Analysis of Trends in Amazon Product Popularity

CMSC 12300 – Nick McDonnell, Vinay Reddy, Sasha Trubetskoy — link to repository

Dataset

Given the database of amazon product reviews given thanks to Julian McAuley we were able to access 1.2 GB of Amazon product clothing reviews from May 1996 - July 2014. This amounted to 5,748,920 Reviews ratings. Our dataset also included metadata, which includes descriptions, sales-rank, price, brand info, and co-purchasing links.

Hypotheses

Our general hypothesis was whether or not certain seemingly unrelated pairs of products would demonstrate similar changes in popularity over time. Our goal was to discover pairs of products that were both well liked at the same time, and yet not explicitly related.

We tested several hypotheses of comparing datasets. The first, *inter-fad* would take pairs of products and yield a similarity if the net distance between the two had the smallest distance between them.

Another hypothesis we tested is *intra-fad*. In this case, we took a different approach to find products with similar popularity curves, where we would take normalized vectors of sets of points of the two graphs and generate pairwise distances to compare products.

Through these methods of graph comparison, we hope to generate two separate analyses of amazon products that will show similar and different results. These results will allow us to interpret the success and of our hypotheses and relate the two different ways of relating the data.

Algorithms and big data approaches

Data preparation

Our approach required binning the ratings by month to create a vector for each product, where each component represents the number of ratings for that month. We divided each component of the vector by the total for that month so that each product vector represented how popular the product was in relation to all the products sold that month. This was to account for the fact that Amazon purchases dramatically increased each year from 2007 to 2014.

For both algorithms, we defined distance to be the sum of the absolute values of the differences between each i-th component of the two vectors:

$$distance(a, b) = \sum |a_i - b_i|$$

This is essentially the Manhattan distance between the two vectors. (Why did we use Manhattan and not Euclidean distance?)
(define "normalized vector" i.e. normed by component sum)

Algorithm 1: Interfad

Hypothesis 1 required us to pick a product and test every other product against it for distance. Testing every pair of products and simply finding pairs with the smallest distance would result in a bias towards products with small numbers of reviews. This is because of the way we chose to measure distance, which uses subtraction and is vulnerable to scale effects. Thus, this algorithm makes more sense when we start by first selecting products with a desirable number of reviews, and then testing all other pairs against the chosen products. Given product A, the algorithm outputs product B, that satisfies:

$$min\{distance(A, B_i)\} \quad \forall i$$

Algorithm 2: Intrafad

Hypothesis 2 looks for vectors with the same shape, same timing, but different component sums. In order to test this, we simply iterate through the list of pairs and compute the distance between the two normalized vectors. This happens in the mapper, which gives us a list of pairs for which the distance falls below a certain arbitrary threshold. The reducer step uses a leader product as the key, and neighbor products as values, joining on the key. The result is a list of neighbors for each product—essentially a cluster.

This is not a good list of clusters, however, since it contains many redundancies. We use a simple method of removing these:

1. **Sort neighbor list by number of neighbors (descending)**
2. **Go down this list,**
    a. **adding the first value to the final cluster list,**
    b. **then adding its neighbors to a skip list and skipping over them;**
    c. **and continuing in this manner.**

This ensures that the clusters that we get are distinct enough to be interesting. While this is not a perfect way to get every cluster, it gave us a fairly large amount of interesting clusters to look at. We can implement this without MapReduce because we are dealing with a relatively small number of products, rather than many pairs of products.

Big data approaches

As mentioned previously, our primary big data method was using MapReduce, implemented in python with the Mrjob package. With Google Cloud services giving us access to many computing nodes, this approach would greatly reduce the time required to perform the pairwise calculations that are necessary for our algorithms.

Challenges faced

Algorithms

In order to compare products we noticed we needed nested for loops in mrjob, but once we realized this was not possible we generated a list of pairs in order to run the comparisons we wanted.

In order to correctly compare the individual datasets for each product over time, we realized that the increase in popularity in amazon would skew our data analysis. This lead us to normalize the data based on the increase so that it would not interfere with our results.

Although there are a variety of algorithms in order to compare the shape of two graphs, we chose the algorithms discussed above over various other comparison techniques. (No cut and dry method to compare the graphs) This would impact the accuracy of our comparison model in regards to providing a precise comparison between two products.

Google Cloud

We faced a variety of issues while trying to get Google Cloud to work. Due to laggy VM's, we tried to SSH into a VM instance, but ran into permission errors. Then, resorting to the VM, our code seemed to run nearly to fruition, but would frustratingly throw an error right before completing the final mrjob step, and the error logs proved indecipherable. Given a couple more days, we would be able to sort out the technical issues with the Google Cloud setup, but as of now, we unfortunately cannot provide the results from running our code on the Google Cloud clusters.

Results

Intrafad Results

After running several small-scale trials, we decided to set the distance threshold at 0.2. This means that if two vectors have a distance of less than 0.2, they are considered "connected". Thresholds above 0.2 produce many clusters, but upon casual inspection they have a lesser significance. Thresholds below 0.2, meanwhile, produce too few clusters (and clusters too small) to be interesting. The chosen threshold gives a good balance between reducing spurious correlation while maintaining a large number of clusters returned.

The pairs.txt file contains 124,669,945 pairs, which would take 31 days to run on a late 2013 MacBook Pro (dual core 2.6 GHz). This is prohibitively expensive, even with MapReduce on a cluster. We attempted to run half of this dataset, in order to more realistically expect being able to complete it within a few hours. Due to the challenges mentioned above, we did not receive a full result for the large-scale implementation.

On a smaller-scale implementation, which we ran on personal machines, the algorithm gave us some interesting results. The product clusters were a combination of seemingly spurious products and genuine fashion or seasonal trends.

Some spurious-seeming clusters looked like this:

```
[8 Of Hearts Waist Firm Compression Waist Cincher,
Southpole Mens Belted Ripstop Basic Cargo Short,
Burlesque Classical Satin Plus Size Corset,
Unisex Hip Hop Side Zipup Extra Long Tshirt,
Skull Simple Band Mid Finger Rings Set,
Samsung Gear 2 Smartwatch Metallic Orange,
Samsung Gear 2 Neo Smartwatch Orange]
```

Certain groups of items within the cluster appear to be related, but it is hard to associate orange smartwatches with plus-size bustiers. On the other hand, many clusters appeared rather coherent, such as these:

```
[Skechers Womens Go Walk 2 Super Sock Walking Shoe,
Women Short Sleeve Chiffon Top,
Womens Printed Summer Tropical Multicolor Maxi Dress,
4 Set Packing Cubes Travel Organizers with Laundry Bag,
Mens Sandals By Peshantis]

[Mens Thermal Underwear Set Topamp Bottom Fleece-Lined,
Soongquot Womens Soft Fur Eskimo Boots,
Womens Pullover Jumper Hoodie,
Hanes Mens X Temp Thermal Pant,
Women Warm Knit Long Scarf,
Boots Over The Knee Faux Suede Leather Shoes]
```

The first one represents a real fashion trend—according to Google Trends, maxi dresses peaked in spring of 2014, as did chiffon tops and Sketchers Go Walk shoes. The second represents more generic items that have highly seasonal demand across all years.

These results came from only a small subset of the data, and this algorithm is easily scaled to encompass all Amazon clothing items and other products.

Interfad Results

The pairs_200.txt file contains 3,152,400 pairs. We constructed the pairs by taking the top 200 products by number of ratings and matching them with all other products. We chose 200 arbitrarily because since the algorithm requires us to first select products, we believed choosing popular products would yield a more interesting comparison. Based on small-scale trials, it would take approximately 21 hours to run run on a late 2013 MacBook Pro (dual core 2.6 GHz). Since we were unable to run it on Google Cloud, we ran the algorithm with 20000 pairs which took 8 minutes on our personal machines.

Here are two examples of results that we got when we calculated the most similar popularity profiles for some popular products:

```
["High Sierra Access Backpack"]:
[["Carhartt Mens Long Sleeve Workwear Henley Shirt"],
["Soft Style Womens Angel II Pump"],
["Clarks Mens Escalade Slip On"],
["Flying Fisherman Maverick Polarized Sunglasses"],
["Seiko Mens SKA347 Kinetic Silver Tone Watch"]]
```

This first example is pretty straightforward. With the exception of the women's pump shoe, this set of products exemplifies a "rugged sophisticated" male look that took hold in late 2013, as corroborated by Google Trends.

The second example:

```
["Bridesmaid Prom Holiday Formal Long Dress Junior Plus-Size"]:
[["Nine West Womens Shiza Knee High Boot"],
["The Dragons Lure Stud Gothic Earring Right Ear"],
["The Walking Dead I Heart Daryl Juniors TShirt"],
["SODIAL T MMulti Vintage Colorful Crystal Peacock Bracelet Bangle"],
["Classic Dragon Ear Wrap Cuff Earring Stud"]]
```

This shows similar profiles to a popular plus-size prom dress. The popularity of the dress corresponds with the popularity of boots and earring accessories. Furthermore, it is not unreasonable to make the connection between young (likely teenage) girls buying prom dresses and t-shirts featuring Daryl, a young male character from the show *Walking Dead*.

We see that this method is good for identifying and quantifying trends in complementary and substitute goods of similar volume.