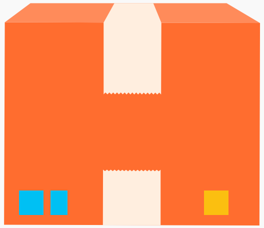
CART-ALYTICS: Understanding Customer Purchase Patterns with Outsta-cart

Group H: Ali Khan, Alicia Wilson, Luis Villazon, Morgan Tucker, Vi Tran

Project Goals

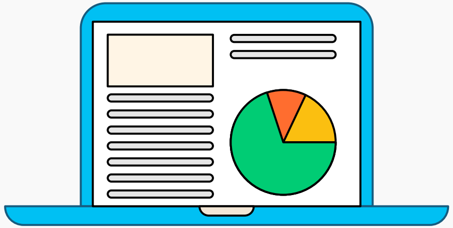
1



Retail

Using market basket analysis
to adjust store layouts

2



Online

Using clustering to make
consumer-specific recommendations

Approach

1 Data Formatting
Created item-specific dummy variables. Reorganized customer purchases

2 EDA
Studying patterns and features

3 Market Basket Analysis
Determined the rules based on Lift calculations

4 Store Layout
Use Market Basket Analysis to suggest store layout

5 Clustering
Clustered customers based on purchase history

6 Recommendations
Consumer-specific grocery selections using clustering

Data Formatting

	Member_number	Date	itemDescription
1629	1000	27-05-2015	soda
13331	1000	24-06-2014	whole milk
8395	1000	15-03-2015	whole milk
4843	1000	15-03-2015	sausage
17778	1000	27-05-2015	pickled vegetables
...	...	...	...
34885	5000	10-02-2015	semi-finished bread
25489	5000	16-11-2014	other vegetables
9340	5000	16-11-2014	bottled beer
27877	5000	09-03-2014	onions
3578	5000	10-02-2015	soda

	Member_number	Date	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	bags	baking powder	bathroom cleaner	...	turkey	vinegar	waffles	whipped/sour cream	whisky
4843	1000	15-03-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
8395	1000	15-03-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
20992	1000	15-03-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
24544	1000	15-03-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
13331	1000	24-06-2014	0	0	0	0	0	0	0	0	...	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3578	5000	10-02-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
19727	5000	10-02-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
34885	5000	10-02-2015	0	0	0	0	0	0	0	0	...	0	0	0	0	0
9340	5000	16-11-2014	0	0	0	0	0	0	0	0	...	0	0	0	0	0
25489	5000	16-11-2014	0	0	0	0	0	0	0	0	...	0	0	0	0	0

One-hot encoding

Exploratory Data Analysis

3898

Customers

There were almost 4000 customers that shopped at the grocery store.

167

Products

There were over 150+ unique products provided by the grocery store

Frequently bought Items

Least Bought Items

Market Basket Analysis

Support IFP

Total number of baskets with IFP

Total number of baskets

Support Bags

Total number of baskets with Bags

Total number of baskets

	Member_number	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	bags	baking powder	bathroom cleaner
0	1000	0	0	0	0	0	0	0	0
1	1001	0	0	0	0	0	0	0	0
2	1002	0	0	0	0	0	0	0	0
3	1003	0	0	0	0	0	0	0	0
4	1004	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
3893	4996	0	0	0	0	0	0	0	0
3894	4997	0	0	0	0	0	0	0	0
3895	4998	0	0	0	0	0	0	0	0
3896	4999	0	0	0	0	0	0	0	0
3897	5000	0	0	0	0	0	0	0	0

Confidence (IFP | Bags)

Total number of baskets with IFP & Bags

Total number of Bags

Lift (IFP | Bags)

Confidence (IFP | Bags)

Support IFP

Market Basket Analysis: Code Chunk

```python
from mlxtend.frequent_patterns import apriori, association_rules

# Assuming you've already prepared the 'df_grouped' DataFrame with one-hot encoding
# If not, include your code here to create 'df_grouped' as you mentioned.

# Generate frequent item sets
frequent_item_sets = apriori(df_grouped.iloc[:,2:169], min_support=0.001, use_colnames=True)

# Generate association rules
rules = association_rules(frequent_item_sets, metric="lift", min_threshold=1.0)

# Create a DataFrame for the rules
pd.options.display.float_format = '{:,.6f}'.format
final_df = pd.DataFrame(columns=['Left Hand Side', 'Right Hand Side', 'Support(%)', 'Confidence(%)', 'Lift'])

# Process the rules
for _, row in rules.iterrows():
    LHS = list(row['antecedents'])
    RHS = list(row['consequents'])
    SUPPORT = row['support'] * 100
    CONFIDENCE = row['confidence'] * 100
    LIFT = row['lift']

    # Convert lists to strings and concatenate them
    LHS_str = ', '.join(LHS)
    RHS_str = ', '.join(RHS)

    new_row = {'Left Hand Side': LHS_str, 'Right Hand Side': RHS_str, 'Support(%)': SUPPORT, 'Confidence(%)': CONFIDENCE, 'Lift': LIFT}
    final_df = final_df.append(new_row, ignore_index=True)

final_df['Rules'] = final_df['Left Hand Side'] + ' -> ' + final_df['Right Hand Side']
print('Number of Rules:', final_df['Rules'].count(), 'Rules')
final_df.head()
```

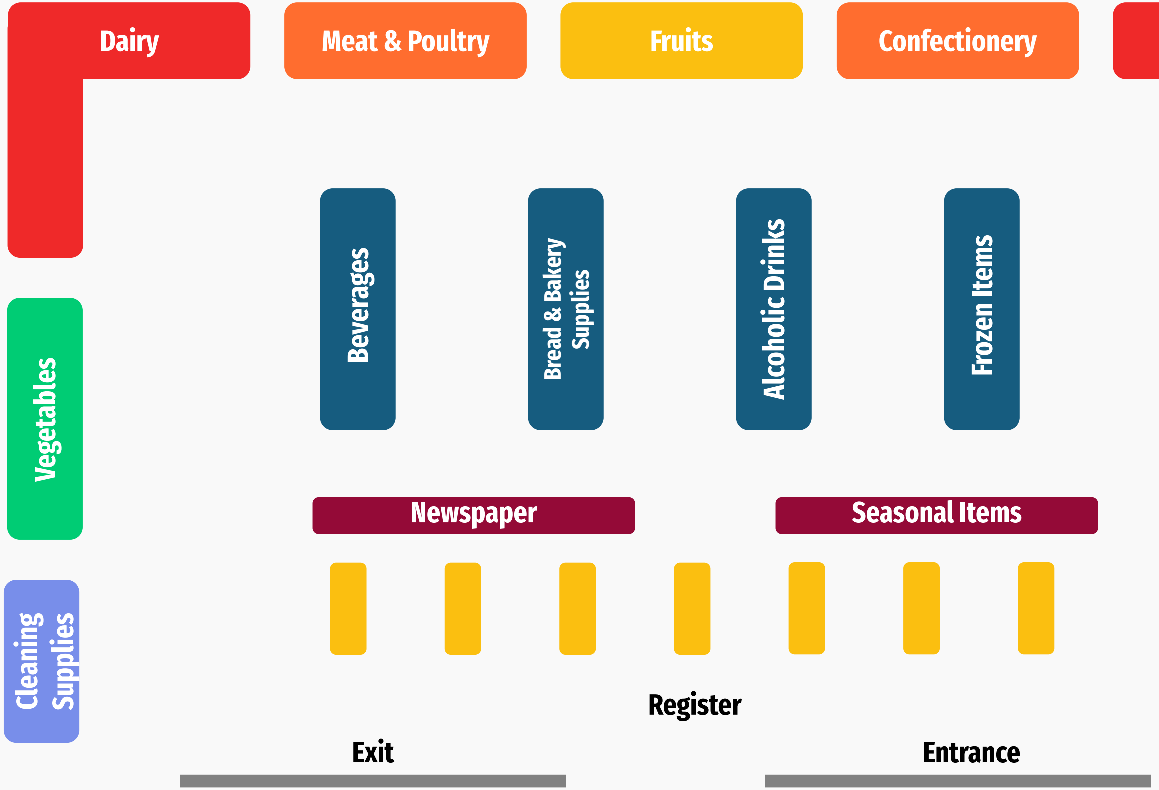
Market Basket Analysis: Results

Rules	Lift	Support	Confidence
Sausage -> Yogurt/Whole Milk	2.18	0.147%	2.44%
Citrus Fruit -> Special Chocolate	1.65	0.140%	2.64%
Tropical Fruit -> Flour	1.62	0.107%	1.58%
Beverages -> Sausages	1.54	0.153%	9.27%
Pastries -> Napkins	1.52	0.174%	3.36%

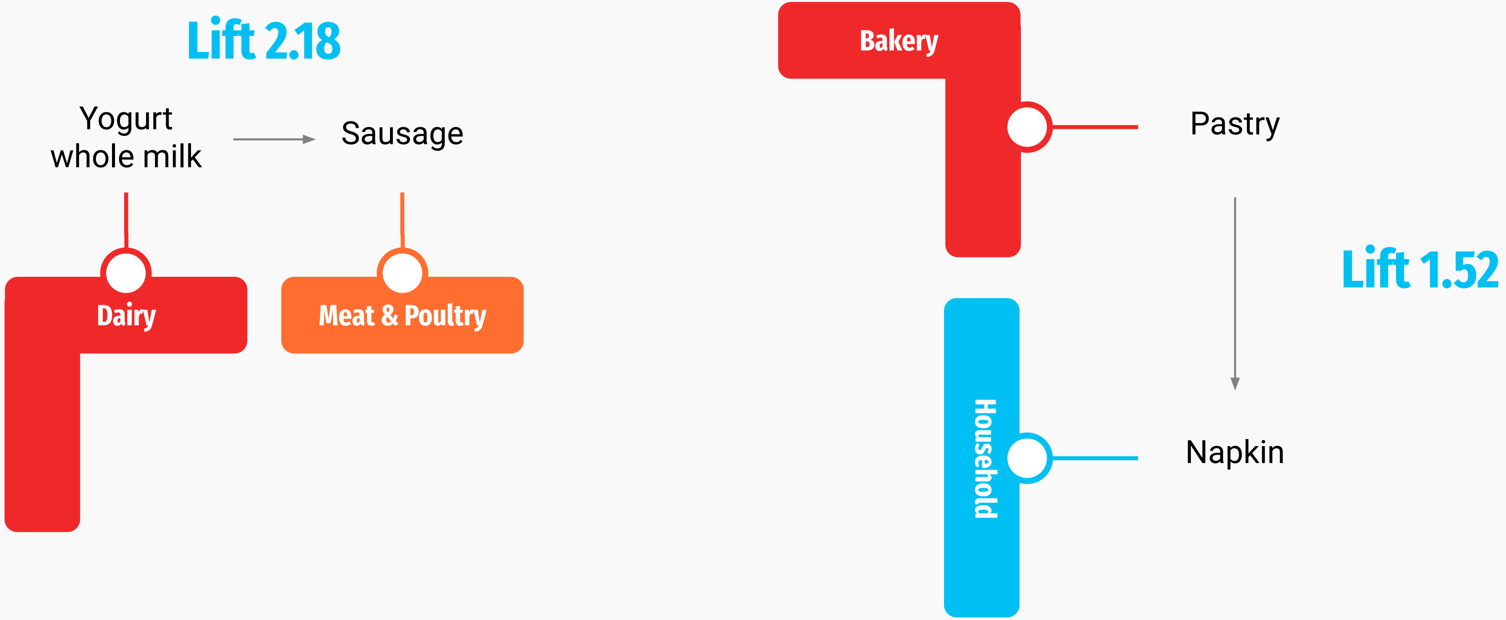
Top Lift Rules

Sausage	2.18	Yogurt/Whole Milk	People who buy sausages are **twice as likely** to buy yogurt or whole milk!
Citrus Fruit	1.65	Specialty Chocolate	People who buy citrus fruits are **1.65 times more likely** to buy speciality chocolate!
Tropical Fruit	1.62	Flour	People who buy tropical fruits are **1.62 times more likely** to buy flour!
Beverages	1.54	Sausages	People who buy beverages are **1.54 times more likely** to buy sausages!
Pastry	1.52	Napkins	People who buy pastry are **1.52 times more likely** to buy napkins!

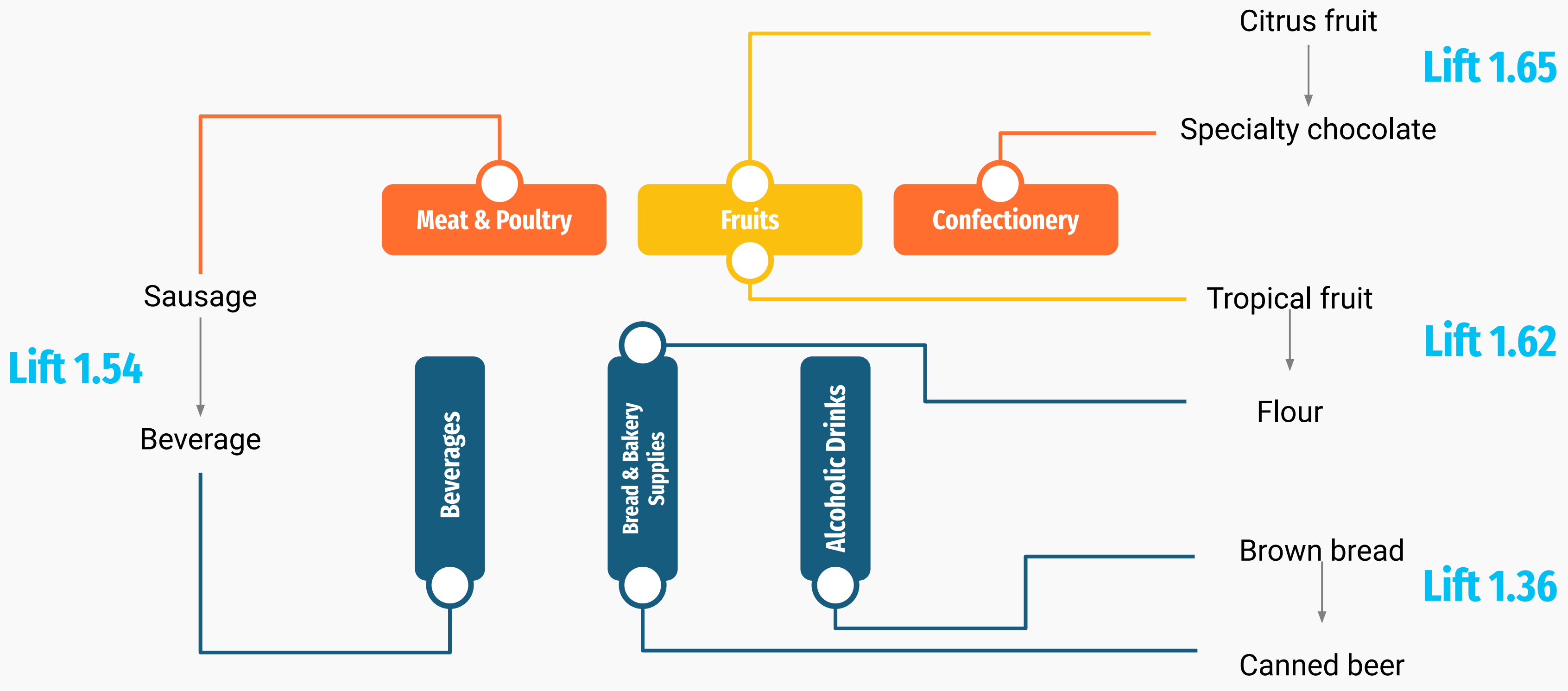
Suggested Store Layout Using Lift Values



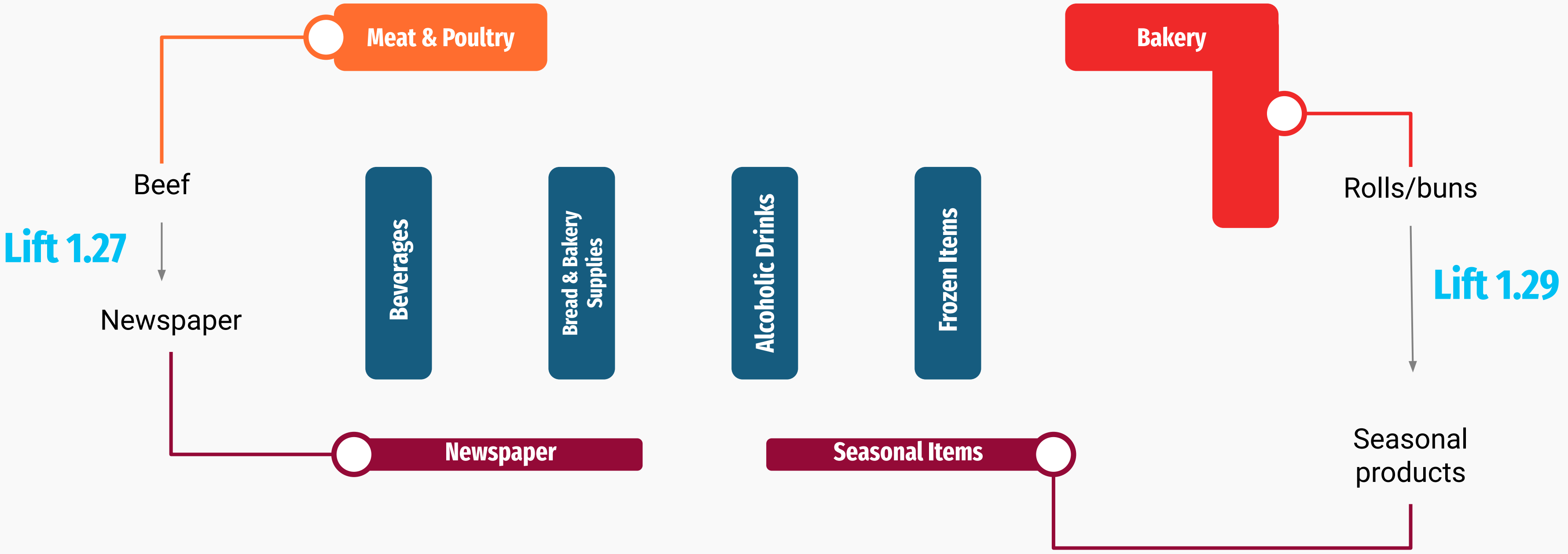
Zooming into the Store Layout



Zooming into the Store Layout

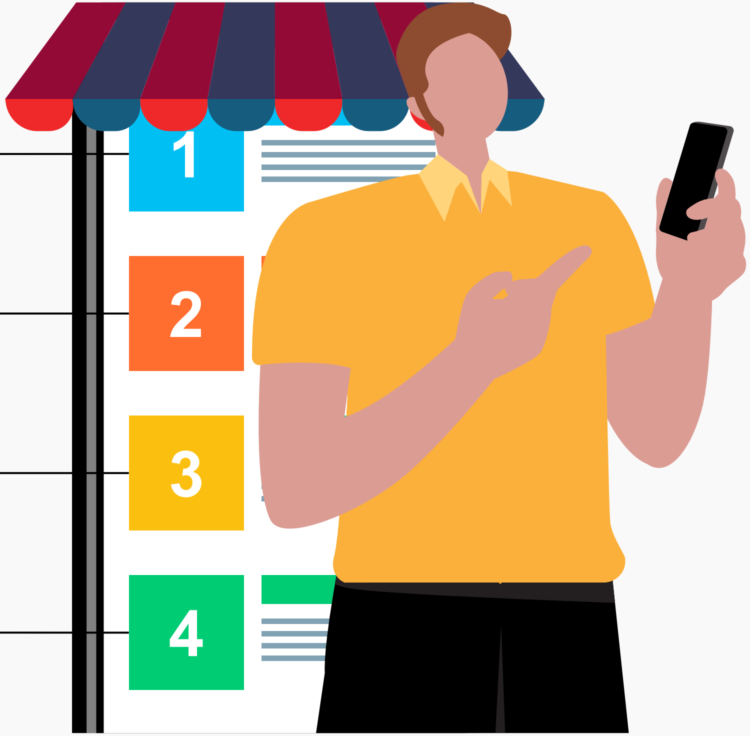


Zooming into the Store Layout



Part II: Clustering (Consumer Specific Recommendations)

Create Dictionary (E.g. Meats and Seafood': ['beef', 'frankfurter', 'chicken', 'hamburger meat'...])	**Categorization**
Cross validate with an Elbow Plot (k = 5) & run clustering	**Finding *k***
Calculate the mean count of each ‘Key’ category for 5 clusters	**Cluster Profiling**
Visualize clusters using PCA & analyze loading scores	**PCA Plotting & Interpretation**

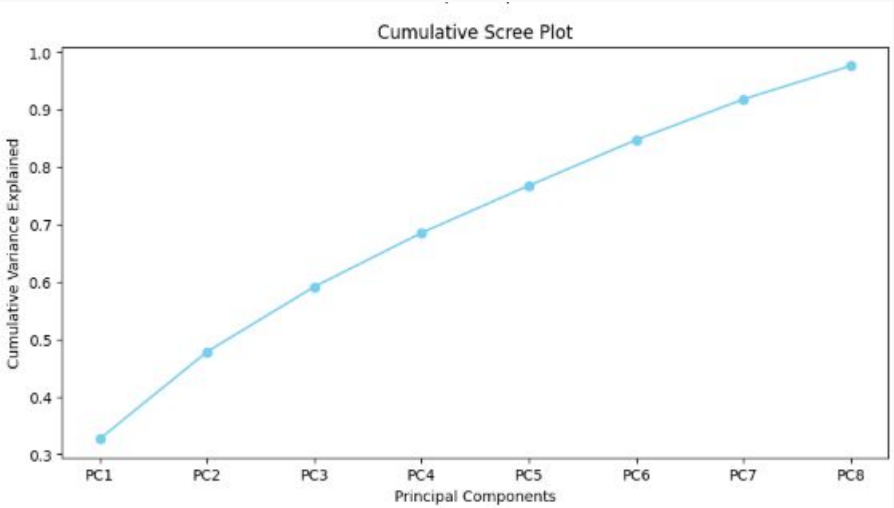


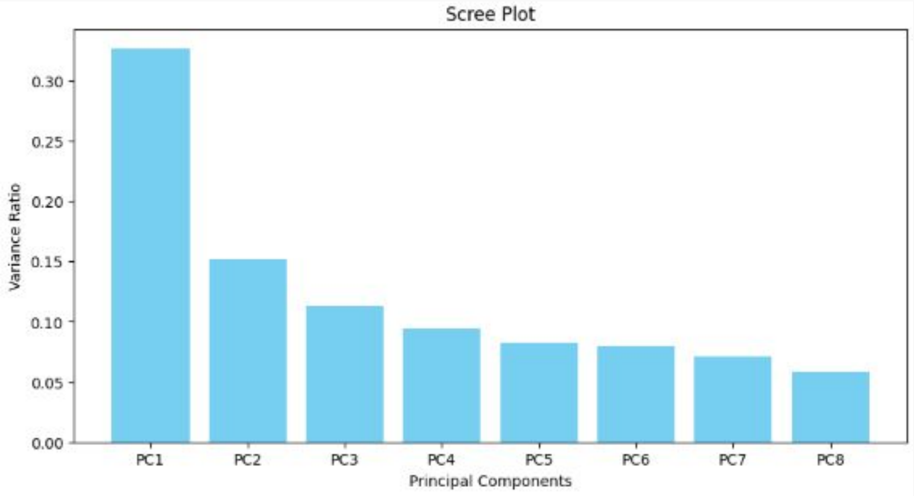
Cluster Profiling Summary: Average # of Items

Cluster	Bakery/ Cereals	Beverages	Dairy/ Eggs	Fresh Produce	Frozen/ Refrigerated	House-hold/ Pet	Meats/ Seafood	Misc.	Pantry/ Dry Goods	Total Shoppers
0	1.409	1.489	2.536	2.068	0.181	2.438	0.944	1.141	1.021	425
1	0.509	0.660	1.048	0.827	0.042	0.332	0.382	0.546	0.512	1372
2	2.103	2.503	4.032	3.188	0.357	0.970	1.751	2.247	2.286	437
3	1.161	1.296	2.167	1.678	1.229	0.530	0.896	0.959	0.991	645
4	1.448	1.345	2.421	2.073	0.000	0.389	1.067	1.031	0.966	1019

Visualizing Clusters: Cluster 0

An Alternative: Principal Component Plotting





PCA: Loading Scores

	PC1	PC2
Bakery/Cereals	0.263	-0.154
Beverages	0.283	-0.215
Dairy/Eggs	0.706	0.661
Fresh Produce	0.441	-0.680
Frozen/Refrigerated	0.047	-0.018
Household/Pet	0.140	-0.074
Meats/Seafood	0.191	-0.058
Miscellaneous	0.224	-0.071
Pantry/Dry Goods	0.220	-0.126

High Positive Load

Dairy /Eggs: **high positive** load for both PCAs

Contrasting Loads

Fresh Produce: **high positive** load PC1, **high negative** load PC2

Insignificant Loads

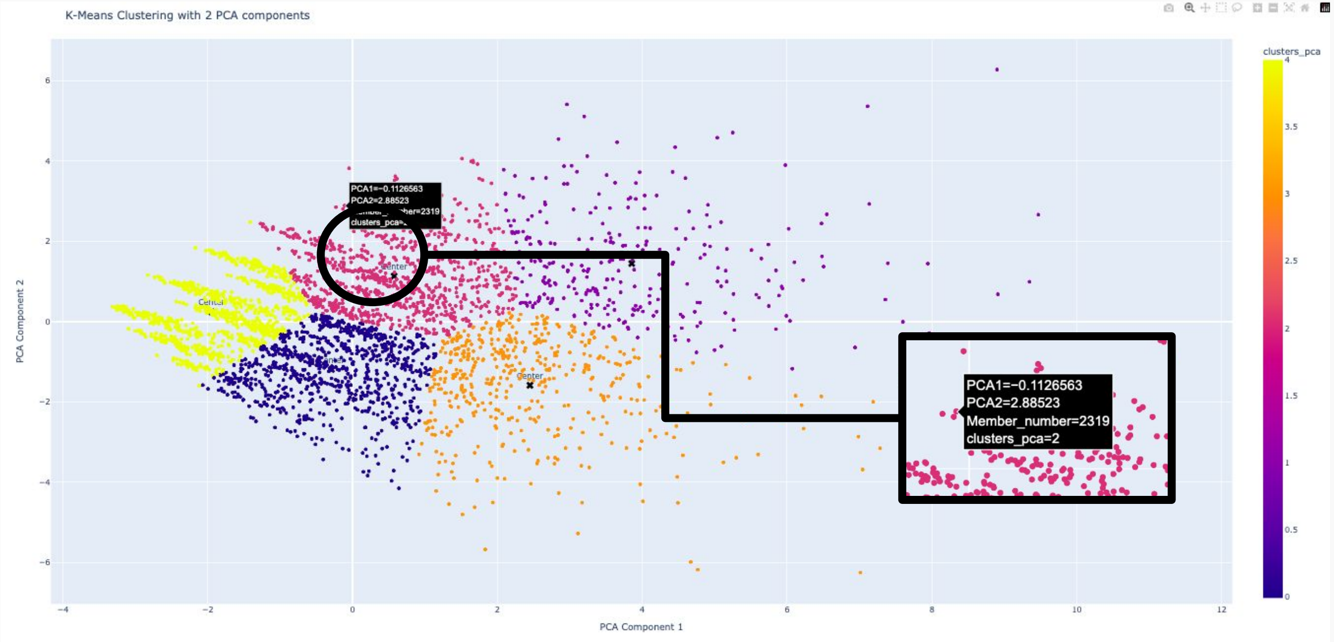
PC2: Household/Pets, Frozen/Refrigerated, Miscellaneous…

Further Visualizations: PCA Plot

Variance PC1 + PC2

Cumulative variance captured roughly **50%**

Further Visualizations: Python Interactive Aspect



Cluster Takeaways: Consumer Recommendations

Profiling: Know your consumer's cluster profile: their cluster profile reveals a lot!

Item Frequency: Not all baskets are big: some shoppers are selective while others shop for many items

Dairy is reliable: Most consumers purchase eggs/dairy: if you cannot understand your consumer, advertise dairy/eggs

Frozen is unreliable: Most consumers do not purchase frozen meals: only select customers do

Consumer Recommendations: Cluster 0

Dairy/Eggs

Suggest more dairy/egg products such as artisan cheeses and yogurts on offer

Household/Pets

Suggest the newest pet food and household cleaning items

Fresh Produce

What fruit and vegetables are on offer? Make sure this is being highlighted

Frozen Foods

Do not waste precious advertising space on frozen foods since this consumer rarely purchases them!

Final Remarks

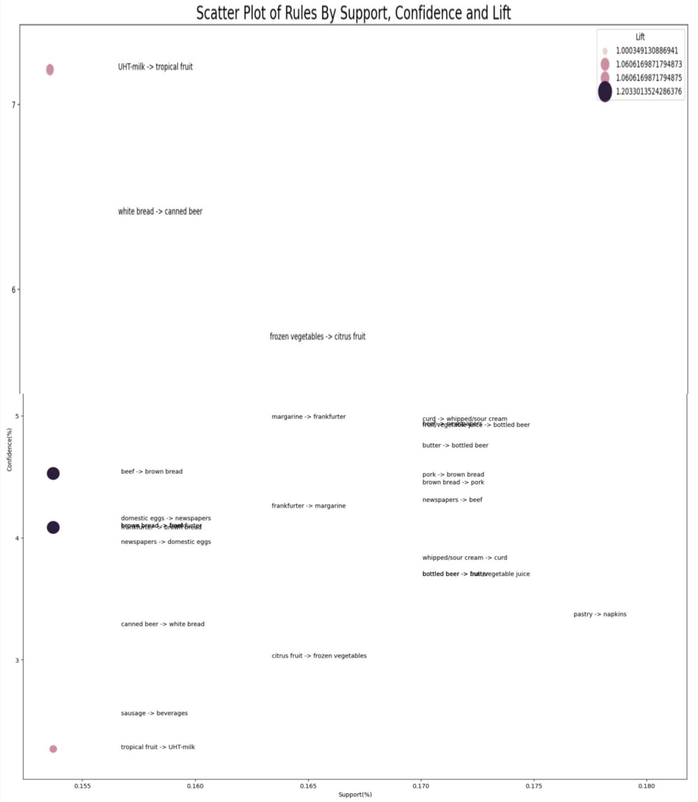
- **Lift values** provide insights into **item associations, aiding store layout** redesign

- In our digitally-driven world, **clustering** is essential for precise and efficient **customer product recommendations**. By segmenting customers based on their preferences, we would be able to offer product recommendations that are most often bought by the segment

- By combining **clustering** with **lift** data, we could identify item pairs and put together **promotional offers** on the store's website/app to help boost sales of highly associated products

QUESTIONS?

Appendix

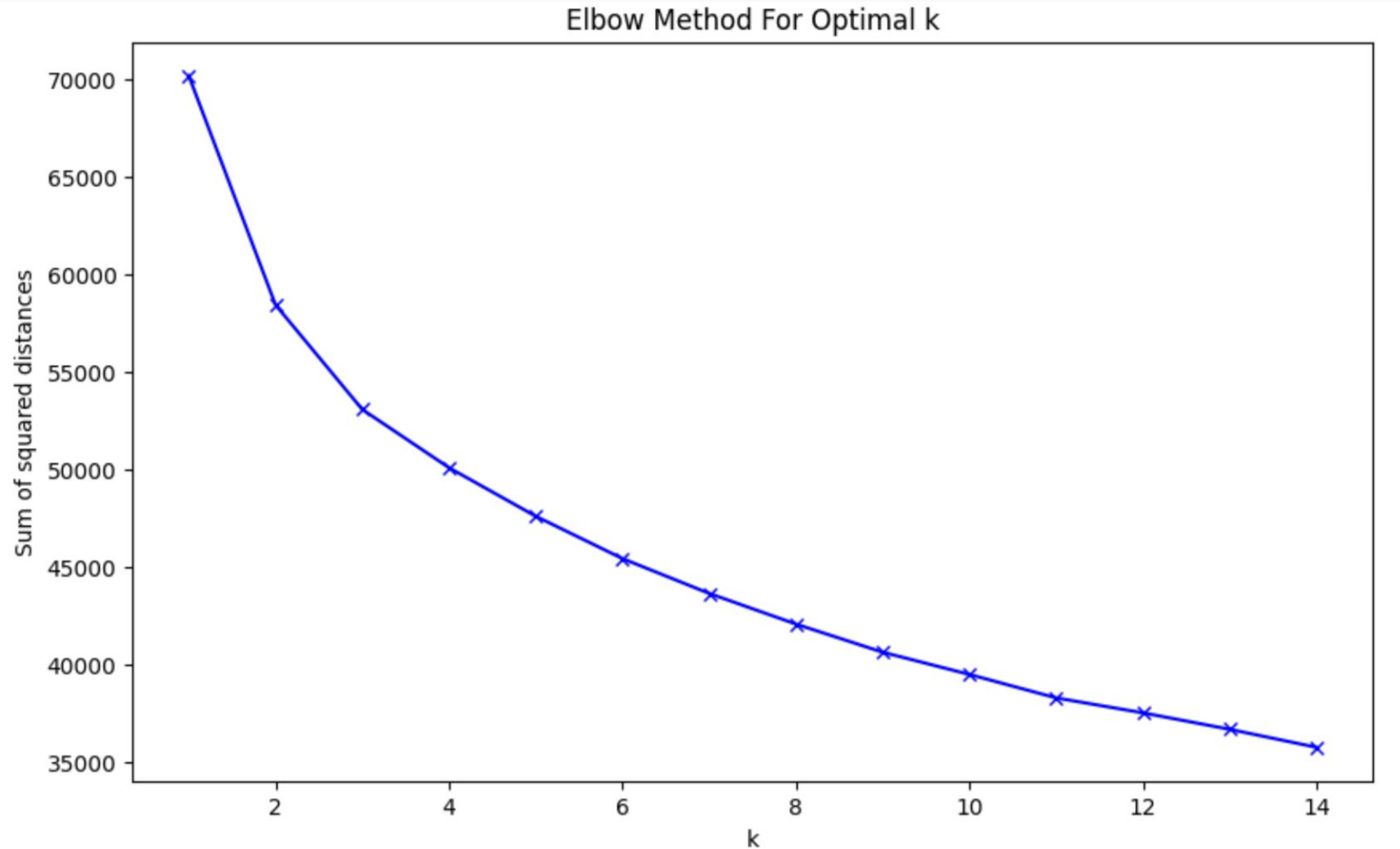


We initially attempted to visualize the association rules using a scatter plot based on support, confidence, and lift. However, the resulting plot proved to be less intuitive and challenging to interpret. As a result, we opted not to utilize the graph for our analysis.

Our Code

```python
# Since the previous code execution wasn't displayed, I'll define the categories and map the items again.
# Define the categories
categories = {
    'Fresh Produce': ['tropical fruit', 'pip fruit', 'other vegetables', 'citrus fruit', 'root vegetables', 'berries', 'herbs', 'grapes', 'onions', 'potato products', 'specialty vegetables'],
    'Dairy and Eggs': ['whole milk', 'butter', 'butter milk', 'yogurt', 'curd cheese', 'processed cheese', 'curd', 'hard cheese', 'cream cheese', 'UHT-milk', 'domestic eggs', 'margarine', 'whipped/sour cream', 'sliced cheese', 'specialty cheese', 'spread cheese', 'soft cheese
    'Bakery and Cereals': ['rolls/buns', 'brown bread', 'pastry', 'baking powder', 'white bread', 'semi-finished bread', 'zwieback', 'waffles', 'long life bakery product', 'cereals', 'cake bar'],
    'Meats and Seafood': ['beef', 'frankfurter', 'chicken', 'hamburger meat', 'pork', 'ham', 'turkey', 'fish', 'meat', 'frozen fish', 'canned fish', 'liver loaf', 'organic sausage', 'meat spreads'],
    'Beverages': ['fruit/vegetable juice', 'canned beer', 'coffee', 'misc. beverages', 'red/blush wine', 'soda', 'sparkling wine', 'bottled beer', 'white wine', 'liquor', 'liquor (appetizer)', 'prosecco', 'brandy', 'rum', 'liqueur', 'cocoa drinks', 'tea'],
    'Pantry and Dry Goods': ['packaged fruit/vegetables', 'chocolate', 'specialty bar', 'flour', 'sugar', 'frozen meals', 'detergent', 'pasta', 'finished products', 'condensed milk', 'cleaner', 'ice cream', 'candy', 'salt', 'oil', 'vinegar', 'nuts/prunes', 'hygiene articles',
    'Frozen and Refrigerated': ['frozen potato products', 'frozen vegetables', 'frozen chicken', 'frozen dessert', 'frozen fruits'],
    'Household and Pet': ['pot plants', 'dog food', 'hair spray', 'photo/film', 'shopping bags', 'dish cleaner', 'pet care', 'female sanitary products', 'cling film/bags', 'soap', 'house keeping products', 'decalcifier', 'cat food', 'bathroom cleaner', 'dental care', 'roll pr
    'Miscellaneous': ['bottled water', 'sausage', 'newspapers', 'popcorn', 'beverages', 'dessert', 'specialty chocolate', 'whisky', 'chocolate marshmallow', 'bags', 'honey', 'nut snack']
}

# Map each product to its category
item_to_group = {}
for category, items in categories.items():
    for item in items:
        item_to_group[item] = category

# Now we have a mapping of products to their categories
item_description = [
    "tropical fruit", "whole milk", "pip fruit", "other vegetables", "rolls/buns", "pot plants",
    # ... (continues with the rest of the items)
]

# Create the category variable for each item description
category_variable = [item_to_group.get(item, 'Miscellaneous') for item in item_description]

# Display the full mapping of 'ItemDescription' to the new category variable
item_description_to_category = list(zip(item_description, category_variable))
item_description_to_category[:10]  # Show only the first 10 items for brevity

# Assuming 'df' is the DataFrame that contains the 'ItemDescription' column.
# We'll need to create a function to map each item description to its category, using the 'item_to_group' dictionary.

def map_item_to_category(item):
    return item_to_group.get(item, 'Miscellaneous')

# Now, let's apply this function to the 'ItemDescription' column to create a new 'category_variable' column
# I'll mock a small DataFrame here as an example.
import pandas as pd

# This assumes 'df' is your full DataFrame that contains 'ItemDescription' and other columns you want to keep.
df['category_variable'] = df['itemDescription'].apply(map_item_to_category)
```

https://colab.research.google.com/drive/1Sq3mMJtERBc3CvxoTNpznngHezvxJ-2W?usp=sharing



Elbow Method For Optimal k
70000
65000
60000
55000
50000
45000
40000
35000
Sum of squared distances
2
4
6
8
10
12
14
k

Marketing Analytics Project Proposal

Group H: Ali Khan, Alicia Wilson, Luis Villazon, Morgan Tucker, Vi Tran

Research Question: How can we utilize market basket analysis to reveal complex product associations within customer purchase patterns and utilize this knowledge to offer personalized product recommendations, thereby enhancing cross-selling opportunities?
Methodology: We will employ association rule mining algorithms to identify frequent itemsets from the available transactional data. Subsequently, generated association rules will be evaluated based on support and confidence thresholds, enabling us to provide tailored product recommendations for cross-selling opportunities.

Data & Description: https://www.kaggle.com/datasets/heeraldedhia/groceries-dataset

- The dataset is a simple one with three columns: Member ID, Date, and Item Description
- It contains 38,765 rows of purchase data from grocery stores