APMTH 220 Geometric Methods for Machine Learning **Spring 2024**

Graphing Substitute-Complement Relationships

Name: Omar Abdel Haq *Submission Date: 05/03/2024*

1 Abstract

This project explores the dynamics of product relationships in modern markets, focusing on the distinctions between complementarity and substitution. Leveraging online purchase data from Amazon, we employ link prediction algorithms using Graph Neural Networks (GNNs) to infer these relationships with reasonable accuracy. By analyzing customer co-viewing and co-purchase behaviors, we identify pairs of items that exhibit strong indications of being complements or substitutes. The robustness, utility, and efficacy of using GNNs in modeling product relationships is proven with a replicable implementation codebase.

2 Introduction

This project explores the possibility of modeling the relationships between products, namely in regards to complementarity and substitution. Complementary items are products that are typically used together, enhancing the value or utility of each other. They have a positive relationship, meaning the demand for one product increases the demand for the other, boosting each other's consumption and creating a mutually reinforcing dynamic. On the other hand, substitute items are products that can be used interchangeably to fulfill a similar need or purpose. They have a negative relationship, where the increase in demand for one product reduces the demand for the other, creating a situation where they vie for market share [1].

Understanding whether products are substitutes or complements is crucial for business owners for a multitude of reasons. One use case is in that recognizing substitutes allows businesses to adjust prices more strategically. If two products are substitutes, a price reduction in one might attract customers away from the other. Conversely, for complements, pricing strategies can be coordinated to maximize overall revenue. Another use case is inventory management; knowing the substitutability or complementarity of products aids in choosing which items go on shelves, minimizing redundancy in choice offerings and informing product placement.

Determining the relationship between products is also useful for economists because it helps in analyzing consumer behavior, market dynamics, and making predictions about how changes in prices or demand for one product might affect another; this helps in understanding purchasing patterns and preferences [2]. Despite these numerous uses, no large scale methods exist to detect common complementary or substitute products.

The overarching project motivation is to determine whether online-purchase data obtained from Amazon can be used to detect with high confidence substitute or complement pairs; for this we use a link prediction approach using GNNs on co-viewing and co-purchase Amazon data. Link prediction algorithms have been utilized for tasks such as suggesting friends on social media platforms, recommending products to users based on their preferences, and predicting future interactions in social networks, but no significant body of literature exists on their utility in evaluating the socioeconomic qualities of

goods [3]. Past explorations focused on predicting the relationship between goods based on price elasticity calculations, but those and other methods have proven difficult to implement on a larger scale in practice due to lack of data [1]. Medium-sized explorations have been suggested by building bipartite graphs modeling baskets of goods and individual products as two kinds of nodes, but the reliability of using the approach on larger scale industrial datasets has not been proven [2]. Our approach, as a result, focuses on inferring these product relationships based on whether or not they were viewed and/or bought together by customers, modeling using simpler homogeneous graphs with one type of edge. This allows for the use of extremely rich feature vectors and massive datasets.

3 Background & Notation

Key Modeling Assumption

This project attempts to model complementary and substitute goods using the Also-View and Also-Buy features of Amazon products, which indicate whether customers viewed sets of products at the same time or purchased them together, respectively. Complementary goods are defined as products that are typically used and/or purchased together, while supplementary goods are alternatives that can substitute for each other, with consumers usually picking between them when making purchases.

The key assumption we make is that goods that are viewed together but not purchased together (with only one of them bought) are likely substitutes. On the other hand, goods that are viewed together and bought together are likely complements. This however, is not always strictly true; some users buy similar items and only keep the one they like, while some pairs of goods are neither substitutes nor complements but are still purchased together. An example of sets of products fitting into each of the three categories are:

- *Substitutes*: Purchasing a blue pair of pants or a navy pair of pants. The items are hypothetically viewed around the same time but only one of them is purchased (so they aren't bought together).
- *Complements*: Purchasing a TV remote along with batteries. The items are hypothetically both viewed and bought around the same time.
- *Noise (Neither Substitutes nor Complements)*: Purchasing a pillow cover and a pair of socks. The products may be viewed or bought during the same shopping session but do not inherently have a relationship connecting them.

The Amazon dataset used will inherently include all of these different types of observation relationships. We use GNN modeling to predict co-viewing and co-buying, in hopes of filtering out those noisy cases. Two types of graph convolutional layers are used in the modeling portion of this project: `GCNConv` and `SAGEConv`.

Graph Convolutional Layer Framework

A GCN layer is a layer with the goal of learning representations for each node in the graph that capture both its own features and the features of its neighboring nodes. This is done via a message-passing function, which in this case is defined as:

$$h_v^{(k)} = \sigma\left(W^{(k)} \sum_{u \in N(v) \cup \{u\}} \frac{h_u^{(k-1)}}{\sqrt{|N(u)| \cdot |N(v)|}}\right)$$

Where $h_v^{(k)}$ represents the representation (or embedding) of node v at the k-th layer of the GCN, $W^{(k)}$ is the trainable weight matrix associated with the k-th layer, $N(u)$ represents the set of neighboring nodes of node u in the graph, and σ is an activation function applied element-wise to the output [4].

The message-passing operation works as follows: For each node v, the GCN aggregates information from its neighboring nodes. This is done by multiplying the feature vectors of the neighboring nodes by the learned weight matrix $W^{(k)}$. The aggregated information from the neighbors is then summed up and divided by the square root of the product of the degrees of the nodes involved. This normalization step ensures that the updates are scaled appropriately based on the number of neighbors each node has. Finally, an activation function σ is applied to the sum to introduce non-linearity into the model. This is all done in the hopes of arriving at meaningful embeddings for the nodes in question [5].

Alternatively, the other graph convolutional layer used in this project is `SAGEConv`. Unlike GCN, which aggregates information from all neighboring nodes, `SAGEConv` operates in a mini-batch setting and samples a fixed-size neighborhood around each node. First, for each node v, a fixed-size neighborhood is sampled. Next, the sampled neighborhood is aggregated to compute the representation of the central node v. The message-passing operation can be mathematically expressed as:

$$h_v^{(k)} = \sigma\left(W^{(k)} \cdot \texttt{CONCAT}\left(\texttt{AGGREGATE}\left(\{h_u^{(k-1)}, \forall u \in \mathcal{N}(v)\}\right), h_v^{(k-1)}\right)\right)$$

Where the additional operation `AGGREGATE` is the aggregation function, which combines information from the sampled neighbors, and the additional operation `CONCAT` denotes concatenation, which combines the aggregated information with the previous representation of the central node v [6].

Link Prediction

This project utilizes link prediction to create the substitute-complement relationship predictor. The process of link prediction typically involves first learning embedding representations of graph nodes (in the case of this project, using the convolutional layers described above). Then, these learned representations are used to estimate the likelihood of a connection between nodes that do not currently share an edge.

Link prediction models are trained by splitting the existing edges into training and validation/test sets, training the GNN model on the training set, and evaluating its performance on the validation set by measuring how accurately it predicts the presence or absence of edges. To do this, during training, "negative edges" are sampled to show the model which relationships are significant and should share a node and which should not [3].

4 Proposed Approach

Dataset Description & Graph Construction

This project uses the Amazon Review Data collection, a dataset compiled by researchers at UCSD [7]. This dataset offers a comprehensive collection of user reviews, product metadata, and connections, offering valuable insights into consumer behavior and product characteristics within the Amazon platform. Each product exhibits the following features: information on the product's price, rating, and bag-of-words vector representations of the product's description and reviews left by customers.

Two homogeneous graphs were created where each node represents products and their associated attributes. One graph's edges represent co-viewing status while the other's represent co-purchasing status. The labels are one-hot-encoded category vectors of each item (not used for the purposes of this project). The final graphs exhibit the following attributes:

Nodes	Features	Also-View Edges	Also-Buy Edges	Categories
58,779	2,674	97,491	101,336	3

Table 1: *Dataset attributes.*

Sample nodes from the constructed graph showcasing example substitute and complement relationships as seen in the dataset can be seen below:

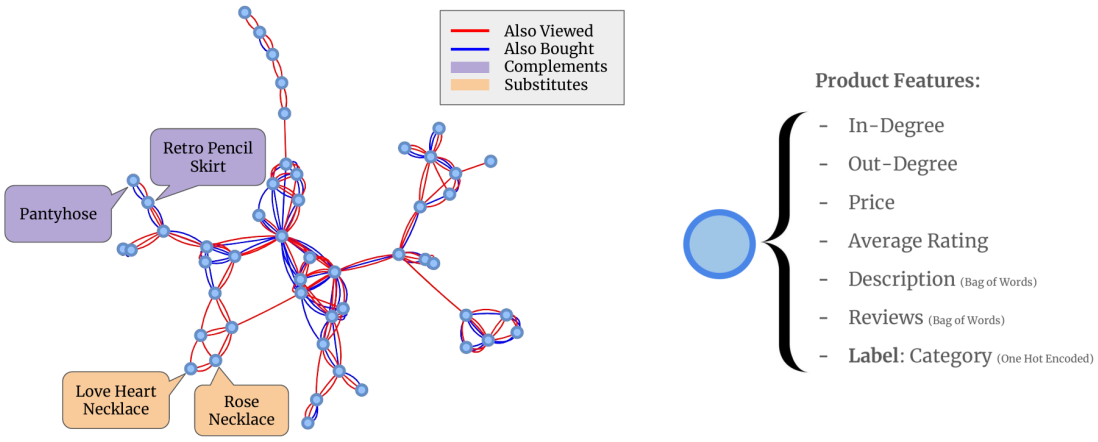

Figure 1: *Sample nodes and their respective features. Items such as pantyhose and skirts exhibit both viewing and purchasing edges, indicating complementarity. Conversely, various necklace designs are linked solely through viewing edges, suggesting substitution.*

Dropout Rate & Robustness to Noise

We train each of our two chosen types of models (one built using `GCNConv` layers and the other using `SAGEConv` layers) on different dropout rates. We do this to select the dropout rate that results in the best performing model on the validation set. This procedure was done for both models, and is used to later show model robustness to noise when injecting fake edges to our data.

The models are constructed as follows: two graph convolutional layers are used, each followed by ReLU activation and dropout (the hyperparameter we are trying to tune), and end with post-message-passing processing involving two linear layers with dropout in between. The GNNs create node embeddings, which are then passed onto a link predictor. The link predictor takes in two embeddings and consists of multiple linear layers with ReLU activation in between. Dropout (0.3) is applied between layers to prevent overfitting, and a sigmoid activation is used to output edge probabilities.

Each training epoch for a model involves iterating over batches of positive and negative edges, concatenating them with optionally added noisy edges, running message passing to update node embeddings, predicting class probabilities for edges, calculating binary cross-entropy loss, and finally updating parameters through backpropagation.

Models with the different types of layers and drop out are trained to select the optimal dropout rate (validation set performance is used as the selection criterion). Afterwards, we train the selected model on datasets with varying amounts of injected noise. In the case of this project, injected noise are edges not existent in the dataset that the model is told are real edges (also-view or also-buy edges, depending on the model in question); this factors in as a penalty in the loss. This is done to show that the model performance is near independent of the amount of noise captured, relying only on the key assumption's

trend to make inferences.

Extracting Product-Level Results

We select two of the trained models (one for co-viewership and the other for co-purchases) and examine its outputted probabilities on edges in the test set. If the key assumption holds, substitute, complement, and noisy observations should be easily discernible.

5 Results & Discussion

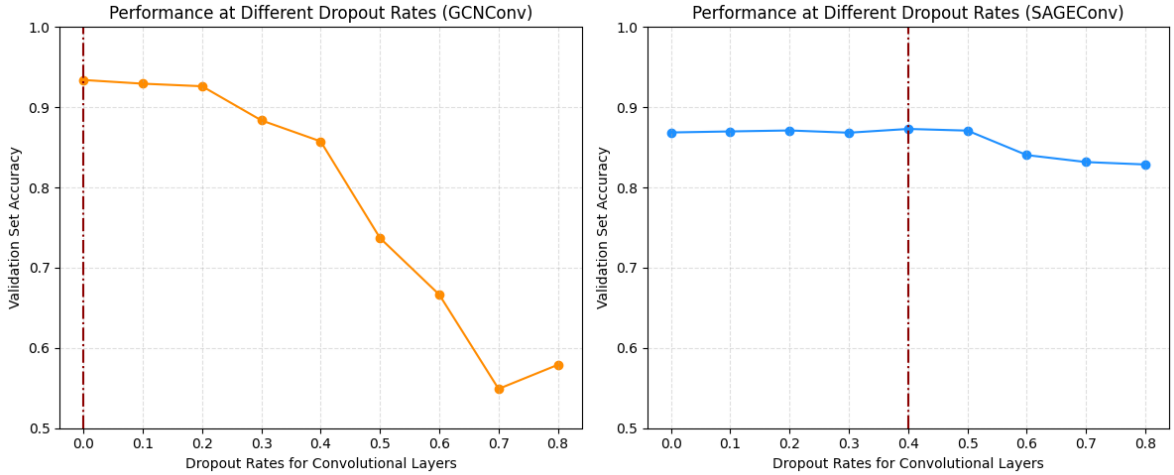

Figure 2: *Sample nodes and their respective features. Items such as pantyhose and skirts exhibit both viewing and purchasing edges, indicating complementarity. Conversely, various necklace designs are linked solely through viewing edges, suggesting substitution.*

We first visualize the result of using different drop out rates on Also-View models built either with `GCNConv` or `SAGEConv`. The models built with `GCNConv` perform best on the validation set with very little dropout (no dropout results in the best model over 0.1 dropout, but only by a slim margin). On the other hand, models built with `SAGEConv` perform best with a medium amount of dropout, with a rate of 0.4 resulting in the best performance. Of the best performing models across both types of networks, the one built using `GCNConv` performs better than the pest performing `SAGEConv` model. This intuitively makes sense, as `SAGEConv` models use only some of the neighbors' information in message passing per epoch to reduce computational complexity, as compared to `GCNConv`'s approach of message-passing via all edges.

After choosing the optimal dropout rates for the different types of models, we then prove the robustness of our modeling to varying levels of noise. The models are trained with either 0%, 5%, or 10% noisy observations during training. For the Also-View edge prediction problem, we notice two trends. The first is that, as seen in the dropout selection simulations, `GCNConv` layers tend to perform better than `SAGEConv` layers when it comes to creating node embeddings. The difference is quite significant, with an approximately 4-5% difference in the validation set's accuracy between `GCNConv` models and `SAGEConv` models. The second main result is that Also-View prediction modeling is quite invariant to noise. For both `GCNConv` and `SAGEConv` models, injecting noise resulted in worse models when evaluated on the validation set. However, the difference in the performance is quite small, much less than a single percentage point of accuracy; this is also multiple times smaller than the performance difference between the two different types of models. This shows that modeling products and their relationships using GNNs

Layer Used	Noise	Also–View Accuracy		Also–Buy Accuracy	
		Train	Validation	Train	Validation
`GCNConv`	0%	98.82%	**92.73%**	87.92%	86.44%
`GCNConv`	5%	99.32%	92.09%	88.96%	**88.34%**
`GCNConv`	10%	98.95%	92.12%	86.59%	87.05%
`SAGEConv`	0%	98.94%	88.11%	99.34%	86.89%
`SAGEConv`	5%	98.78%	87.81%	99.36%	87.56%
`SAGEConv`	10%	99.21%	87.48%	98.93%	87.77%

Table 2: *Model performance on train and validation set edges with different amounts of injected noise, exhibiting strong robustness to added noise.*

is invariant to added noise, and this robustness is more prominent as performance differences are higher based on architecture than they are based on amount of noise added.

Inspecting the results of the Also-Buy prediction problem also reveals some interesting trends. Again, we see a good amount of invariance to noise by noting the validation accuracies of the models at different amounts of noise. Similar to the Also-View models, additional noise either minimally reduces peak validation accuracy, or in some cases marginally improves it, indicating a sort of regularization property that this type of noise injection can act as.

One notable difference, however, is that the validation set accuracies for the Also-Buy problem are lower than those of the Also-View problem. This difference in performance is approximately 5% large, implying that the co-purchase prediction problem is more difficult than the co-viewership problem. From the results of this project alone, it can't be discerned whether this difference is model-based, meaning the problem requires a deeper or complex GNN to capture product relationships, or if the difference is based on lack of relevant features that can aid in the task. The Amazon dataset is rich both textually and in terms of quantitative features – these were not utilized to their full extent as a result of compute resource constraints, and this question poses an interesting future exploration to pursue.

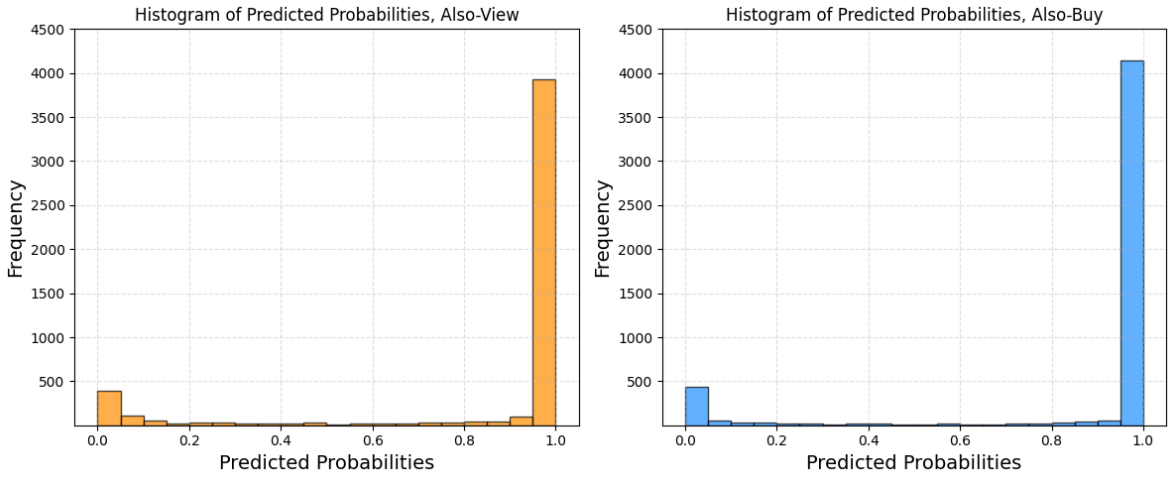

Figure 3: *Output probability distributions for Also-View and Also-Buy problems from best validation set link predictors. All edges in the test sets are positive.*

Of the models tested, we choose the Also-View predictor using `GCNConv` layers and no injected noise, and choose the Also-Buy predictor using `GCNConv` layers and 5% noise. To determine whether the model is working properly, we plot their predicted probabilities of co-viewership and co-purchase status. Given

that the dataset itself is a rich source of noise obscuring the assumed trend, the models should, if they actually captured the trend, predict most of the product pairs as containing a link, but also filter out the noise. The plotted probabilities on a withheld test set (separate from the validation set), are plotted above.

We see that for both models, by far the majority prediction is that two products have an edge; this is consistent given that the entire test set only includes positive edges. However, we also see that the models also output some strong negative predictions; for a set number of product pairs, the models predict no edge even though the dataset itself contained an edge. This makes sense; these negative predictions likely correspond to noisy edges in the dataset which have no underlying relationship beyond being purchased by a random customer.

Next we examine, given our two models, whether they can actually predict substitute and complement relationships. Below is a table of sample products and model predictions for whether or not the products should share an Also-View and/or Also-Buy edge.

Product A	**Product B**	**Co-View Prediction**	**Co-Buy Prediction**	**Relationship**
Leather Pendant Necklace	Cross Shield Necklace	89.25%	33.15%	Substitutes
Long Floral Scarf	Chiffon Floral Scarf	83.44%	0.05%	Substitutes
World War II Pin	World War II Military Cap	96.11%	63.55%	Complements
Lord of the Rings Costume	Lord of the Rings Kit	84.33%	99.79%	Complements
Flat Head Pins	Tibetan Silver Earrings	22.97%	33.13%	Neither
Cotton Wedding Day Socks	Wedding Decoration Tissues	38.42%	24.01%	Neither
Glow in the Dark Decor	Brass Necklace	64.55%	89.65%	Neither
Gold Plated Watch	Golad Plated Watch	100%	100%	Substitutes

Table 3: *Sample product relationships captured by Graph Neural Network modeling.*

From the table above, we see that for items such as two variations on a men's necklace (a leather pendant necklace and a cross shield necklace), which should presumably be classified as substitutes, are predicted to share a view edge but not a buy edge. Another such example is if we were to examine two variations of a scarf (a long floral design as opposed to a chiffon floral design), the two are predicted to share a view edge but not a buy edge. For these pairs of items, the initial modeling assumption holds; customers usually peruse multiple variations of one type of product and then select whichever one they prefer to purchase.

Next, we examine predicted complementary relationships. The set of World War II pin and military cap are part of one hypothetical army-related costume, and are predicted to share both a view edge and a buy edge. Similarly, for a Lord of the Rings costume and an associated accessory kit, we see that two are predicted to share a view edge and a buy edge. Again, this correctly lines up with our intuition; customers view complementary goods around the same time and then purchase them all in one go.

Although the model can identify these broad categories of relationships, it is also useful to determine whether or not the models can also make negative determinations; if two products are neither substitutes nor complements, having little to nothing to do with each other, then the model should tell us that. For example, flat head pins and Tibetan silver earrings, items that share little to nothing in common, are predicted to neither have a view edge nor a buy edge. Similarly, cotton "wedding-day" branded socks and wedding decoration tissues, although sharing the wedding-day wording in their descriptions/titles, share nothing in common, and are predicted to share neither a view edge nor a buy edge. These sets of items have little to do with one another, and so it is fair to label them as neither substitutes nor complements.

Even though the model outputted roughly correct descriptions, some discrepancies were also notice-

able. To illustrate those, we turn to the last two examples. One of them is glow in the dark decor and a brass necklace, a pair of items that share little in common apart from the fact that a parent may have viewed and bought them together. However, the model outputs that these items are viewed and bought together, despite the underlying lack of a relationship between the two; it fails to deem this pair of items as neither substitutes nor complements. Another example, which is a lot more prominent in the dataset than the first, is the case of two very similar gold plated watches, with very few differences. The model predicts that the two, with very high certainty, share both a view edge and a buy edge, even though we would expect it to output that the two are viewed together but not bought together, as the modeling assumption suggests. The model fails to correctly label this pair of products.

The last example shows a common theme in the Amazon dataset; extremely similar items with small variations such as look, size, or material, are usually present in the dataset as having both Also-View and Also-Buy edges. Even though this doesn't align with the initial modeling assumption, it does highlight an important phenomenon: many users both view and buy different variations of a product *to try them out on themselves*, and depending on which they prefer, return one and keep the other. This is rather common behavior, which would make it rather difficult to distinguish between substitutes and complements along these lines. One avenue for future simulations could involve incorporating product return data, if available, to more concretely decipher this difference. Finding and incorporating this data was unfortunately beyond the scope of the current project.

On the whole, the link prediction modeling works, and even though it doesn't work perfectly for all pairs of items, it reasonably estimates the project's key modeling assumption. Incorporating such a model into business workflow is relatively simple – displaying pairs of items with high/low pairings of edge probabilities, depending on what the target relationship is, can help curate an initial list of candidates for such relationships. This reduces both the time and resource complexity burdens of finding pairs of substitutes or complements in massive product datasets.

6 Conclusion & Future Directions

This project offers a novel approach to understanding the relationships between products, focusing on the crucial distinctions of complementarity and substitution. By leveraging online purchase data from Amazon and employing link prediction algorithms with Graph Neural Networks, we have successfully demonstrated the ability to infer these relationships with reasonable accuracy. The incorporation of this model into business workflows offers practical benefits, simplifying the process of identifying potential substitute or complement pairs in large product datasets. By displaying pairs of items with high or low probabilities of being related, businesses can efficiently curate lists of candidates for further analysis, reducing both time and resource burdens.

While the model may not provide perfect predictions for all pairs of items, its ability to reasonably estimate the underlying relationships provides valuable insights for businesses. This includes informing pricing strategies, optimizing inventory management, and enhancing overall decision-making processes. Moving forward, further refinements to the model and continued exploration of alternative data sources could enhance its accuracy and applicability. Exploring data return data, using richer feature vectors for the nodes, and exploring additional edge types available in the Amazon dataset such as "Buy After Viewing" can result in a sharper and more accurate model. Nevertheless, this project represents a significant step towards leveraging graph techniques to better understand the complex dynamics of product relationships in modern markets at a large scale.

7 References

[1]. Tian, Y., Lautz, S., Wallis, A. O., & Lambiotte, R. (2021). *Extracting Complements and Substitutes from Sales Data: A Network Perspective.* EPJ Data Science, 10(1). https://doi.org/10.1140/epjds/s13688-021-00297-4.

[2]. Lattin, J. M., & McAlister, L. (1985). *Using a variety-seeking model to identify substitute and complementary relationships among competing products.* Journal of Marketing Research, 22(3), 330. Retrieved from http://search.proquest.com.ezp-prod1.hul.harvard.edu/scholarly-journals/using-variety-seeking-model-identify-substitute/docview/1297378495/se-2.

[3]. Goel, M. (2022, December 8). *Recommending Amazon Products Using Graph Neural Networks in PyTorch Geometric.* W&B. https://wandb.ai/manan-goel/gnn-recommender/reports/Recommending-Amazon-Products-using-Graph-Neural-Networks-in-PyTorch-Geometric–VmlldzozMTA3MzYw.

[4]. Hamilton, W. L. (2020). *Graph representation learning. Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3), 1–159. Morgan and Claypool.

[5]. Kipf, T. N., & Welling, M. (2016, September 9). *Semi-Supervised Classification with Graph Convolutional Networks (Version 1)* [Manuscript submitted for publication].

[6]. Hamilton, W. L., Ying, R., & Leskovec, J. (2017, June 7). *Inductive Representation Learning on Large Graphs (Version 1)* [Manuscript submitted for publication].

[7]. Rappaz, J., McAuley, J., & Aberer, K. (2021). *Recommendation on Live-Streaming Platforms: Dynamic Availability and Repeat Consumption.* In Proceedings of the 15th ACM Conference on Recommender Systems (RecSys).

8 Code Appendix

A notebook with all of the relevant code for the project can be found at the following link.