8067216
3161221	4218011	9230294	5739904

Appendix 2 – Source Code

```
#%% OBTAIN DATA FROM TRANSACTION TABLE & WRITE NEW TABLE IN jaa977_schema
# ******* Note: This section of the code may take over an hour to run and requires Northwestern VPN connection
********

import psycopg2
import pandas as pd
from sqlalchemy import create_engine

#Connect to database
connection = psycopg2.connect(host = "gallery.iems.northwestern.edu", user = 'jaa977', password = 'jaa977_pw',
dbname = 'iems308')

#Establish cursor
cur = connection.cursor()

#Obtain dataframe of Biloxi, MS transactions
cur.execute("SELECT * FROM pos.trnsact WHERE c2 = '4902'")
df_biloxi_transact = pd.DataFrame(cur.fetchall())

df_biloxi_transact.columns =
['sku','store','register','trannum','seq','saledate','stype','quantity','amt','orgprice','ignore_orgprice','interid','mic','ig
nore']

#Remove all columns except sku, trannum, stype
df_biloxi = df_biloxi_transact[['sku','trannum','stype']].copy()

#Obtain subset of Biloxi datafraem that contains only purchases
df_biloxi_purchases = df_biloxi[df_biloxi['stype'].str.contains('P')]

#Remove stype column now that all transactions are purchases
del(df_biloxi_purchases['stype'])

#Strip unnecessary spaces and turn data to integers
df_biloxi_purchases['sku'] = df_biloxi_purchases['sku'].str.strip()
df_biloxi_purchases['trannum'] = df_biloxi_purchases['trannum'].str.strip()


df_biloxi_purchases['sku'] = df_biloxi_purchases['sku'].astype(int)
df_biloxi_purchases['trannum'] = df_biloxi_purchases['trannum'].astype(int)


# Create new table in database, biloxi_purchases, and insert data into the new table
cur.execute("CREATE TABLE jaa977_schema.biloxi_purchases (sku integer PRIMARY KEY, trannum integer);")
engine=create_engine('postgresql+psycopg2://jaa977:jaa977_pw@gallery.iems.northwestern.edu:5432/iems308')
df_biloxi_purchases.to_sql('biloxi_purchases',engine,schema = 'jaa977_schema')
connection.commit()

#Close the connection
cur.close()
connection.close()
#%% FOR USE AFTER NEW TABLE CREATED ONLY - BEGIN ASSOCIATION RULE ANALYSIS

import numpy as np
import pandas as pd
import psycopg2
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

#Connect to database
connection = psycopg2.connect(host = "gallery.iems.northwestern.edu", user = 'jaa977', password = 'jaa977_pw',
dbname = 'iems308')

#Establish cursor
cur = connection.cursor()

#Obtain dataframe from biloxi_transactions
cur.execute("SELECT sku, trannum FROM jaa977_schema.biloxi_purchases")
df_biloxi_purchases = pd.DataFrame(cur.fetchall())
df_biloxi_purchases.columns = ['sku','trannum']

#Close the connection
cur.close()
connection.close()

#Add column to df_biloxi_purchases that specifies number of times SKU appears
sku_counts_series = df_biloxi_purchases['sku'].value_counts()
df_sku_counts = sku_counts_series.to_frame()
df_sku_counts = df_sku_counts.reset_index()
df_sku_counts.columns = ['sku','count']
df_biloxi_with_counts = pd.merge(df_biloxi_purchases,df_sku_counts, on=['sku'])

#Remove duplicates where both sku and trannum match other records
df_biloxi_removed_duplicates = df_biloxi_with_counts.drop_duplicates()

#Create subset of df_biloxi_removed_duplicates where all counts are 5 or higher
df_biloxi_before_encoding = df_biloxi_removed_duplicates[df_biloxi_removed_duplicates['count'] > 4]

#One hot encode the transactions so that each row is a unique transaction and each column denotes
#the presence of an SKU (1) or the lack of an SKU (0) for the given transaction
df_biloxi_encoded = pd.get_dummies(df_biloxi_before_encoding,columns=['sku'])
del(df_biloxi_encoded['count'])
df_biloxi_reencoded = df_biloxi_encoded.groupby('trannum').sum()

#Run the apriori algorithm
frequent_itemsets = apriori(df_biloxi_reencoded,min_support=0.103, use_colnames=True)
rules = association_rules(frequent_itemsets,metric="lift")

#Eliminate rules with antecedents having over 2 items
good_rules = rules[rules['antecedants'].map(len)<=2]



#Obtain a subset of the association rules where support, confidence, and lift all meet minimum threshold values
#or if any one of support, confidence, or lift meet a higher threshold value
good_rules2 = good_rules[((good_rules['support']>.25) & (good_rules['confidence']>.5) & (good_rules['lift']>3)) | \
      ((good_rules['support']>.6) | (good_rules['confidence']>.75) | (good_rules['lift']>4))]

good_rules2 = good_rules2.reset_index()

#Determine the total number of unique antecedent SKUs in good_rules2
antecedents = []
for i in range(0,np.shape(good_rules2)[0]):
   antecedents.append(list(good_rules2['antecedants'].iloc[i])[0])
   if len(good_rules2['antecedants'].iloc[i])==2:
      antecedents.append(list(good_rules2['antecedants'].iloc[i])[1])
antecedents = np.asarray(antecedents)

#Add most frequently bought SKUs to the list (provided they aren't already in the list)
frequent_counts =
df_biloxi_before_encoding.sort_values(by='count',ascending=False).drop_duplicates(subset='sku')

antecedent_integers = []
for num in antecedents:
   antecedent_integers.append(int(num[4:]))

counter = 0
total_skus = np.unique(antecedent_integers)
while len(total_skus)<100:
   if frequent_counts['sku'].iloc[counter] not in total_skus:
      total_skus = np.append(total_skus,frequent_counts['sku'].iloc[counter])
   counter = counter + 1

#Output final SKUs to move
print(total_skus)
```