Product Name Match and Item Relationship Exploration on Transaction Logs

MSA Practicum NCR Corporation Project Final Presentation

Team: Pedro Pinto, Hanna Hamilton, Qihang Zhang, Xinze Wang, Jingyu Li

Content

NCR looks to standardize product names from various vendors and generate insights for merchant operations

Background

- NCR serves thousands of retailers and restaurants which all have their own product catalogs.
- There is a lack of consistent product names and categories across product catalogs from different vendors, leading to poor decision making, excess labor expended on manual catalog curation, supply chain errors, and mistakes that decrease customer satisfaction.

Business Value

- To resolve these inconsistencies and to conduct analysis on vendors' transaction logs, it helps NCR create more data-driven product offerings and add-value services that improve and automate merchant operations.
- Win-win for NCR and the clients.

The project is broken down into two sub-problems: product name match and item relationship exploration

Objective

Apply the techniques in data analytics and machine learning

Product Name Match

- Given a master catalog including the generic name of different products, how to match the name variations of the identical product from different vendors to the generic name in the master catalog

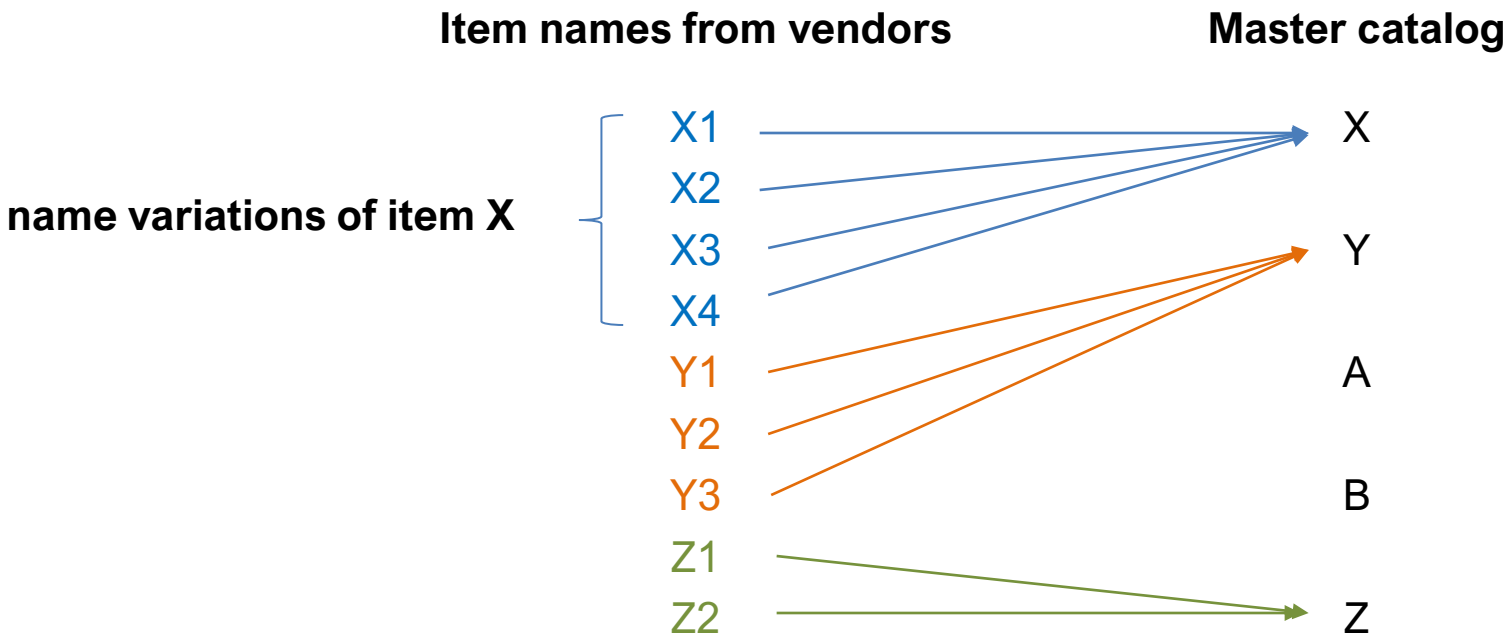
Item Relationship Exploration

- How to identify meaningful relationships and connections between product items, which forms the basis of product recommendation.

Content

Task: Match name variations of an identical product to a generic name

- Given a master catalog including the generic name of different products, how to match the name variations of the identical product from different vendors to the generic name in the master catalog.



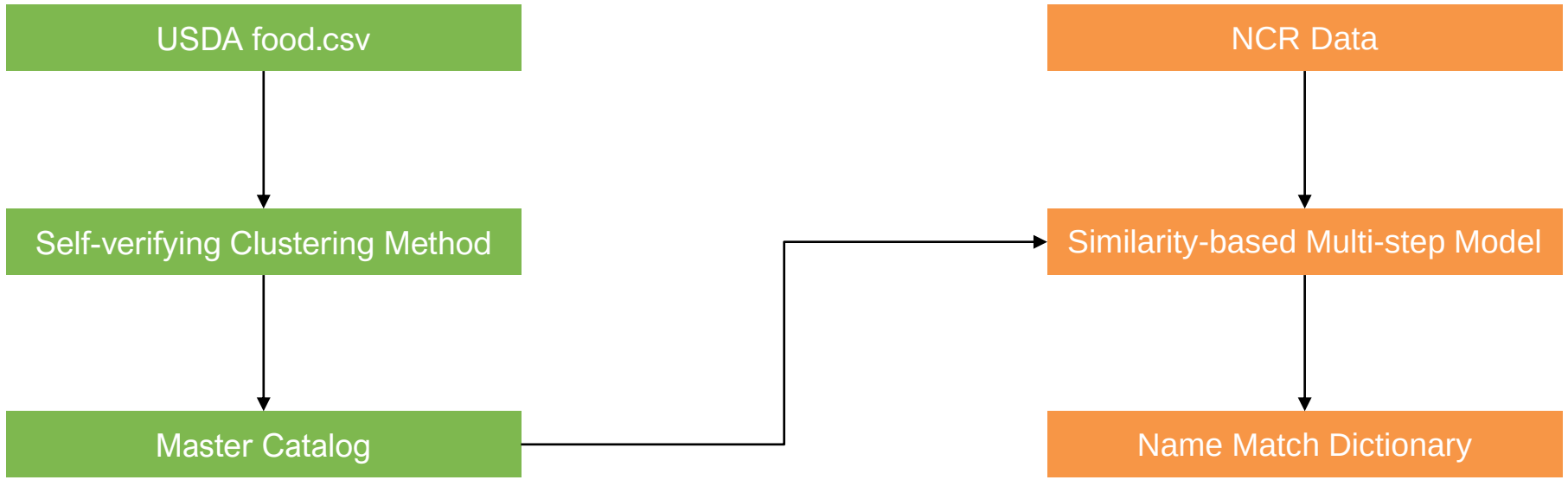
Approach overview: Similarity-based Multi-step Model

Prerequisite Task: Master Catalog Generation

A grocery product catalog at UPC level which contain different items without any duplicated items.

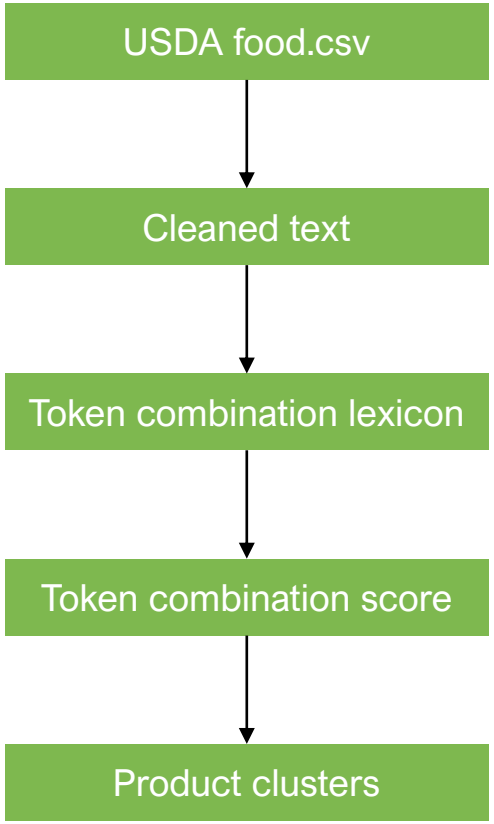
Core Task: Product Name Match

Match the name entries in NCR data to the master catalog.



Master catalog generation: Group similar product names into clusters

- Key algorithm: Generate token combinations for each original product name, then score each combination and choose the combination with highest score as the generic name tag for the original product name. Product names with the same tag are clustered into the same group.



Tokenization:
- Lower case
- Replace unit token to abbreviate form (e.g. ounce → oz)
- Remove all punctuations in the product name except dot in decimal, % in percentage and line in fraction
- Remove the space between a number and a unit token

$$I(c) = \frac{kY_c^2}{a + d_c/f_c} \log(f_c),$$

$$in\ which\ Y_c = \sum_{w \in c} idf(w) \frac{1/Q_{\text{type}}}{1 - b + bk/\bar{l}_c}$$

Cluster result example

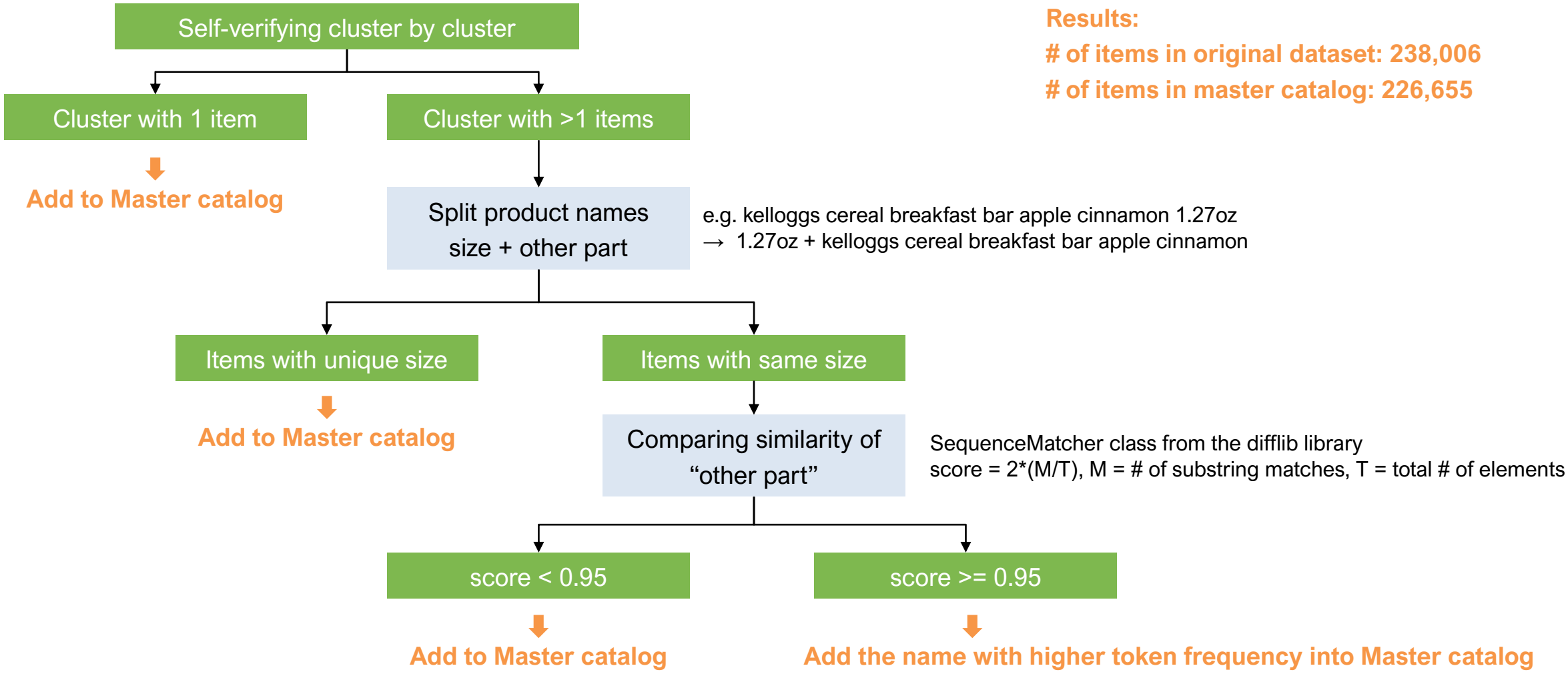
honeysuckle ground turkey 85% :
- ['honeysuckle whtie 85% ground turkey',
- 'honeysuckle white 85% ground turkey',
- 'honeysuckle white fresh ground turkey 85%']

sunshine cheezit crackers hot spicy :
- ['sunshine cheezit crackers hot spicy 7oz',
- 'sunshine cheezit crackers hot spicy 12.4oz',
- 'sunshine cheezit crackers hot spicy 1.65oz',
- sunshine cheezit crackers hot spicy 3oz',
- 'sunshine cheezit crackers hot spicy 9oz',
- 'sunshine cheezit crackers hot spicy 3.25oz']

frosted mini donuts :
- ['mini frosted donuts',
- 'frosted mini donuts',
- 'freihofers frosted mini donuts',
- 'mini donuts frosted',
- 'sweetsixteen frosted mini donuts']

Akritidis, L., Fevgas, A., Bozanis, P., & Makris, C. (2020). A self-verifying clustering approach to unsupervised matching of product titles. Artificial Intelligence Review, 1-44.

Master catalog generation: Self-verify to remove duplicates



Results:

# of items in original dataset: 238,006

# of items in master catalog: 226,655

Akritidis, L., Fevgas, A., Bozanis, P., & Makris, C. (2020). A self-verifying clustering approach to unsupervised matching of product titles. Artificial Intelligence Review, 1-44.

Product name match: Similarity-based multi-step model

Step1: N-gram Tokenization

- Product names are broken up into substring tokens of predetermined lengths called N-grams. The value of N is a hyperparameter.

Example

“Heinz Ketchup” 5-gram
[‘Heinz’, ‘einz ’, ‘inz K‘, etc.]

Step2: TF-IDF Vectorization

- Term Frequency - Inverse Document Frequency assigns higher value for terms (n-grams) that appear on very few product names and lower value for common terms. The resulting matrix has one row per product name and one column per unique n-gram.

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF
Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

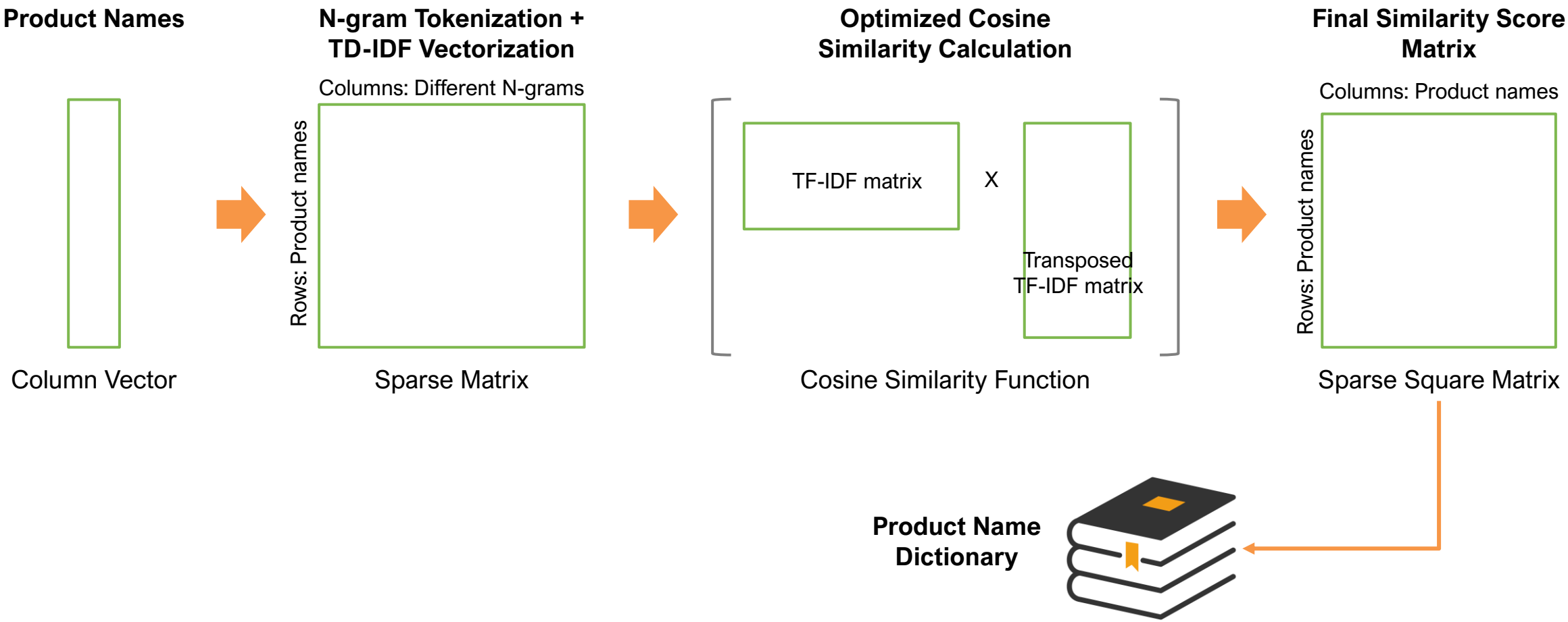
Step3: Optimized Cosine Similarity

- To match different product name vectors, we used an optimized implementation of the cosine similarity that takes advantage of the sparsity of the TF-IDF matrix. We can also input a minimum matching score threshold and a maximum number of top matches per product.

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

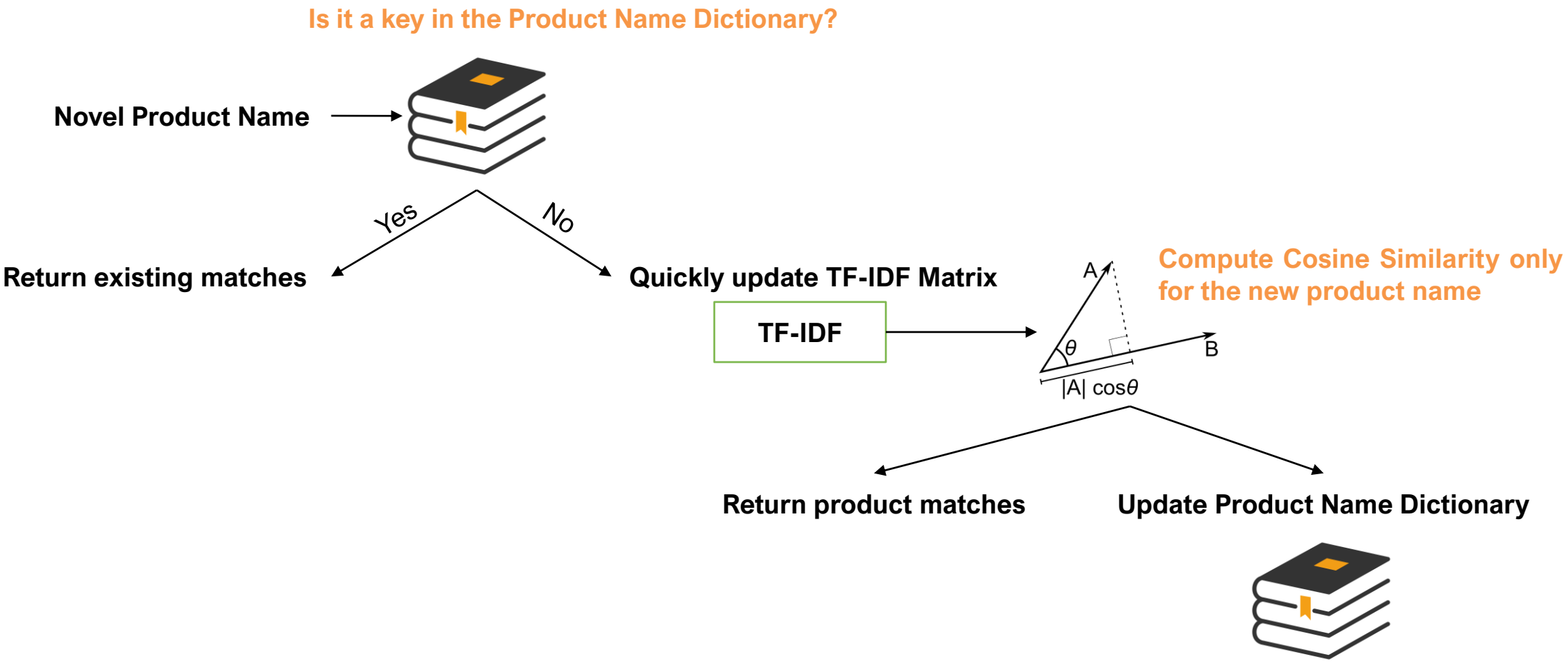
Product name match: Similarity-based multi-step model

- Model diagram:



Product name match: Similarity-based multi-step model

- How to deal with novel product name in the future:



Product name match: Similarity-based multi-step model

- Results:

Merging NCR dataset with Master Catalog:

- Size: 275,206 product names
- Negligible time difference when computing TF-IDF matrix.
- Considerable time increase when computing final similarity matrix (from ~5 seconds to ~15 minutes) compared with using loops to compute pairwise similarity.
- Can be performed in batch in the background.
- To achieve same level of granularity, the threshold was set in the range of 0.80 - 0.83

Example of Product Name Dictionary

{**'kelloggs poptarts wild cherry 14.1oz':**
[['kelloggs poptarts wild cherry 14.1oz',
'kelloggs poptarts wild berry 14.1oz',
'kelloggs poptarts root beer 14.1oz',
'kelloggs poptarts chocolate caramel 14.1oz'],
[1.0, 0.954, 0.753, 0.752]],

'kellogg poptarts frosted strawberry 22oz':
[['kellogg poptarts frosted strawberry 22oz',
'kelloggs poptarts frosted strawberry 22oz',
'kellogg poptarts spring strawberry 22oz',
'kelloggs poptarts strawberry 22oz',
'kelloggs poptarts frosted cherry 22oz',
'kelloggs poptarts frosted blueberry 22oz'],
[1.0, 0.928, 0.855, 0.837, 0.837, 0.807]],

'kellogg poptarts frosted strawberry slurpee 3.67oz':
[['kellogg poptarts frosted strawberry slurpee 3.67oz',
'kelloggs poptarts frosted blueberry slurpee 3.67oz',
'kelloggs poptarts frosted strawberry 3.67oz'],
[1.0, 0.873, 0.760]],

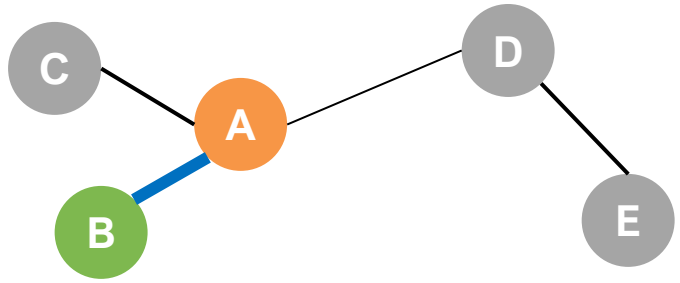
Content

Task: Find complementary or substitute relationships based on transaction logs

- Identify meaningful relationships and connections between product items, which forms the basis of product recommendation.
  - Since there's no opportunity for us to evaluate the results based on ground truth or through online experiments, we mainly focused on implementing different algorithms, which could form the basis for NCR to further explore in the production level.

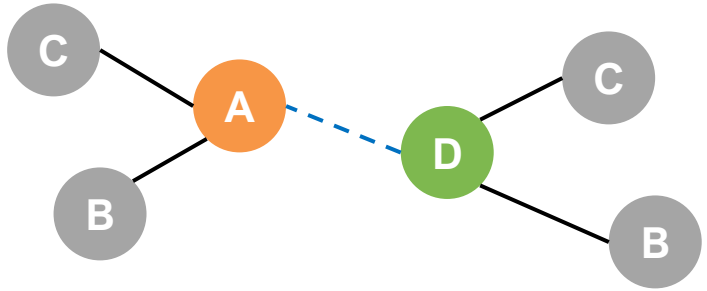
Type 1: Complementary Product

Given a target product, find another product or a group of other products that people usually buy and use together, which can add values to the target product.



Type 2: Substitute Product

Given a target product, find another product or a group of other products that serve the same purpose or need as the target product.



Complementary relationship: Association rule mining by Apriori Algorithm

- Algorithm introduction:

Set a frequency threshold
- Called minimum support count
- Model hyperparameter

Select all frequent items
- Select items which have frequencies >= the minimum support count

Itemsets with size k = 2, 3, 4, …
- Create itemsets of k=2 from frequent items from previous iteration
- Select itemsets which have frequencies >= the minimum support count
- Increase k by 1 and repeat until no more frequent itemsets found or stopping criterion

Identify complementary relationships by calculating lifts of itemsets

$$Lift(\{X\}->\{Y\}) = P(X,Y)/(P(X)*P(Y))$$

A lift well above 1 indicates a strong complementary relationship between X and Y.

Complementary relationship: Association rule mining by Apriori Algorithm

- Results:

Top 5 Complementary Relationships Based on All Transactions or Small Transactions

All Transactions		Small Transactions	
Rule	**Lift**	**Rule**	**Lift**
yellow bell pepper -> (green bell pepper, red bell pepper)	69.5	red bell pepper -> yellow bell pepper	180.6
espinazo fresco de puerco -> patitas frescas de puerco	69.0	green bell pepper -> red bell pepper	82.9
red bell pepper -> (green bell pepper, yellow bell pepper)	68.7	costillas de puerco -> trocitos frescos de puerco	48.8
red bell pepper -> yellow bell pepper	57.2	chayote squash -> mexican squash	44.1
chamorrow de res -> espinazo de res fresco	43.5	carrots -> chayote squash	34.6

We found 14,176 complementary relationships (with lifts greater than or equal to 1.5) with all transactions and 490 complementary relationships (with lifts greater than or equal to 1.5) with small transactions.

Substitute relationship: Transaction Co-occurrence Similarity Approach

- Key idea: Regard the other items appearing in the same transaction as the context of the target item. Thus, substitute relationship satisfies that 1) the two items appear in similar context, which means they appear with a similar basket of other items, and 2) the two items rarely appear in the same transaction.

Step1: Item Co-occurrence Matrix

# of times the product pair appears in the same transaction

	A	B	C	D	E	F
A	0	1	**3**	**6**	**2**	**5**
B	1	0	**2**	**5**	**2**	**5**
C	**3**	**2**	0	18	**22**	**0**
D	**6**	**5**	18	0	**7**	**20**
E	2	2	22	7	0	3
F	5	5	0	20	3	0

Step2: Substitute Score

Context similarity

$$score = \frac{X_i \cdot X_j}{\|X_i\|\|X_j\|} e^{-\frac{C_{ij}}{\min(C_{ii}, C_{jj})}}$$

Co-occurrence penalty

Substitute relationship: Transaction Co-occurrence Similarity Approach

- Results:

Example of Substitute Relationships

Target Product	Coke Classic, 20 Fl Oz		Coca-Cola Cherry, 2 Liter		Lay's Classic Potato Chips, 8 Ounce		HEINZ KETCHUP, 38 OZ	
	Product	Scores	Product	Scores	Product	Scores	Product	Scores
Top 10 Substitute	Coca Cola, Coca Cola Classic Bottle, 16.9 Fl Oz	0.986	Jarritos Mineral Water Drink 1.5 Lt	0.958	Barcel Takis Fuego, 4 Ounce	0.929	Minute Maid Juice, Berry Punch, 128 Oz	0.844
	Jarritos, Mineragua, Club Soda, 12.5 Ounce	0.986	Jarritos, Soda Grapefruit, 1.5 Liter	0.954	KINDER JOY CANDY, 0.7 OZ	0.928	Clamato Tomato Cocktail From Concentrate, 32 oz	0.840
	Mexican Coke Glass Bottle, 12 fl oz	0.984	Jarritos Soda, Mandarin, 1.5 L Bottle	0.952	Cheetos, Flamin Hot Crunchy, 8.5 Ounce	0.928	Minute Maid All Natural Fruit Punch, 128 Oz	0.839
	Monster Energy, 16 Ounce	0.980	Cactus Cooler 2 Liters, 67.63 Fl Oz	0.950	MINI CONCHITAS, 18 OZ	0.926	Ocean Spray Juice Cocktail, Cranberry, 101.4 oz	0.828
	Squirt 12 Fl Oz Glass Bottle	0.977	Manzanita Sol Apple Soda, 68 Ounce	0.946	Guerrero Tortillas Riquisima, 24 Count	0.925	SIMPLY LEMONADE, 52 OZ	0.826
	Jarritos Mandarina Soft Drink, 12.5 oz.	0.976	7UP, 2 L Bottle	0.945	Barcel, Fuego Takis Chips, 9.9 oz	0.924	Sunny Delight Beverage, Florida Style, 128 Ounce	0.826
	Sidral Mundet, Soda Apple, 12 Fl Oz	0.974	Jarritos, Pineapple Soda, 67.63 Fl Oz	0.943	2VL CHESTERS HOT FRIES, 5.25OZ	0.924	Mott's Inc Clamato Picante, 32 oz	0.822
	Red Bull Energy Drink	0.972	Jarritos Tamarindo Soft Drink	0.942	family store_brand grade aa medium eggs, 12 cnt	0.923	Camaronazo Picante / Spicy Tomato Shrimp Cocktail	0.820
	Monster, Energy Zero Ultra, 16 Fl Oz	0.972	Sunkist Orange Soda, 2 L bottle	0.942	Cheetos Flamin Hot Limon, 9.7 Ounce	0.922	Clamato, Tomato Cocktail, Picante, 64 Ounce	0.820
	Mexican Pepsi 12 Fl Oz	0.972	A&W Root Beer, 2 L bottle	0.941	store_brand corn tortillas 90 count	0.921	Simply Lemonade, Simply Lemonade, 89 Ounce	0.814

Substitute relationship: GraphSWAG model

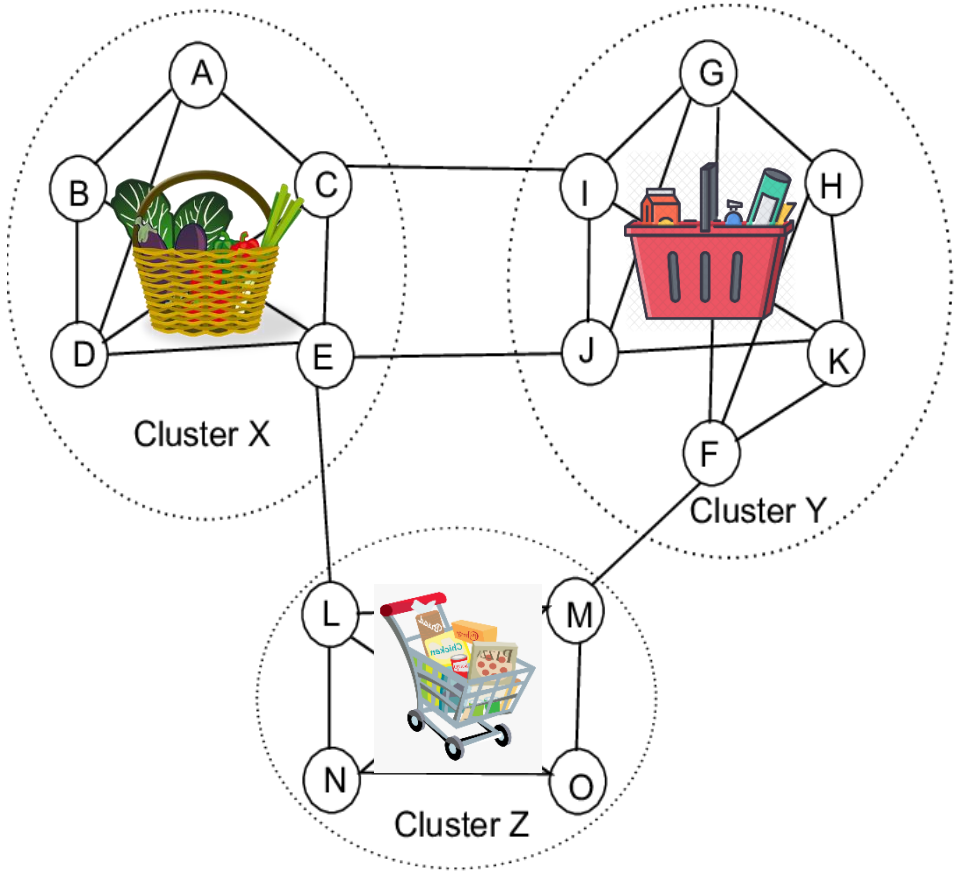
- **Introduction:**

The purpose of this approach is to convert the transactional data into a weighted graph, and mine it to find suitable substitutes for any given product.

Based on our research, **GraphSWAG**[1] is a good algorithm for mining transactional graphs [2].

GraphSWAG is a **Graph Convolutional Network (GCN)** developed by researchers at Target that combines sales data, product descriptions and product images to generate node embeddings for large graphs.

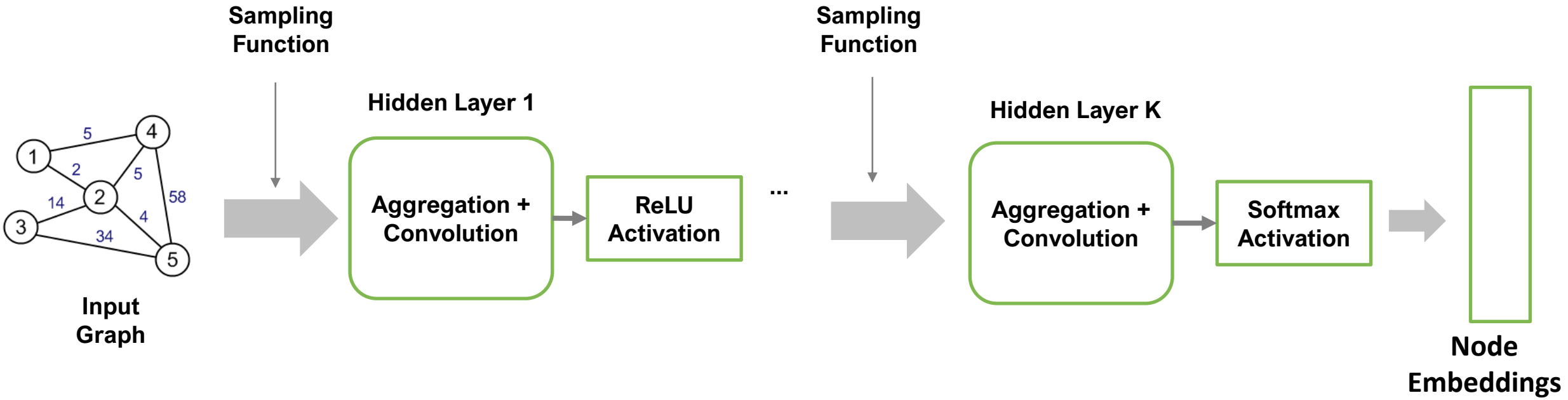
Graph Structure:
- **Nodes**: individual products.
- **Edges:** pairwise co-occurrence of products weighted by frequency



[1] Pande Et Al. - SWAG: Item Recommendations using Convolutions on Weighted Graphs
[2] Pande Et Al. - Substitution Techniques for Grocery Fulfillment and Assortment Optimization Using Product Graphs

Substitute relationship: GraphSWAG model

- **GCN structure:**



Substitute relationship: GraphSWAG model

- **First steps:**

Generating the Graph:

•**Nodes**: Individual Products

- Non-food items such as plastic bags were removed
- Items without names were removed

•**Edges**: Weighted Pairwise Co-Occurrence

- Weights are calculated by adding co-occurrent transactions adjusted by **time-decay** and transforming them to a [0,1] range using the **arctan** function

$$w(i,j) = arctan\left(\sum_{t=1}^{t=N} \frac{1}{Rec(t)}\right)$$

$w(i,j)$: weight between products i and j
t: t_{th} co-occurrent transaction
$Rec(t)$:Recency of the t_{th} transaction in $days$

Generating the Word Embeddings:

•Item names were standardized by lower casing and removing special characters

•Used Gensim's implementation of Word2Vec to convert item names to 200 dimensional vectors

Substitute relationship: GraphSWAG model

- **Algorithm introduction:**

Sampling
- K neighbors of each node are sampled before each layer.
- A beta hyperparameter assign a higher or lower importance to the edge weights when generating sampling probabilities.

Aggregation
- The feature vectors of all sampled neighbors of a node a combined with some aggregation function.
- We used the Mean aggregator, similarly to the Target paper.

Concatenation
- Each aggregated feature vector is concatenated to the output of the previous layer before being input to the next layer.

Layer Weights + Activation Function

Final Output: 200-long feature vectors embedding each node of the graph.

These features vectors are used to build a ***Nearest Neighbors*** models with a neighborhood of size *n*. The *n* neighbors of each product will represent its substitute recommendations.

Substitute relationship: GraphSWAG model

- **Loss function and clustering:**

Loss function

$$\mathcal{L}_{\mathcal{G}}(z_u) = -r(u,v)^{\alpha} \log(\sigma(z_u^{\top} z_v)) - Q \cdot \mathbb{E}_{v_n \sim \mathbb{P}_n(v)} \log(\sigma(-z_u^{\top} z_{v_n})),$$

- z_u: vector representation of node v (output from aggregation algorithm)
- r(u,v): geometric mean of weights along edges connecting u and v (u and v may not be neighbors)
- Alpha: hyper-parameter in [0,infinity)
- Sigma: sigmoid function
- Q: number of negative samples
- This loss function encourages nearby nodes to have similar representations, while enforcing that the representations of different nodes are highly distinct.

Clustering

- We used k-means clustering to learn which items are likely to be actual substitutes (same cluster) and which items are not likely to be actual substitutes (different clusters).
- We clustered using the following variables: dept_num, item_price, qty_is_weight, category, item_type
- For a given item, a positive sample is from within its cluster and a negative sample is from outside its cluster.

Substitute relationship: GraphSWAG model

- **Preliminary Results Examples (without training the network):**

Substitutes to **MINUTE MAID PREM OJ ORIGINAL**:

- **FLORIDAS NATURAL OJ W/PULP**
- COLGATE TOTAL TP CLN MINT
- KNORR LETRAS SOUP
- CACIQUE RANCHERO QUESO FRESCO
- **CRUSH PINEAPPLE**

Substitutes to **STEAK EYE ROUND**:

- PENCA DE MAGUEY
- fc pasta vermicelli box
- DAWN DISH ULTRA APPLE BLOSSOM
- JIMMY DEAN SAUSAGE EGG CHS CROISSAN
- TRES GENERACIONES PLATA

Substitute relationship: GraphSWAG model

- Ideas for future work:

    1. Train Word2Vec instead of using pre-trained model

    2. Incorporate item descriptions and item images

    3. Tune hyper-parameters
        1) Length of item vectors in Word2Vec
        2) Number of neighbors in sampling algorithm
        3) Beta in sampling algorithm
        4) Gamma in aggregation algorithm
        5) Alpha in loss function
        6) Number of layers

Thank you for your kind attention