Best Buy Recommendation System

Team: BestBuyIn

Members: Ayushi Gaur, Ganapathy Subramanian, Sahana Bhat, Varun Kandukuri

1. Introduction

Recommender systems primarily use two ways to provide meaningful recommendations. Namely, content-based filtering and collaborative filtering. In this report, we have used a content-based filtering to recommend products to customers using data provided by Best Buy on Xbox games.

1.1 Motivation

In the wake of the 2020 COVID-19 pandemic, many local retail stores have had to shut their business down. While information technology has transformed the way people make purchases online, not all physical stores have moved to this format. With fierce competition from online e-commerce giants like Amazon.com, retail stores can be left in an especially vulnerable position during times of crisis. Studies show that for retail stores, a large number of low-volume sales of niche products form a major chunk of their revenues (Anderson 2006; Brynjolfsson et al. 2006). However, it can be expensive to keep a large variety of products in stock, especially so when business is slow.

We wanted to explore how best the stores can direct their stock-keeping efforts for their local consumers in such situations. Using sparsely available consumer data, stores can generate recommendations and make inventory decisions on this basis.

The data we use for our research contains consumer purchase behaviour for Xbox Games. Strictly speaking, games may not be deemed as essential products in times of crisis, but the method of recommendation can be reapplied to other categories of products.

1.2 Research Question

Can we accurately recommend top k products to a user based on their previous search queries?

- Each observation contained information on what a user searched for and what they subsequently clicked on, accompanied by the times at which these events took places. For our secondary question, we wanted to analyze the difference between click time and query time based on the search query and the product that was clicked.

1.3 Previous Work

The primary goal of an e-commerce business is to drive their profits, and the way they achieve this is by maximizing sales on their platform. At the fundamental level, a sale is most likely to go through if a customer is given exactly what they want, and this is where recommender systems come in. Although there has been a lot of prior work in this field, two papers in particular had piqued our interests.

1.3.1 BestBuy Recommendation System - Collaborative Filtering

The first one is a recommendation system for Best Buy based on collaborative filtering. In this paper, they have implemented four different different methods to identify the users with similar search patterns of the target user and

recommend products based on the search patterns of the other users:

1. Query based collaborative filtering.
2. Item to item collaborative filtering.
3. Query based clustering.
4. Product based clustering. (Rajendra, N et al., 2012).

Although we used the very same dataset, our approach differs from that of Rajendra's.

As you will see in our exploratory analysis, our findings around user behaviour prompted us to use content-based filtering for more practical results.

1.3.2 Movie Recommendation System using Content based Collaborative Filtering

The second one is a movie recommendation system on the Movielens dataset using a combination of content based filtering and collaborative filtering where they use the content based filtering to recommend the movies based on the users past watching history. In their proposed algorithm, firstly they use a set matching comparator - a content based filter, in which the users are grouped by the tags and genres of the movies watched. Secondly, they identify the cluster in which the target user falls and then recommend movies to them from the cluster which is essentially a collaborative filtering (Pal, A et al., 2017).

1.4 Plan

The work of this research was distributed equally among all the members of the team. However, certain tasks were handled and contributed towards exceptionally well by some members of the teams which we would like to highlight below:

1. Literature Review: Ganapathy
2. Exploratory Data Analysis: Sahana
3. Data extraction and cleaning: Ayushi
4. Recommendation System algorithm: Varun

2. Data Source and Description

The primary dataset was taken from a Kaggle Competition (Data Mining Hackathon on (20MB) Best Buy mobile web site - ACM SF Bay Area Chapter).

2.1 Train & Test data

The main dataset being **Train.csv** is the training dataset consisting of ~42k records of user search queries and their click times for category of Xbox games along with their sku codes which are the product IDs of products clicked on. The train dataset contains the following fields:

1. **user**: the customer ID
2. **sku**: the stock-keeping unit that the user clicked on which is a unique ID for each product
3. **category**: the category the products belong to
4. **query**: The search terms that the user entered
5. **click_time**: Time the sku was clicked on
6. **query_time**: Time the query was run

The **test.csv** contains all the same fields as train.csv except for sku.

2.2 Products

As part of the original dataset, we were given a **small_product_data.xml** file which had a lot of details for each product. For our initial data analysis of the product catalogue, we first had to extract the data from XML into a dataframe and then finalize on the features useful for this research out of a total of 106 feature lists. For

436 products in our catalogue, we finalized with the following five attributes:

1. **Genre:** There were 16 unique genres available for games like board games, action, race, sport, etc. which we converted to dummies.
2. **Manufacturer:** The organization that developed and launched the game. There are a total of 56 unique manufacturers which we converted to dummies.
3. **ESRB rating:** Age and content rating given to the game by the Entertainment Software Rating Board and the 5 unique values for this were converted to dummies.
4. **ESRB descriptor:** It is the additional text description given by the ESRB for the game. It involved descriptive words like cartoon, mild violence, blood, drug and alcohol reference, strong language, etc. These texts were vectorized.
5. **Short and long description:** The short description had a maximum length of 90 characters and the long description has extensive synopsis of the game. These were clubbed together, cleaned and stemmed and then vectorized.

3. Analysis

In our preliminary analysis, we explored the following variables of interest that are required for building our recommendation system.

Popular search queries: The query terms used to search for the interested products are plotted into a word cloud as shown below in Fig 3.a. The size of the words and intensity of the color are based on the number of times these words have been searched for. Gears of War 3 and Dead Island are the most searched terms.

Fig 3.a Word Cloud of Search Queries

Popularity of the products clicked on: The top 15 most popular products based on number of clicks are as shown below in Fig 3.b. The colors of the bar represent the respective manufacturers of the game. From the bar chart, we see that Gears of War 3 is the most popular Xbox game and Microsoft is the leading manufacturer of the most popular Xbox games.

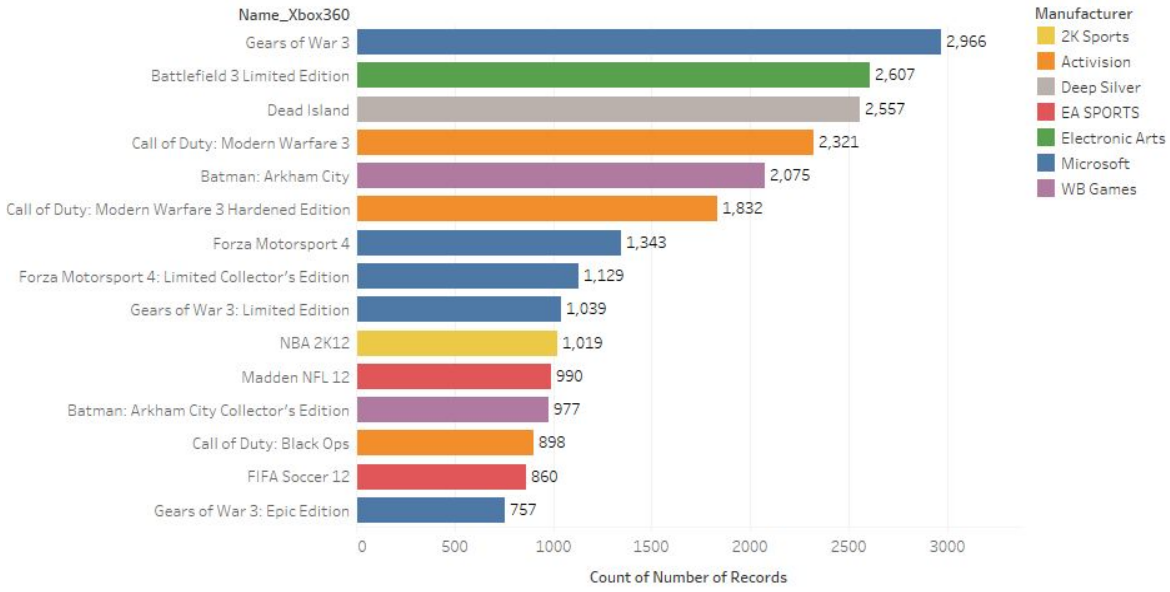

Fig 3.b Top 15 popular products clicked on

Classifying games based on ESRB Rating: The ESRB rating refers to the rating assigned by the Entertainment Software Rating Board. The visualization represents games being classified based on their ESRB Ratings as in Fig 3.c. We see that most of the games are under the category of Mature. Games for Everyone and Teens are almost the same in number.

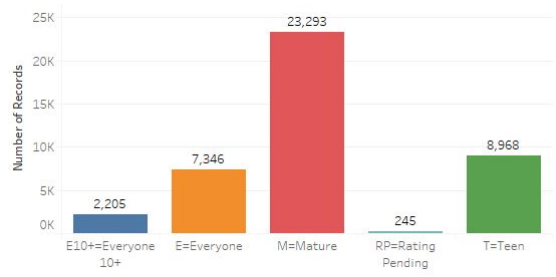

Fig 3.c Classifying games based on ESRB Rating

3.1 Exploratory Data Analysis

Understanding the difference between click_time and query_time can provide us more information on the user behaviour. This helps in designing our algorithm for Query and User based recommendation. A visualization table was created using variables: User, Query, Query_time, Name, Click_time and Difference between click and query time (i.e Diff_time) in seconds. The table represents Users grouped based on the Queries searched for and the products clicked on.

For the purpose of our recommendations we decided to not use collaborative filtering due to a noticeable disparity in user behaviours. Based on our observation, we initially found two types of users:

1. Users who searched for one game but clicked on a multitude of games except for the one they searched for as in Fig 3.1.a
2. Users who searched for a multitude of games and clicked exactly on the game they were looking for, as in Fig 3.1.b

However, we later noticed a third type of user who both clicked on the products they searched for as well as other unrelated games. (Fig 3.1.c)

Inference: Our initial assumption that users can be put into two distinct categories was refuted. There was no concrete behaviour pattern with respect to difference in click and query time. Given this behaviour, it would be remiss to lump in all behaviours and preferences as being the same.

Why content based filtering?

It accounts for the erratic user-behaviour. In content based filtering, the recommendations we make will depend entirely on similarity to the game a user is interested in and not on preferences of different users. While we ideally want to make personalized recommendations, by the time we are done with cleaning and preliminary analysis we realized that it is far more viable to make semi-personalized recommendations. This would also allow us to validate our results more accurately.

User	Query	Query Time	Name	Click Time	
e4e3490a79ceb6225f6..	gears of war 3	02:30.9	Battlefield 3 Limited Edition - Xbox 360	03:47.4	76
			Batman: Arkham City Collector's Edition - Xbox 360	04:03.2	92
			Halo 4 - Xbox 360	05:26.3	175
			Tom Clancy's Ghost Recon: Future Soldier - Xbox 360	06:10.4	219
			Warhammer 40,000: Space Marine - Xbox 360	07:23.7	293

Fig 3.1.a Users who clicked on products not searched for

User	Query	Query Time	Name	Click Time	
be1466e856ddd79a7e..	Batman_Arkham..	06:34.9	Batman: Arkham City - Xbox 360	07:02.5	28
	Carnival xbox	59:04.0	Carnival Games: Monkey See, Monkey Do - Xbox 360	59:17.6	14
	Kinect sports	20:50.1	Kinect Sports - Xbox 360	20:53.7	4
			Kinect Sports: Season Two - Xbox 360	21:12.4	22
	Madden 12	42:21.2	Madden NFL 12 - Xbox 360	43:08.3	47

Fig 3.1.b Users who clicked on products searched for

b215506e..	Gears of war 3	46:22.5	Gears of War 3 - Xbox 360	46:46.8	24
			Halo: Combat Evolved Anniversary - Xbox 360	47:56.3	94
			Call of Duty: Modern Warfare 3 - Xbox 360	48:14.8	112
			Battlefield 3 Limited Edition - Xbox 360	48:33.9	131
			Batman: Arkham City Collector's Edition - Xbox 360	49:26.6	184

Fig 3.1.c Users who clicked on products they searched for and later some other products

3.2 Procedure

3.2.1 Product Clusters

As we discarded the possibility of implementing collaborative filtering, we decided to look into content based filtering. For this we first decided to form clusters of similar products for which we chose K-Means unsupervised clustering.

Out of the 5 variables in the Products Catalogue we chose, the Genre, Manufacturer and ESRB rating were converted to dummies. The Short and Long Description of the games were concatenated after cleaning to form a single Description field. The ESRB descriptors were also stemmed and cleaned. The Description and ESRB descriptors were then vectorized. However, in order to get more uniform clusters

we decided to explore the accurate cluster size using CountVectorizer and TfidfVectorizer for Unigrams and Unigrams & Bigrams each, using the Elbow Method.

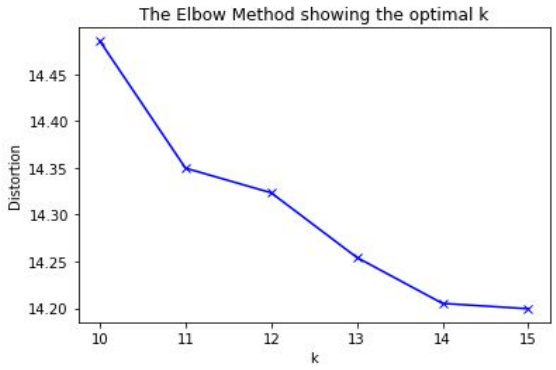

Fig 3.2.1.a CountVectorizer: Unigrams

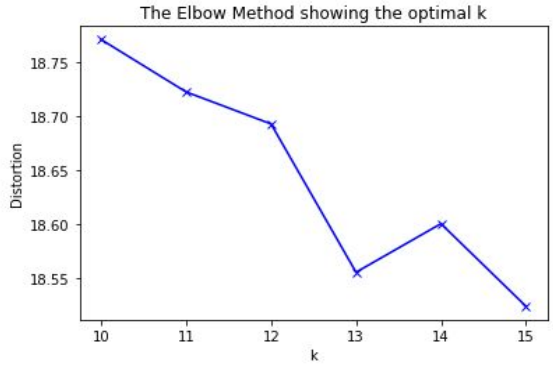

Fig 3.21..b CountVectorizer: Unigrams & Bigrams

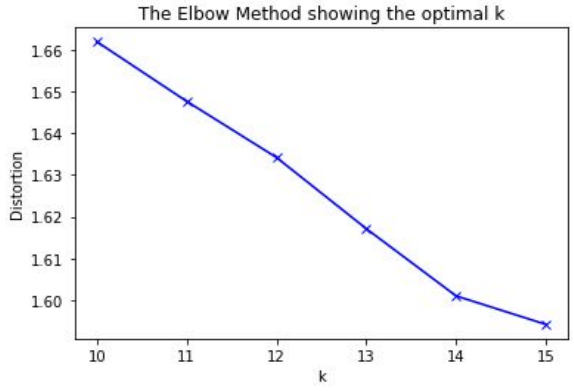

Fig 3.2.1.c TfidfVectorizer: Unigrams

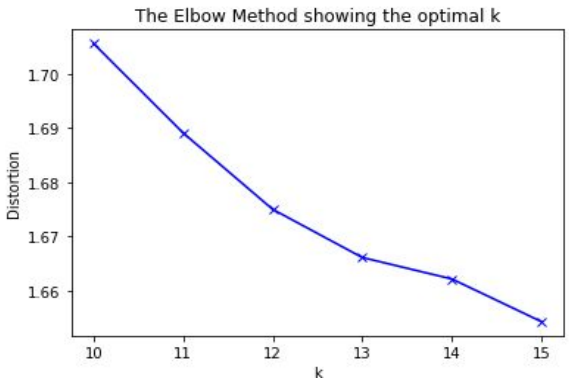

Fig 3.2.1.d TfidfVectorizer: Unigrams & Bigrams

Out of all the permutations, we chose TfidfVectorizer with both Unigrams and Bigrams (Fig 3.2.1.d) where there is an indistinct elbow at k = 13. Even though we see very clear elbows in the rest of the three figures, the cluster distribution was not very uniform. They had at least 5 to 7 clusters with only 1-2 products and few clusters having more than 150 products. Such cluster distributions could not really help us in producing more reliable results. In TfidfVectorizer with Unigrams & Bigrams, we got the cluster distribution as seen in Fig 3.2.1.e, where the smallest cluster has 15 products and the biggest cluster has close to 55 products, which is a more uniformly distributed cluster as compared to rest all. Visualizing our final cluster on the first two axes of PCA, we then see the clusters as seen in Fig 3.2.1.f.

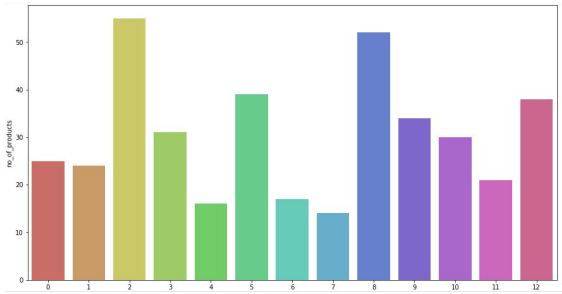

Fig 3.2.1.e Cluster distribution of TfidfVectorizer: Unigrams & Bigrams

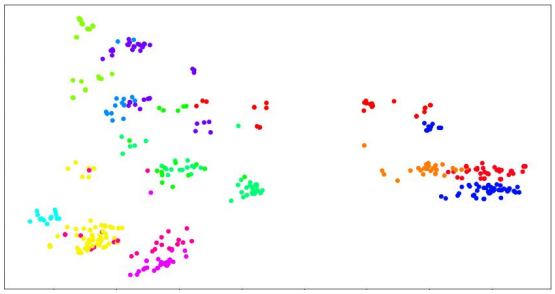

Fig 3.2.1.f Cluster on PCA two axes

3.2.2 Search Queries

In our train and test data, we looked into the patterns of user queries and we found out that there were several queries referring to the same product but they differed by a couple of characters or words (Fig 3.2.2.a), which we wanted to address.

3.2.2.1 Spell-check with best match algorithm

For every user query, we calculated cosine similarity with all the names of the products in our product catalogue and returned the best match product name with the highest cosine similarity as the "spell-checked user query". This helped us club all the similar queries and also get rid of any spelling mistakes in any of the queries.

At the end of this what we are left with is a list of users and their cleaned, spell-checked search queries and our clusters of similar products based on our selected product features.

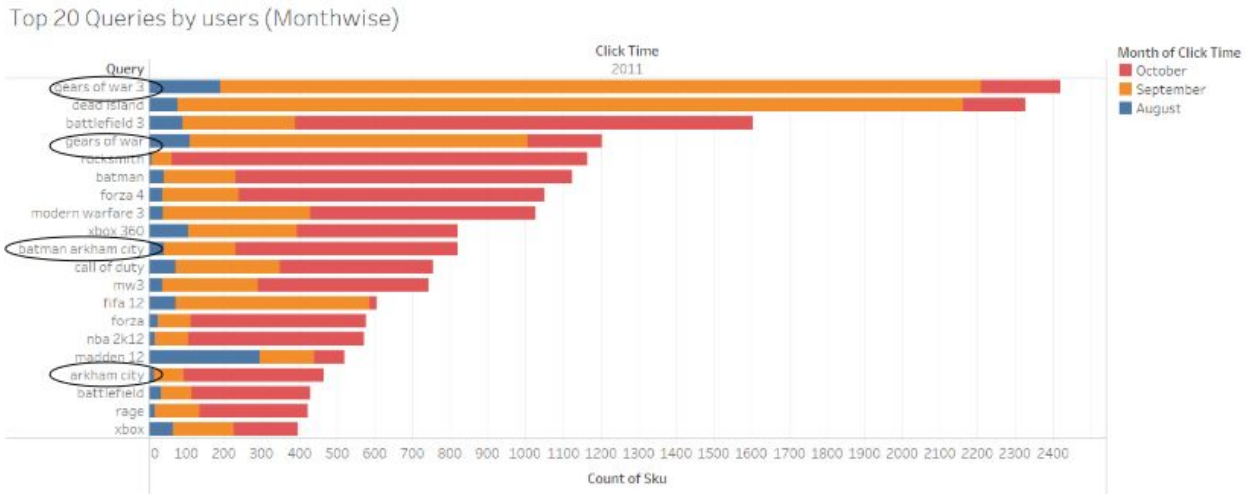

Fig 3.2.2.a Duplicate queries

3.3 Evaluation Metric

The idea behind a recommendation system is that it knows what a user would like before a user has to look for it. We wanted a validation metric that would help us check if the games we suggest for a user are items which the user is actually interested in.

Our training data by itself was extremely sparse (no user had more than 6 unique queries). Therefore, we decided to club all the queries across the train and test data sets. Since we had established that user behaviour was very erratic with regard to clicks, we could not rely on that attribute to give us any significant information. We decided that we will be basing our research only based on the user queries. The assumption is that if a user explicitly searches for something, they are likely to be interested in the product. Going by this, we clubbed the train and the test

data with only the user IDs and the search queries. We ran our spell check algorithm on these queries to match them to actual product titles.

To get viable results, we got rid of all users with less than 4 search queries, after which we were left with a total of 551 users. For each of these users, we randomly sampled 2 queries as our Validation products and the rest as our Training products (Fig 3.3.a). The maximum number of queries for any user was 8. At this point, we had the products each user looked for matched actual titles, and a cluster of similar products.

The final step was to generate a list of 10 recommendations for each user from these clusters, and check to see if our validation products are present in this list. The proportion of users for which the above holds true is called the *Hit rate* of our system, which was the most appropriate metric for our purposes.

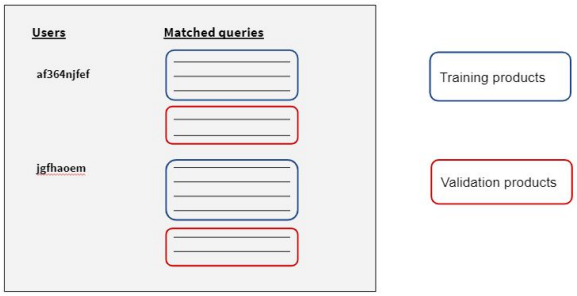

Fig 3.3.a Splitting our data

3.3.1 Recommendations

For each user, we looked at the training products and found the clusters they belonged to, and from those clusters we retrieved 10 recommendations considering the following:

1. **Weighted by cluster:** We adjusted the proportions returned from each cluster based on a user's preferences. As an example, if out of three queries in our training products, 2 belong to the same cluster, we retrieve more products from this cluster as compared to the others.
2. **Non-repeating:** None of the products in recommendations repeat.
3. **Filtered by rating:** We do not return products with an average user rating less than 3 on a scale of 1 - 5.
4. **Sorted by popularity:** We calculated popularity of each product based on how many times a product was searched for. When returning from the cluster, we sorted the most popular products first.
5. **Newest first:** Some games like Gears of war have multiple versions, in cases like these we returned "Gears of war forza 4" over "Gears of war 3".

3.3.2 Validation

The top 10 products returned in the recommendations list for each user were then compared with the actual searches made by the user which we kept aside as our Validation products. The final accuracy of this system was then calculated in terms of how many users we got both of them right, and for how many we got only 1 right.

4. Results

Of 551 users, we were able to generate accurate recommendations for 174.

130 users got fully accurate recommendations i.e. both their validation products were present in the 10 recommendations, whereas 44 users got half-accurate recommendations i.e. one of their products was present in the recommendations.

By this definition, our model has an accuracy of approximately 32%.

4.1 Limitations

The data we used was extremely sparse. Upon eliminating redundancies, performing transformations, and subsetting the data to include only relevant observations for our research question, the original 41,000 records were reduced to 551. This was not a loss of information. Rather, it was a result of restructuring data for the purpose of validation. We needed more data in two regards:

1. The number of users that made search queries.
2. The number of unique queries per user.

This additional information would lend more credence to our results, and perhaps increase our accuracy.

Most importantly, it would allow us to use precise metrics of evaluation like *Average reciprocal hit rate* which accounts not only for the overlap between validation sets and training sets, but also the position at which validation items show up in our recommendations.

4.2 Future Directions

Our approach towards solving this problem explored only the rudiments of a recommendation engine. Given more time and effort, the analysis could potentially benefit from using more modern techniques, some of which are:

1. Matrix Factorization methods (Singular Value Decomposition)
2. Deep Learning (Restricted Boltzman Machines)

5. References

Anderson C (2006) *The long tail: how endless choice is creating unlimited demand.* Random House, New York

Brynjolfsson E, Hu YJ, Smith MD (2006) *From niches to riches: anatomy of the long tail.* Sloan Manag Rev 47(4):67–71

Pal, A., Parhi, P., & Aggarwal, M. (2017). *An Improved Content Based Collaborative Filtering Algorithm For Movie Recommendations.* IEEE Xplore. Retrieved from https://ieeexplore-ieee-org.offcampus.lib.washington.edu/stamp/stamp.jsp?tp=&arnumber=8284357

Rajendra, N., Dewan, A., & Colakoglu, M. C. (2012). *Best Buy Recommendation System.*

Data Mining Hackathon on (20MB) Best Buy mobile web site - ACM SF Bay Area Chapter. (n.d.). Retrieved from https://www.kaggle.com/c/acm-sf-chapter-hackathon-small/data

6. Appendix

Data:

Products:

shortDescription	manufacturer	esrbRating	longDescription	genre	esrb_description
control power deat	City Interactive	M=Mature	hand steadi lock onto targe	na	na
say monopoli illeg	Electronic Arts	E=Everyone	watch teammat roll dice h	board game trivia	na
take sky high fli acti	Electronic Arts	E10+=Everyone	bright blue sky stretch ban	flight	cartoon violenc
soccer action got lo	EA SPORTS	E=Everyone	hurtl field cleat bare dig di	sport outdoor	na
take fun amus park	Electronic Arts	E=Everyone	watch favorit game come l	board game trivia	mild violenc
give sim run neighb	Electronic Arts	T=Teen	throw open door new real	strategi simul	crude humor sexual theme
never much advent	SouthPeak	M=Mature	majest landscap antaloor s	role play	blood partial nuditi sexual t
experi raw vicious i	Activision	M=Mature	draft time travel globe figh	shooter	na
save world end day	Konami	M=Mature	castlevania lord shadow er	action adventur	blood gore nuditi violenc bl
switch gear feel nee	Electronic Arts	E10+=Everyone	fasten seatbelt get readi dr	race	violenc
readi next legendar	Capcom	T=Teen	readi two power univers c	fight	mild languag partial nuditi s
use guitar hero skill	Activision	T=Teen	guitar hero warrior rock jo	music danc parti	lyric mild fantasi violenc mil
take turntabl becon	Activision	T=Teen	light flash dark air hot hea	music danc parti	lyric mild suggest theme
slam sweep smash	THQ	T=Teen	watch jack swagger chokes	sport outdoor	blood languag suggest them
make america hom	THQ	M=Mature	learn homefront flash dem	shooter	blood strong languag violen

Train:

user	sku	category	query	click_time	query_time
0001cd0d10bbc585c9ba287c963e00873d4c0bfd	2032076	abcat0701002	gears of war	22:56.1	21:42.9
00033dbced6acd3626c4b56ff5c55b8d69911681	9854804	abcat0701002	Gears of war	35:42.2	35:33.2
00033dbced6acd3626c4b56ff5c55b8d69911681	2670133	abcat0701002	Gears of war	36:08.7	35:33.2
00033dbced6acd3626c4b56ff5c55b8d69911681	9984142	abcat0701002	Assassin creed	37:23.7	37:00.0
0007756f015345450f7be1df33695421466b7ce4	2541184	abcat0701002	dead island	15:34.3	15:26.2
000878e35cb70ace315dbdbef54c22477066f07e	3046066	abcat0701002	Rocksmith	44:36.8	44:22.9
0008b7e06bc3d329b4dc052268e10b98c9121452	2977637	abcat0701002	Nba n2k	10:06.5	09:33.5
0008f35cccf771838c635196205951403f13da50	2670133	abcat0701002	Call of duty	05:40.9	05:06.4
000c22a204a9248e3e5458ed52a084b98ddfcf56	9328943	abcat0701002	rock band	02:54.2	02:44.2
000f573f22d30ead7a0a3d7fce64d3eb1cb8d0be	1180104	abcat0701002	xbox	39:51.7	36:40.2
000f573f22d30ead7a0a3d7fce64d3eb1cb8d0be	2598445	abcat0701002	xbox	41:16.2	36:40.2
001020de898602550 7e5e05633b60c408861c5f6	1563461	abcat0701002	Crysis 2	47:22.6	46:56.0

Test:

user	category	query	click_time	query_time
00025eb02b249434554fe2cacd8562db325df127	abcat0701002	child eden	05:24.1	04:34.6
00033dbced6acd3626c4b56ff5c55b8d69911681	abcat0701002	Revelations	38:14.7	37:58.4
000548d17532b70071b7d59edd4797aed1823c60	abcat0701002	Gears of war	34:43.6	33:43.2
0006f15231a422156a9d005735d0969a5e5a0ac4	abcat0701002	batman	14:48.4	14:33.7
000a16ce5371b0fb3ad0c7f6183a5476b434a95b	abcat0701002	Assassins creed: revelations	31:53.1	31:39.8
000a16ce5371b0fb3ad0c7f6183a5476b434a95b	abcat0701002	Assassins creed: revelations	33:07.0	31:39.8
000b4145ab7f939f09f049c6058bbab49e7c7bad	abcat0701002	Gears of war	38:52.6	36:52.3
000c29a2b54b786aa00a67644d926056feeaaae1	abcat0701002	minexraft	15:00.2	14:38.1
000e334a6026a051898a235d0b7907c0c5d8b0bc	abcat0701002	forza	51:27.9	51:13.3
001030de8986025507e5e95633b60c408861c5f6	abcat0701002	Masters 12	44:35.6	44:24.6
001140869525532997ed12fedebc98a7944e184e	abcat0701002	Halo reach legendary edition	31:09.8	30:57.8
0017808de8955e5adfb9bf4f942835bf50f49296	abcat0701002	Ufc personal trainer	00:22.8	00:11.4
0018142557 8d2f4b2047aaf2a26da05204a56f78	abcat0701002	Madden 12	18:04.0	15:17.6

Visualization based on Grouping Users based on Query and Products Clicked on: