Amazon Toy Products Network Analysis

Aditya Garg
Stanford University
agarg94@stanford.edu

Sergey Ermolin
Stanford University
sermolin@stanford.edu

Sudhanshu Singh
Stanford University
ssingh9@stanford.edu

Abstract

Given the increasing user-privacy concerns, the usefulness of recommendation engines that learn from user-attributes or user-purchase history will soon come into question. We therefore proposed a recommendation engine for Amazon's products based on the network analysis of its Toy Product Dataset. Our approach leverages graph theory and network analysis as well as NLP processing of product name & product description entries. While the engine was developed for the Toy Products dataset, the dataset structure was sufficiently generic to be applicable to other Amazon product categories as well.

1. Introduction

Recommending similar products to consumers who showed propensity for purchasing items that fit certain description criteria and/or belong to a certain category has become a standard method for creating e-commerce recommendation engines. The body of work that formed the basis for recommendation engines in early 2000's used matrix factorization and collaborative filtering techniques. Recently, a new approach has emerged which leverages graph theory and network analysis whereby the event of customer 'c' buying an item 'i' was expressed as a probability of a network edge (c-i) being formed.

The novelty of our approach lies in the fact that we base our recommendation engine on the probability of item 'i' triggering the purchase of item 'ii' based on the probability of edge (i - ii) formation in a graph of item relationships rather than user behavior.

For our work, we were using the Amazon Toy Product dataset to create multiple graphs describing the relationship between items and propensity of the consumers to buy them in the future. We have also employed an NLP recommendation engine based on Jaccard & Cosine similarity to determine how closely related the dataset's items were. We applied this method to "product name" and "description" fields to build a graph of similar items. We then created graph edges between nodes whose similarity score exceeded a certain threshold. This edge generation method formed the basis of our recommendation engine.

2. Related Works

Most of the inspiration for this approach was drawn from McAuley et.al [1] and Shihui Song et.al [3] which described how to create a network from a list of Amazon product reviews, how to uncover its latent structures, and how to predict the probability of edge formation via regression [1] or clustering [3]. The information extracted from user reviews or product descriptions was used to determine what other products could be complementary to the one being reviewed as well as what products could be substituted. The class of recommendation engines that we chose to focus on in this project was based on the concept of representing individual Amazon products as nodes and then creating the edges between similar or related products if the probability of consumer's purchase is higher than a given threshold (eg. 92%). A recommendation engine was then used to suggest alternative products for a user during shopping process and once a purchase is complete, a list of complementary products for follow-on purchases.

The paper [2] helped us realize that if we use only product-related information, rather than user-related one in our recommendation engine, not only we skirt privacy-related issues, but we also avoid the "cold-start" problem. A "cold-start" term refers to the situation when a recommender system does not have enough information to match items characteristics to a relevant user's profile. This paper first showed a unique way to use the combination of item taxonomies (human annotated categories and subcategories) of the items and Latent Factors model to find a workaround for the problems mentioned above, and then also the use of temporal dynamics to make the model recommendation more precise. Due to the limited scope of the project, we did not have time to fully explore the other avenues for improving recommendation engine performance that were described in this paper, such as k-order Markov chains over purchasing sequences (eg. Photo Camera -> memory card -> tripod).

A survey paper [3] helped us understand the existing landscape of clustering algorithms used for recommendation engines. Upon reading it, we determined that the field of cluster-based recommendation engines seemed to be well-researched and decided against focusing our project on a cluster-based approach.

These concepts formed the foundation of our work and we expanded upon them, particularly by enhancing machine learning approaches using NLP and Jaccard similarities. While the idea of edge formation between related products (based on relationship probability being over a certain threshold) seemed to be universal among all the sources, most papers we reviewed used simple clustering or regression approaches to determine such probabilities. We are still not certain if this was due to the relative novelty of NLP approaches or due to the fact that traditional ML methods proved to be good enough for the task at hand.

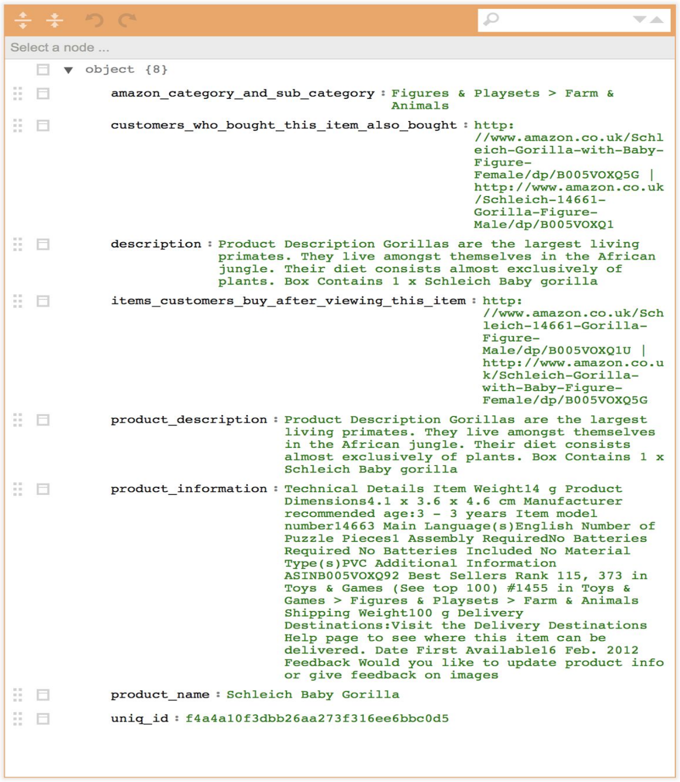
3. Data & Features

We purchased a full Amazon Toy Product dataset (220,843 records) which was published on Kaggle.com [4]. Figure 1 shows a sample is a dataset entry. The columns that we considered useful for our analysis are described below:

- **Product Description** - a text string, human-readable.
- **Product Reviews** - multiple string entries, human-readable. In the future, we will parse this field using an NLP approach to tease out information about item popularity to enhance our recommendation.
- **User-who-bought-this-item-also-bought** - URL strings, separated by " | ".

- **Category & Subcategory** - Amazon-generated, predefined multi-tiered entries
- **Items_customers_buy_after_viewing_this_item** - a field of the same format as "User-who-bought-this-item-also-bought".

Some categories contained duplicate information, such as "Product Description" and "Description". Others contained information not pertinent to our analysis, such as "number of available items in stock". These and similar fields were ignored in our analysis.



```
object {8}
    amazon_category_and_sub_category : Figures & Playsets > Farm &
                                       Animals
    customers_who_bought_this_item_also_bought : http:
                                                 //www.amazon.co.uk/Schl
                                                 eich-Gorilla-with-Baby-
                                                 Figure-
                                                 Female/dp/B005VOXQ5G |
                                                 http://www.amazon.co.uk
                                                 /Schleich-14661-
                                                 Gorilla-Figure-
                                                 Male/dp/B005VOXQ1
    description : Product Description Gorillas are the largest living
                  primates. They live amongst themselves in the African
                  jungle. Their diet consists almost exclusively of
                  plants. Box Contains 1 x Schleich Baby gorilla
    items_customers_buy_after_viewing_this_item : http:
                                                  //www.amazon.co.uk/Sch
                                                  leich-14661-Gorilla-
                                                  Figure-
                                                  Male/dp/B005VOXQ1U |
                                                  http://www.amazon.co.u
                                                  k/Schleich-Gorilla-
                                                  with-Baby-Figure-
                                                  Female/dp/B005VOXQ5G
    product_description : Product Description Gorillas are the largest
                          living primates. They live amongst themselves
                          in the African jungle. Their diet consists
                          almost exclusively of plants. Box Contains 1 x
                          Schleich Baby gorilla
    product_information : Technical Details Item Weight14 g Product
                          Dimensions4.1 x 3.6 x 4.6 cm Manufacturer
                          recommended age:3 - 3 years Item model
                          number14663 Main Language(s)English Number of
                          Puzzle Pieces1 Assembly RequiredNo Batteries
                          Required No Batteries Included No Material
                          Type(s)PVC Additional Information
                          ASINB005VOXQ92 Best Sellers Rank 115, 373 in
                          Toys & Games (See top 100) #1455 in Toys &
                          Games > Figures & Playsets > Farm & Animals
                          Shipping Weight100 g Delivery
                          Destinations:Visit the Delivery Destinations
                          Help page to see where this item can be
                          delivered. Date First Available16 Feb. 2012
                          Feedback Would you like to update product info
                          or give feedback on images
    product_name : Schleich Baby Gorilla
    uniq_id : f4a4a10f3dbb26aa273f316ee6bbc0d5
```

Figure 1: Dataset Example

4. Models/Algorithms/Methods

4.1. Ground Truth Analysis

We first created a graph based on "users-who-bought-this-also-bought-that", "Items_customers_buy_after_viewing_this_item", and "amazon_product_category_and_sub_category" fields.

We looked at the number of nodes with missing attributes and calculated the size and ratio of the largest Weakly Connected Component (WCC) and largest Strongly Connected Component (SCC). We ran other extensive network visualization experiments (see below) to better understand the nature of our dataset. This step, while time-consuming, proved to be very valuable as it allowed us to quickly test and discard several initial approaches, such as attempts to augment the baseline recommendation engine described below (4.2).

Analysing the ratios of disjoint nodes vs WCC nodes allowed us to justify discarding disjoint parts of the graph as outliers. We also identified a lot of items in the "users-who-bought-this-also-bought-that", "Item_customers_buy_after_viewing_this_item" attributes for which other data like product name and description was not available and discarded those. This approach helped us account for these irregularities in our final numerical analysis which otherwise would have led to highly erroneous results.

Since the dataset was large, we performed an extensive data pre-processing to identify outliers in 'product_name' and 'product_description" fields and fixed/removed them before feeding the data to the Jaccard/Cosine similarity NLP engine.

4.2. Baseline: User-Rating Recommender Engine

We also created a baseline recommendation engine where we selected the highest-rated item(s) in each subcategory to be the universally recommended product for anyone who landed on an Amazon webpage that belonged to the same category and sub-category.

The intuition was to recommend the most popular item in a category as an obvious choice provided no user data was available to us. When compared to Amazon's own records of "User-who-bought-this-item-also-bought", however, this engine performed rather poorly - see 'Results/Discussion' section below.

4.3. NLP-Based Engine

The final recommendation engine was based on using both Jaccard and Cosine similarities between Product_Description and Product_Name fields.

The product name and description were converted into word vectors using special tokenization regexes that accounted for space, tab, newline etc. These vectors then underwent extensive processing removing outliers like random characters, unicode symbols etc. The numbers in text were processed into single element in a vector and then the term frequencies were collected for the cosine metric. Below are the key formulas used in the analysis:

- Jaccard $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$
- Cosine $\frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$

Another key challenge was the compute time required to process 220,843 items, permuting each one with another resulting in over 48.7 billion permutations. This required extensive parallelization of code and still took over 8 hours to run Jaccard similarity and over 38 hours to run Cosine similarity on a 4-node, 40-core/node compute cluster. Rather than adapting our code for Apache Spark environment, we wrote our own custom code using Python's multi-thread, multi-processing methods.

The product recommendation was made whenever the similarity score was above a certain threshold. We run the analysis for multiple threshold and picked one that gave the best balance between prediction accuracy and compute time. See Results/Discussion section below.

5. Results/Discussion

Understanding the dataset was critical to find the right feature vectors and remove outliers. We identified 115377 unique items and here are some other interesting findings.

5.1. Dataset Category & Subcategory Analysis:

We created a directed graph with an edge from each category to each subcategory (Fig. 2). This graph's analysis revealed the following characteristics:

- Items with no categories: 13719
- Number of unique Categories & Subcategories: 986
- Number of disjointed graphs: 51
- Size of Largest Weakly Connected Component: 649 nodes

5.2. Dataset Analysis of Items bought in groups:

Graph whose edges connect item's URL with URLs from "User-who-bought-this-item-also-bought-this" attribute (Figure 3). This directed graph shows

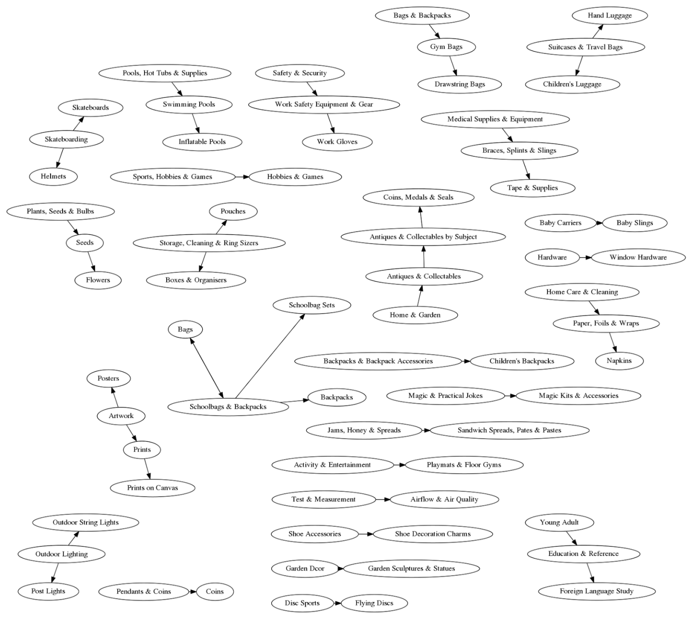


Figure 2: Example: Product Category & Sub-Category

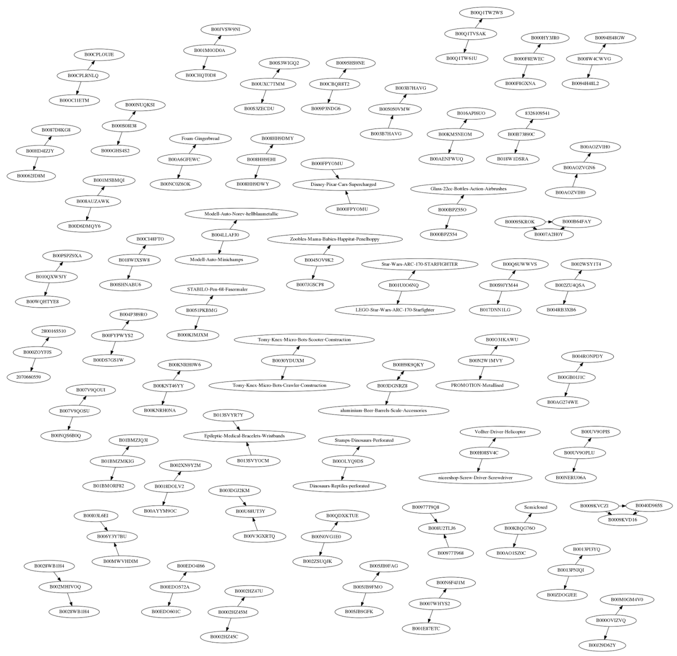


Figure 3: Example:People Bought X And Also Bought Y

future buying propensity triggered by a purchase of a single item. Here is the information on the full graph:

- No. of Items with "User-who-bought-this-item-also-bought-this" attribute: 11408
- Number of Nodes in Graph: 220,843
- Number of Edges in Graph: 546,001
- Number of Reciprocated Edges: 25,458
- Size of largest WCC: 209,411
- Size of largest SCC: 2,237
- Number of disjointed Graphs: 2,754

Examination of the graph data for items bought in groups led us to the following conclusions:

- Less than 5% of the users bought just a single item, so there is a definite "return buyer" behavior present in the dataset.
- Number of disjoint graphs was very small compared to all the graphs. These probably represent an "outlier" behavior and was eliminated from further analysis.c
- We expected more reciprocated edges: if an item A led to a purchase of Item B, one would assume the reverse should be true, but that does not seem to be the case.
- Surprisingly, there was a sizable SCC set. In the future analysis, it deserves a very careful examination since it may represent a set of highly popular items that could be recommended to almost any user.
- The deepest WCC set has 95% of all graph's nodes, pointing to a well-connected network structure.

5.3. Analysis of Items Leading to Other Purchases:

We also considered the graph formed by edges between items and "Item_customers_buy_after_viewing_this_item" (Figure 4). Graph's statistics was as follows:

- No. of Items with no 'Items_customers_buy_after_viewing_this_item' attribute: 33199
- Number of Nodes in Graph: 139320
- Number of Edges in Graph: 281178
- Size of largest WCC: 125265
- Size of largest SCC: 44
- Number of disjointed Graphs: 3411

This graph also has a WCC of almost the same size as the graph itself, but its SCC is very-very small. It would be interesting to see if there was an intersection between this SCC and the one in Section 5.1. If so, it may represent some insanely popular items that could be blindly recommended to anyone visiting a landing page for high-level Toy categories.

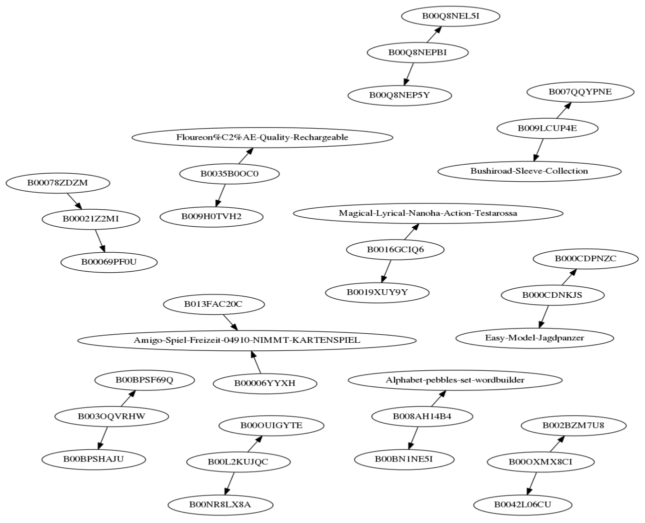


Figure 4: Example: People Viewed X But Bought Y

5.4. Validation Criteria

We treated the "User-who-bought-this-item-also-bought-this" as the ground truth since these were representations of the Amazon recommendations. We created a graph with nodes representing all the products and edges representing amazon recommending an item based on their own recommendation engine as given by the above attribute. Using the above graph, we looked at the list all item pairs along with the similarity confidence as predicted by the different approaches.

We then looked at the node overlap, which is the number of similar nodes above a given threshold that were actually a part of the original graph. Since our recommendation engine had no knowledge of the ground truth attribute, it missed many nodes thus making this an important testing parameter.

We then looked at the edge similarity. Here, we defined a parameter 'distance'. If the shortest path between two nodes was less than a predefined distance threshold and these nodes were recommended by the engine, we considered them to be a valid recommendation. As the distance increased, the percentage of matche predictions did increase as well. This behavior stemmed from the idea that if A is similar to B and B is similar to C, then A is similar to C to a certain extent.

5.5. Baseline - Recommendation Engine Based on User Ratings

This engine performed terribly on the baseline using the average user rating. The engine had 0.3% accuracy which also required a fair bit of fine tuning. To determine its accuracy, we compared the URLs it predicted with Amazon's own records contained in "User-who-bought-this-item-also-bought" field of the dataset. While we were initially dismayed by such a low accuracy, we eventually realized that without knowing how many users reviewed or bought a specific item, high review rating was not very meaningful. Still, since this was a very simple and easy-to-implement engine, we kept it as yardstick for our efforts.

5.6. Jaccard Text Similarity Analysis

Combining Product Name and Description allowed us to run reduce compute time and improve accuracy. We saw the node overlap start at 50% and drop down to almost 10% as we increased the similarity threshold from 30% to 100%. This was as expected, but some more cleaning of the dataset could result in higher overlap. See Figure 5

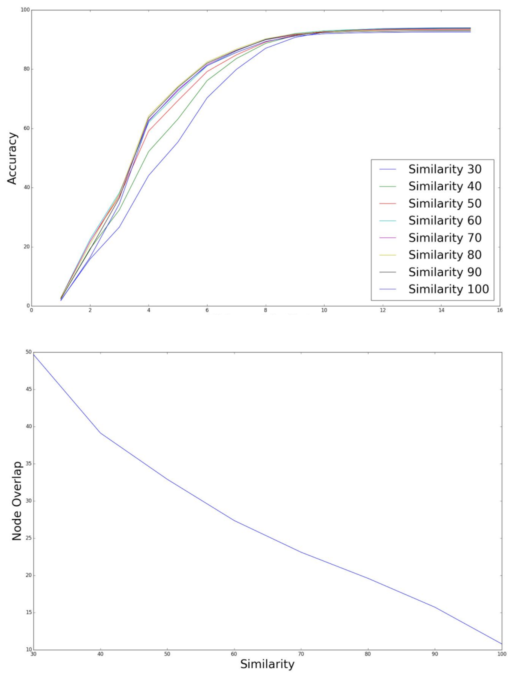


Figure 5: Node Overlap and Edge Probability using Jaccard Similarity

Moreover, the edge overlap was really good as most predictions were within 4 to 6 edges from each other in the ground truth graph.

5.7. Cosine Text Similarity Analysis

For cosine similarity, the results were not as good: the overlap was lower for all thresholds starting at 30% and dropping to 5%. On the contrary, the edge overlap turned out to be better since it almost hit 96% for almost all thresholds. If we can improve the node overlap, cosine text similarity metric could prove to be a valuable method for product similarity analysis (Fig. 6).

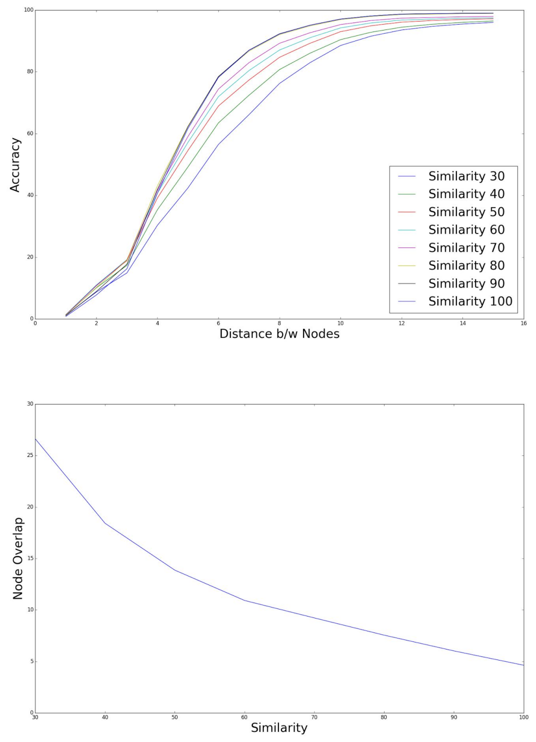


Figure 6: Node Overlap and Edge Probability using Cosine Similarity

5.8. Degree Distribution Analysis (Fig. 7)

The GroundTruth blue graph ("Bought X Also Bought Y") has a spike in node distribution around node_degree=10 which we can't explain - it was just present in the data.

The reason why the Jaccard-based prediction graph (green) started off lower than the GroundTruth graph (blue) was due to the fact that we picked a fairly high edge formation threshold of 75%, resulting in fewer edges being formed. When we reduced the text similarity threshold down to 50% and below (not shown here), the green graph "moved" higher.

The reason why the Jaccard based prediction graph (green) showed more nodes with degree above 100 was due to Amazon's product nomenclature. For example, there were literally hundreds of Star War board-game puzzles with different number of pieces. When we run Jaccard similarity on these with 75% threshold, they all got flagged as "similar" and created a cluster with high node degrees.

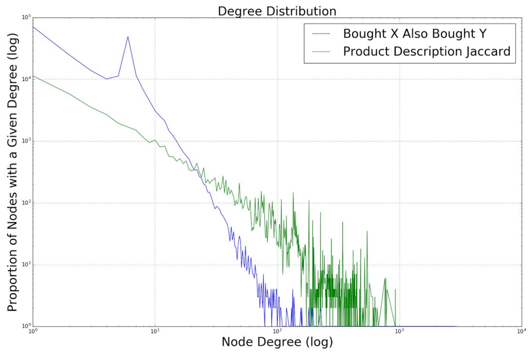


Figure 7: Degree Distribution Comparison

6. Conclusion

We did an extensive analysis of Amazon Toy Product dataset to reveal the relationship between different categories and purchasing trends. We also built a very simple (albeit low-performance) consumer ratings-based recommendation engine that served as our baseline. We employed Jaccard and Cosine similarity metrics to select similar products based on their names and descriptions. Furthermore, we experimented with various similarity thresholds to establish the optimal one for our engine. We mapped the similarity metrics into user purchasing propensity scores and used those to form a product-2-product graph edges in our model. In the end, using the above tools, we were able to predict how likely a new item in the dataset was to be purchased based solely on its name and its description. Our recommendation engine therefore had neither a "cold start" problem, nor was it exposed to the concerns related to user-privacy data. This suggests that with some more advanced approaches, the product attributes could be a viable source of recommendation. We also experienced first-hand the advantage of Jaccard similarity vs Cosine similarity when it came to compute time (without compromising the accuracy of the results)

7. Future Work

To take this work further, we could run the similarity analysis on customer reviews (in addition to product name and description) to further refine the correlation between different items. This approach has the potential of improving the quality of our engine, provided we would be able to limit the scope of recommendations to only those with high-ranking reviews without significantly reducing the size of our dataset.

For the text similarity calculation, we could experiment word embeddings from Word2Vec model and Convolutional Neural Networks (CNN).

As mentioned in the Introduction, we would like to compare the performance of our engine using different ML methods to determine the probabilities of edge formation.

8. References.

[1] McAuley, J., Pandey, R. and Leskovec, J. 2015. Inferring Networks of Substitutable and Complementary Products. Proceedings of the 21th ACM SIGKDD International.

[2] K. bhargav*, Amrahmed*, Spandey, Vanjaj, Yuanjef, lluis@yahoo-inc.com. Supercharging Recommender Systems using Taxonomies for Learning User Purchase Behavior.

[3] Shihui Song, Jason Zhao. Survey of Graph Clustering Algorithms Using Amazon Reviews.

[4] Toy Products on Amazon | Kaggle. Web. 19 Oct. 2017. <https://www.kaggle.com/PromptCloudHQ/toy-products-on-amazon>.

[5] "Amazon Product Co-purchasing Network, March 02 2003." SNAP: Network Datasets: Amazon Co-purchasing Network. N.p., n.d. Web. 19 Oct. 2017. <http://snap.stanford.edu/data/amazon0302.html>.

[6] Srivastava. 2010. Motif Analysis in the Amazon Product Co-Purchasing Network.

[7] P. Melville and V. Sindhwani, "Recommender Systems," in Encyclopedia of Machine Learning Claude Sammut and Geoffrey Webb (Eds): Springer, 2010.

[8] J. Leskovec, L. Adamic and B. Adamic. "The Dynamics of Viral Marketing" in ACM Transactions on the Web (ACM TWEB), 1(1), 2007.

[9] Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. J. McAuley, J. Leskovec. ACM Conference on Recommender Systems (RecSys), 2013.

[10] Almazro et. al. "A Survey Paper On Recommender Systems". https://arxiv.org/abs/1006.5278