Addisalem Kebede

Big Data – CptS 415

Amazon Co-Purchase Metadata Analysis

**Introduction:**

Big data applications are becoming more and more popular all over the world, not only in commercial business and professional manufacturing but also in daily activities. Not outside this trend, Amazon is a leading supplier of evolving data applications some that even have predictive factors that can help to improve sales and profitability. Through data analysis of online time, clicks, items in shopping carts, or reviews from customers Amazon has a ton of information at its fingertips. For decades, Amazon and its developers have been working to continuously develop ways to visualize data in order to impact its decision making. This development has contributed significantly to the success of this technology giant. It can be said that consumers currently face too many choices when buying goods due to the large and diverse volume of products. The question now is how can they use data to continue to grow by helping customers figure out what they want? Increasing sales always comes with data research to make suggestions and decisions that best suit the data collected from customers. This project uses the 2006 Amazon co purchasing network metadata in order to examine how data can be used to impact decisions. In the first part of the project I look at how to use Amazon’s DynamoDB to store data and perform queries in order to visualize that data. The second part will look at how to use the data in order to make product suggestions based on the products that were co-purchased with one another and the similarity of their categories.

**Model/Algorithm/Method:**

**Dataset:**

The dataset was composed of 548,552 products that were either Books, DVDs, Music CDs, or Videos. These products also held information on reviews that were done on that product. Overall, there were 7,781,990 reviews that were part of the dataset. The products that were considered for this project were all products that were not discontinued.

A screenshot of a cell phone

Description automatically generated

**Figure 1: Data Statistics**

**NoSQL Database:**

The first portion of the project was written in python along with Amazons DynamoDB. DynamoDB is a NoSQL database that allows for up to 10 trillion requests per day and 20 million requests per second. While this is excessive for this project in real world applications it can be crucial. A product table and a review table were created on the database in order to provide some structure to the data (Figure 2).

A screenshot of a cell phone

Description automatically generated

**Figure 2: Product and Review Tables**

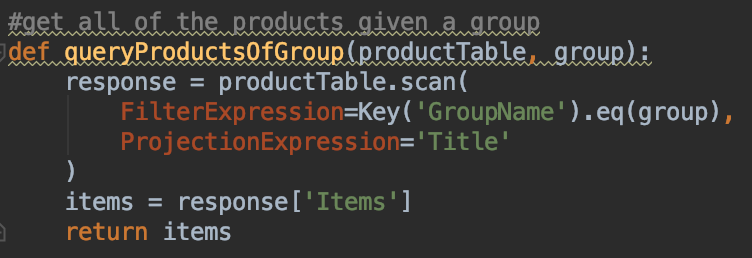
In order to get the dataset from its original state (Figure 3) into a form that could be inserted into these tables, it needed to be parsed and relevant data needed to be extracted from it. After data was extracted then it could be inserted into each of the tables.

A close up of text on a black background

Description automatically generated

**Figure 3: Original Data Form**

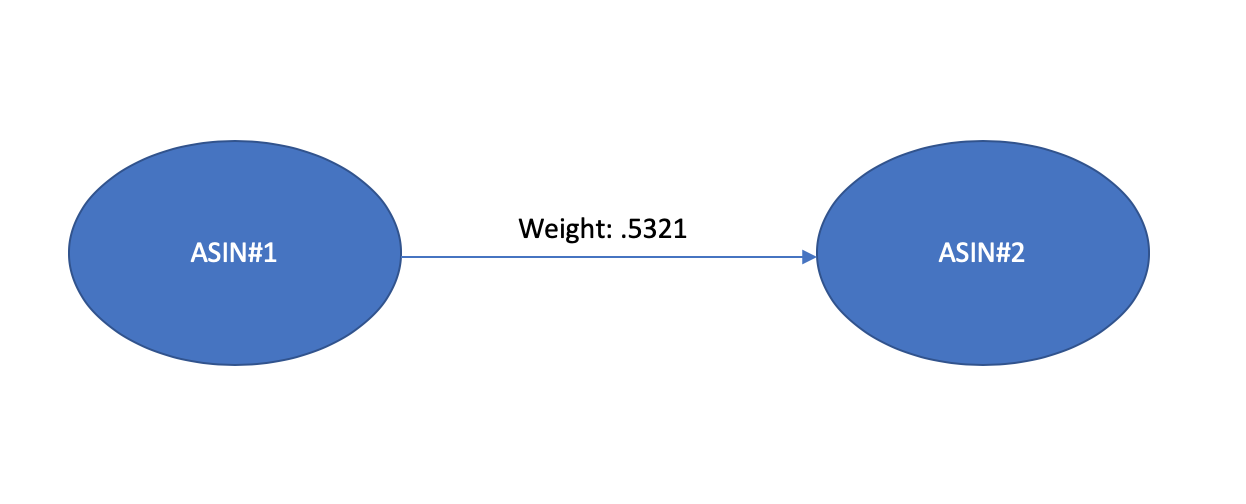
Once data was inserted into the database, the next step was querying it. DynamoDB doesn’t use SQL syntax for its queries but can perform equivalent queries using the query/scan functions (Figure 4) in the boto3 package. In this project 12 queries have been pre-defined for users to extract things from the database. These queries on the data go from getting all of the reviews of a given product to finding all of the reviews a given customer has made. After the queried data is brought down then it is displayed to the user.



**Figure 4: DynamoDB scan function for products of a given group**

**Recommendation System:**

The second part of this project was to build a recommendation system that would recommend products to a user based on the current product the user is looking at. Products were put into a graph where nodes were that products ASIN number and the edges were to products that were co purchased with this product. Each edge of the graph had a weight imposed on it that represented the similarity of the two products it connected (Figure 5).



**Figure 5: Relationship between two product nodes in graph**

In order to calculate the similarity (Figure 6) of two different products, the cosine similarity of their categories is computed. This calculation requires a combination of each products category set as integer values and then a dot product divided by the product of the magnitudes of each category set relative to the combined set. This similarity is what is used when producing the list of recommended products to a user provided a product. The recommendations for products were made based on ordering the co purchased products by the weight of the edges that linked them.

A screenshot of a cell phone

Description automatically generated

**Figure 6: Cosine Similarity Computation in python**

Due to the fact that all of the products that can exist in the graph need to have products that have been purchased with them the data needs to be cleansed of all products that have a similar list with no other products. This reduces the dataset to a smaller yet, still very large set.

**Results and Learnings:**

From this project I was able to create a database in Amazons DynamoDB database, insert data into that database, and then query data back while being able to visualize it in different ways.

A screenshot of a cell phone

Description automatically generated

**Figure 7: All possible preset queries**

I was also able to produce recommendations for nodes that had co purchased products with them. The products that were produced were an ordering based on category similarity of the co purchased products.

**References:**

“Amazon Product Co-Purchasing Network Metadata.” *SNAP*, http://snap.stanford.edu/data/amazon-meta.html.

K, Ben. “Book Recommendation (Python).” *Medium*, Medium, 11 June 2018, https://medium.com/@baemaek/amazon-book-recommendation-system-analysis-d9d72b9a7173.

Rangel, Derek. “DynamoDB: Everything You Need to Know about Amazon Web Service's NoSQL Database.” *Amazon*, Derek Rangel, 2015, https://aws.amazon.com/dynamodb/.