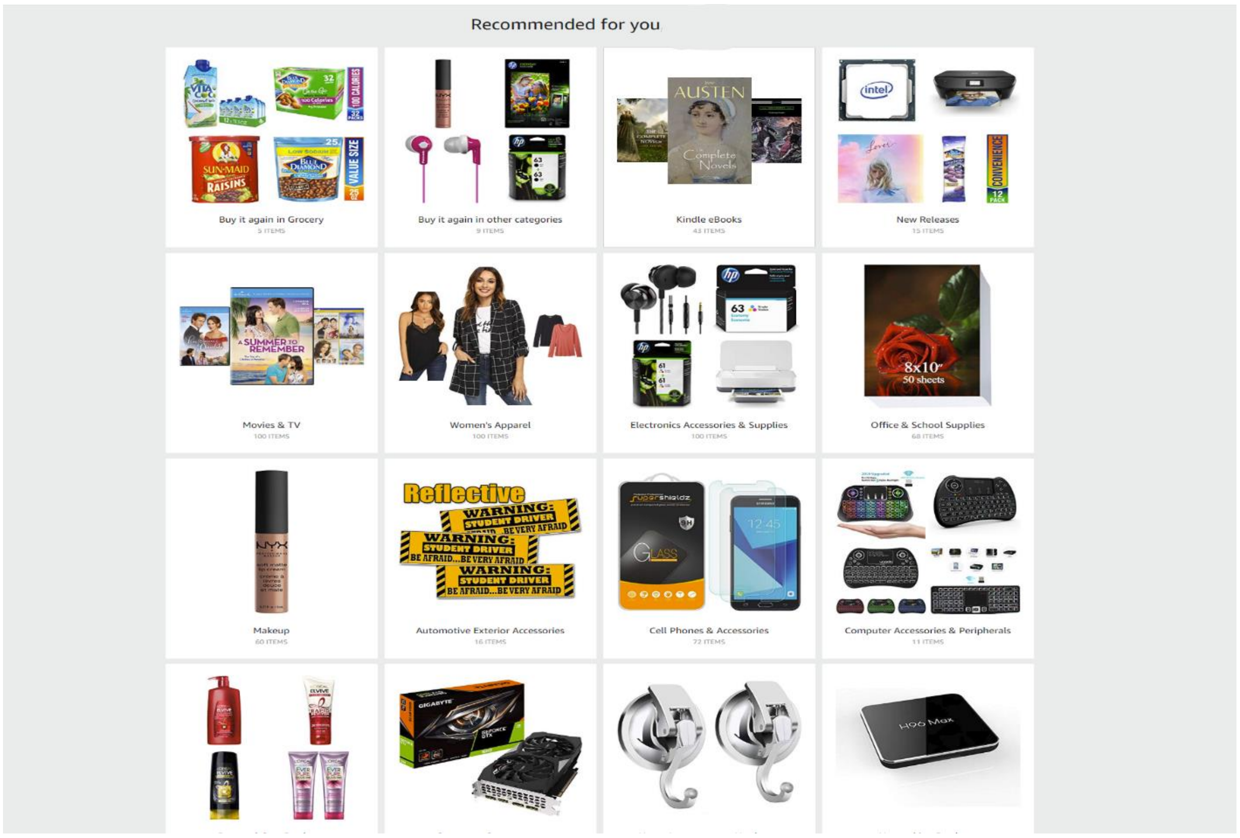


Amazon Product Recommender

29.06.2019

—

Dr. Uma Gajendragadkar
Seattle, WA

Introduction

Recommender Systems have become essential part of the most online applications providing personalized suggestions for online shopping, books to read, music to tune to and movies to watch. Recommender systems are also called recommendation engines.

Recommender systems personalize customer experience by understanding their usage of the system and recommending items they would find useful.

Companies carrying large volume of distinct items use recommender systems to suggest individual items to customers. Amazon, Netflix, Spotify, Instagram, YouTube, Facebook all use recommender systems to help their online customers to find useful items from the large volume of individual items – products, books, films, electronics etc. present in their content catalogues.

Motivation: Why you need a recommender system

1) Recommender systems help businesses by changing the way websites communicate with users. They help businesses to maximize their profit based on the information they gather about each customers preferences and purchases. For businesses, recommender systems drive engagement with content and increase revenue.

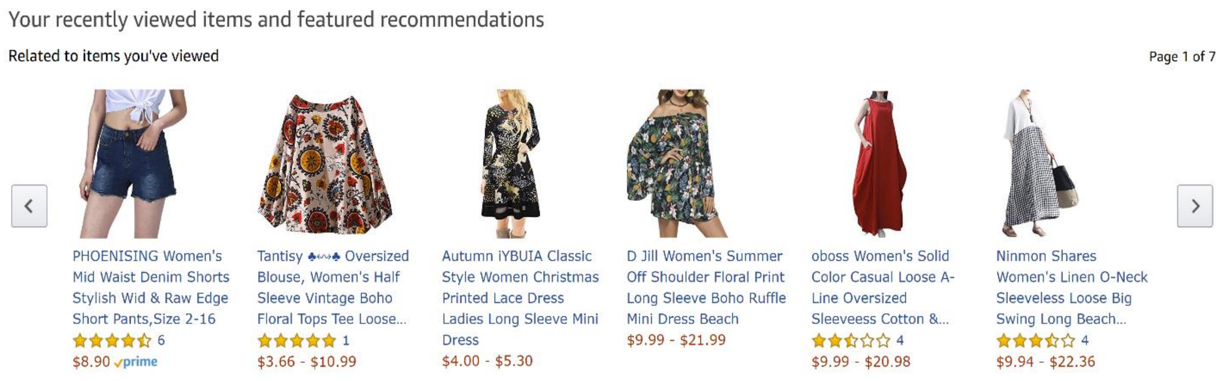
2) Now-a-days, customers are overwhelmed by information overload rather than lack of information. Companies like Amazon, Flipkart, e-bay etc. have millions of items/products available on their websites. Customers have to spend a lot of time browsing through the catalogues to find the right product. Recommender systems help customers by suggesting probable list of products from which they can easily select the right one. They make customers aware of the new and/or similar products available for purchase by providing comparable costs, features, delivery times etc.

Title of Project: Amazon Product Recommendation Engine

1) Problem to be solved

Whenever users visit online shopping sites like Amazon, Walmart, Ebay, flipkart etc. they search for various products to purchase. And many give reviews/ratings for a purchased product. We can make use of these reviews/ratings to recommend different products to current user.

Existing recommendation algorithms couldn't scale to Amazon's tens of millions of customers and products, so they decided to develop their own. Amazon currently uses item-to-item collaborative filtering, which scales to massive data sets and produces high-quality recommendations in real time. [11]



Your recently viewed items and featured recommendations

Related to items you've viewed

Page 1 of 7

PHOENISING Women's Mid Waist Denim Shorts Stylish Wid & Raw Edge Short Pants,Size 2-16
6
$8.90 prime

Tantisy ♣ꕥ♣ Oversized Blouse, Women's Half Sleeve Vintage Boho Floral Tops Tee Loose...
1
$3.66 - $10.99

Autumn iYBUIA Classic Style Women Christmas Printed Lace Dress Ladies Long Sleeve Mini Dress
$4.00 - $5.30

D Jill Women's Summer Off Shoulder Floral Print Long Sleeve Boho Ruffle Mini Dress Beach
$9.99 - $21.99

oboss Women's Solid Color Casual Loose A-Line Oversized Sleeveess Cotton &...
4
$9.99 - $20.98

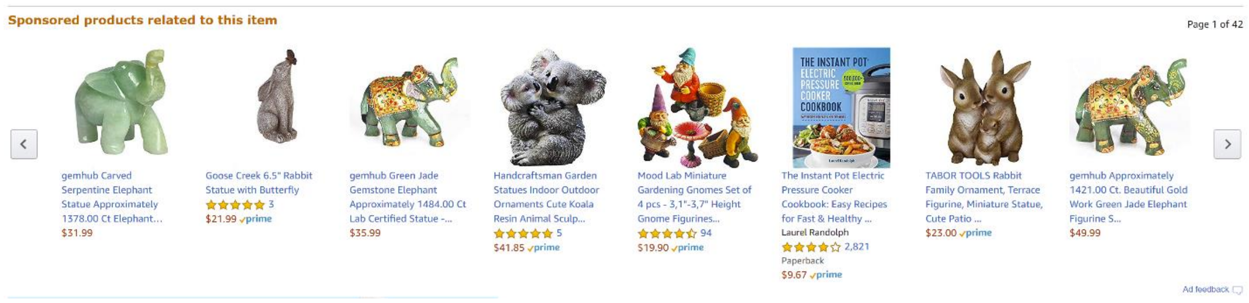
Ninmon Shares Women's Linen O-Neck Sleeveless Loose Big Swing Long Beach...
4
$9.94 - $22.36

We can build a recommendation engine based on

a. Frequently Bought Together (Combo items)

b. Customers Who Bought This Item Also Bought

c. Related products ("users who purchased this also purchased")



Sponsored products related to this item

Page 1 of 42

gemhub Carved Serpentine Elephant Statue Approximately 1378.00 Ct Elephant...
$31.99

Goose Creek 6.5" Rabbit Statue with Butterfly
3
$21.99 prime

gemhub Green Jade Gemstone Elephant Approximately 1484.00 Ct Lab Certified Statue -...
$35.99

Handcraftsman Garden Statues Indoor Outdoor Ornaments Cute Koala Resin Animal Sculp...
5
$41.85 prime

Mood Lab Miniature Gardening Gnomes Set of 4 pcs - 3,1"-3,7" Height Gnome Figurines...
94
$19.90 prime

The Instant Pot Electric Pressure Cooker Cookbook: Easy Recipes for Fast & Healthy ...
Laurel Randolph
2,821
Paperback
$9.67 prime

TABOR TOOLS Rabbit Family Ornament, Terrace Figurine, Miniature Statue, Cute Patio ...
$23.00 prime

gemhub Approximately 1421.00 Ct. Beautiful Gold Work Green Jade Elephant Figurine S...
$49.99

Ad feedback

d. Category wise best seller

e. Brand wise best seller

f. Similar Products

2) Stakeholders

Users/Customers visiting online shopping sites, e-commerce companies like Amazon, Walmart, E-bay, Flipkart, Netflix etc.

3) Data source and aquisition

Data from: **UCSD** http://jmcauley.ucsd.edu/data/amazon/

This dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014. This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).

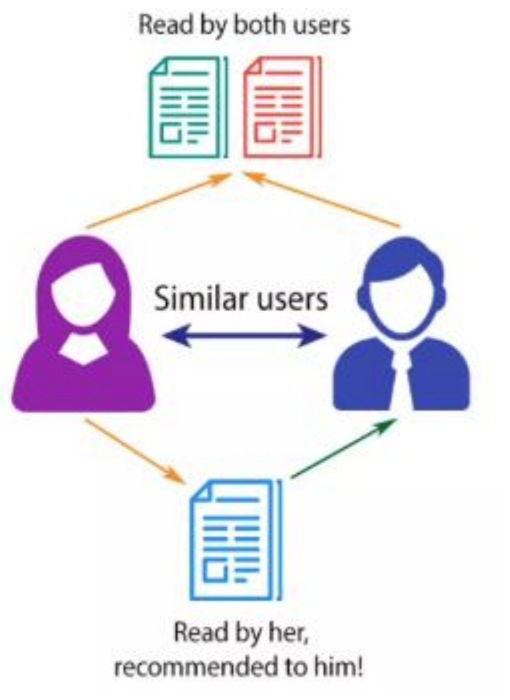
4) Approach: Collaborative filtering

My implementation of recommender system will use collaborative filtering (CF) approach. CF is a recommender systems technique that helps people discover items that are most relevant to them.

“Collaborative filtering (user-based filtering) assumes that if users who are similar to the current user like some items, the current user might also like it. Collaborative Filtering is defined purely in terms of user-user similarity measures. It makes two basic assumptions. The first is that the users with a common interest will have similar item preferences and second that the users with similar preferences share the same interest. For example, two friends with similar preferences for books are very likely to read what the other has already read. In this approach we infer an individual's interest based on other similar users. The actual content of the items these similar users prefer does not have any significance. However, this approach requires a large amount of user preferences data to be available.” [10]

The growth of data on the web has made it harder to employ many machine learning algorithms on the full data sets. For personalization problems in particular, where data sampling is often not an option, innovating on distributed algorithm design is necessary to allow us to scale to these constantly growing data sets.

CF is based on the idea that the best recommendations come from people who have similar tastes. In other words, it uses historical item ratings of like-minded people to predict how someone would rate an item.



Matrix factorization

A very popular, collaborative filtering method is matrix factorization. It has a set of users and a set of items, and a very sparse matrix that represents known user-to-item ratings. We want to predict missing values in this matrix. In order to do this, we represent each user and each item as a vector of latent features, such that dot products of these vectors closely match known user-to-item ratings. The expectation is that unknown user-to-item ratings can be approximated by dot products of corresponding feature vectors, as well. The simplest form of objective function, which we want to minimize, is:

$$\min \sum_{ratings\ u,i} (r_{u,i} - x_u \cdot y_i)^2 + \gamma \cdot \overbrace{\left(\sum_{users\ u} \|x_u\|^2 + \sum_{items\ i} \|y_i\|^2 \right)}^{regularization}$$

Here, r are known user-to-item ratings, and x and y are the user and item feature vectors that we are trying to find. As there are many free parameters, we need the regularization part to prevent overfitting and numerical problems, with gamma being the regularization factor.

It is not currently feasible to find the optimal solution of the above formula in a reasonable time, but there are iterative approaches that start from random feature vectors and gradually improve the solution. After some number of iterations, changes in feature vectors become very small, and convergence is reached.

Alternating least square

Alternating least square (ALS) is one of the methods for Matrix factorization. It is used with nonlinear regression models, when there are two dependent variables (in our case, vectors x and y). The algorithm fixes one of the parameters (user vectors x), while optimally solving for the other (item vectors y) by minimizing the quadratic form. The algorithm alternates between fixing user vectors and updating item vectors, and fixing item vectors and updating user vectors, until the convergence criteria are satisfied.

The Matrix Factorization consist in decomposing the original rating matrix into two, smaller matrices U and P. The recommendation is obtained after multiplying the two of them. As these matrices are forced to be much smaller than the original one, its product will produce a rating matrix without zeros (mathematically, the decomposing matrices U and P have a rank k, with k typically around several tens or a hundred, that is much smaller than the number of users or items).

At this point, it is important to realize the scale of the rating matrix. Thus, we needed to take a scalable, likely affordable solution to this problem.

Deep dive into Spark's Recommender System

There are several efficient and scalable implementations of Matrix Factorization in the industry. Among them, a prominent one is that provided by Apache Spark, a distributed data-processing engine that can be run easily on Amazon Web Services with an Elastic Mapreduce cluster.

Apache Spark implements a distributed version of the Alternating Least Square (ALS) method with Weight Regularization.

5) Is this a supervised or unsupervised problem?

It is a Supervised problem

6) If supervised is it a classification or regression problem?

It is more of a regression problem in this case as I am trying to predict the rating of a product to be recommended

7) What variable is it you are trying to predict?

Rating of the product/item

8) What variables will you use as predictors?

Similarity score between items/products and users

10) What will be your training data?

Available data will be divided as training and testing dataset

11) What are your deliverables?

Algorithm for product recommendation (Code in Python), Charts, Presentation, Report, a blogpost

Data Wrangling

Data Acquisition:

There are two publicly available websites for acquiring the data for Amazon product recommendation.

1) Directly from **Amazon** https://s3.amazonaws.com/amazon-reviews-pds/readme.html

2) Data from: **UCSD** http://jmcauley.ucsd.edu/data/amazon/

I obtained data from UCSD website. I had to exchange few emails with Julian McAuley, UCSD and explain the purpose of my Capstone project and why I need the data. Thanks to Professor McAuley and team for making this dataset available.[1][2]

Challenges:

The data files are huge. I could read the data and clean and load it in pandas dataframe but when I tried to create a sparse matrix of users, items and ratings, my laptop ran out of memory and was unable to handle the huge number of users and items. **So I switched to Apache Pyspark.**

"Spark is an open-source distributed computing framework that promises a clean and pleasurable experience similar to that of Pandas, while scaling to large data sets via a distributed architecture under the hood. It does in-memory, distributed and iterative computation, which is particularly useful when working with machine learning algorithms. Other tools might require writing intermediate results to disk and reading them back into memory, which can make using iterative algorithms painfully slow." [3]

Data Description:

The dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014.

This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).

K-cores (i.e., dense subsets): These data have been reduced to extract the k-core, such that each of the remaining users and items have k reviews each.

Ratings only: These datasets include no metadata or reviews, but only (user, item, rating, timestamp) tuples

5-core (9.9gb) - subset of the data in which all users and items have at least 5 reviews (41.13 million reviews)

Finally, the following file removes duplicates more aggressively, removing duplicates even if they are written by different users. This accounts for users with multiple accounts or plagiarized reviews. Such duplicates account for less than 1 percent of reviews.

Different product categories available in dataset:

Books, Electronics , Movies and TV, CDs and Vinyl, Clothing, Shoes and Jewelry , Home and Kitchen, Kindle Store, Sports and Outdoors, Cell Phones and Accessories, Health and Personal Care, Toys and Games, Video Games, Tools and Home Improvement, Beauty, Apps for Android, Office Products, Pet Supplies, Automotive, Grocery and Gourmet Food, Patio, Lawn and Garden, Baby, Digital Music, Musical Instruments, Amazon Instant Video

Sample review:

{ "reviewerID": "A2SUAM1J3GNN3B", "asin": "0000013714", "reviewerName": "J. McDonald", "helpful": [2, 3], "reviewText": "I bought this for my husband who plays the piano. He is having a wonderful time playing these old hymns. The music is at times hard to read because we think the book was published for singing from more than playing from. Great purchase though!", "overall": 5.0, "summary": "Heavenly Highway Hymns", "unixReviewTime": 1252800000, "reviewTime": "09 13, 2009" }

where

1) reviewerID - ID of the reviewer, e.g. A2SUAM1J3GNN3B

2) asin - ID of the product, e.g. 0000013714

3) reviewerName - name of the reviewer
4) helpful - helpfulness rating of the review, e.g. 2/3
5) reviewText - text of the review
6) overall - rating of the product
7) summary - summary of the review
8) unixReviewTime - time of the review (unix time)
9) reviewTime - time of the review (raw)

Metadata

Metadata includes descriptions, price, sales-rank, brand info, and co-purchasing links:

metadata (3.1gb) - metadata for 9.4 million products

Sample metadata:

{ "asin": "0000031852", "title": "Girls Ballet Tutu Zebra Hot Pink", "price": 3.17, "imUrl": "http://ecx.images-amazon.com/images/I/51fAmVkTbyL._SY300_.jpg", "related": { "also_bought": ["B00JHONN1S", "B002BZX8Z6", "B00D2K1M3O", "0000031909", "B00613WDTQ", "B00D0WDS9A", "B00D0GCI8S", "0000031895", "B003AVKOP2", "B003AVEU6G", "B003IEDM9Q", "B002R0FA24", "B00D23MC6W", "B00D2K0PA0", "B00538F5OK", "B00CEV86I6", "B002R0FABA", "B00D10CLVW", "B003AVNY6I", "B002GZGI4E", "B001T9NUFS", "B002R0F7FE", "B00E1YRI4C", "B008UBQZKU", "B00D103F8U", "B007R2RM8W"], "also_viewed": ["B002BZX8Z6", "B00JHONN1S", "B008F0SU0Y", "B00D23MC6W", "B00AFDOPDA", "B00E1YRI4C", "B002GZGI4E", "B003AVKOP2", "B00D9C1WBM", "B00CEV8366", "B00CEUX0D8", "B0079ME3KU", "B00CEUWY8K", "B004FOEEHC", "0000031895", "B00BC4GY9Y", "B003XRKA7A", "B00K18LKX2", "B00EM7KAG6", "B00AMQ17JA", "B00D9C32NI", "B002C3Y6WG", "B00JLL4L5Y", "B003AVNY6I", "B008UBQZKU", "B00D0WDS9A", "B00613WDTQ", "B00538F5OK", "B005C4Y4F6", "B004LHZ1NY", "B00CPHX76U", "B00CEUWUZC", "B00IJVASUE", "B00GOR07RE", "B00J2GTM0W", "B00JHNSNSM", "B003IEDM9Q", "B00CYBU84G", "B008VV8NSQ", "B00CYBULSO", "B00I2UHSZA", "B005F50FXC", "B007LCQI3S", "B00DP68AVW", "B009RXWNSI", "B003AVEU6G", "B00HSOJB9M", "B00EHAGZNA", "B0046W9T8C", "B00E79VW6Q", "B00D10CLVW", "B00B0AVO54", "B00E95LC8Q", "B00GOR92SO", "B007ZN5Y56", "B00AL2569W", "B00B608000", "B008F0SMUC", "B00BFXLZ8M"], "bought_together": ["B002BZX8Z6"] }, "salesRank": {"Toys & Games": 211836}, "brand": "Coxlures", "categories": [["Sports & Outdoors", "Other Sports", "Dance"]] }

where

1) asin - ID of the product, e.g. 0000031852

2) title - name of the product

3) price - price in US dollars (at time of crawl)

4) imUrl - url of the product image

5) related - related products (also bought, also viewed, bought together, buy after viewing)

6) salesRank - sales rank information

7) brand - brand name

8) categories - list of categories the product belongs to

1. What kind of cleaning steps did you perform?

I used the following steps to clean the data. [4][5]

a) Dropping Columns in a DataFrame

I selected only those columns which are necessary from the dataset and created a subset of the dataset which will be used to build the recommender.

b) Checked for duplicates by using df.drop_duplicates(). The dataset I obtained is mostly deduplicated.

c) Checked for invalid data

d) Type checks

e) ratings if between 1 to 5, Rangecheck

f) All ids look reasonable

2. How did you deal with missing values, if any?

In certain scenarios, it may be necessary to remove rows and columns with missing data from a DataFrame. The .dropna()method is used to perform this action.

I used df.dropna() method on the dataset to delete rows with missing values.

3. Were there outliers, and how did you handle them?

In data subset selected for the project there are no outliers as userid, productid are ids and rating is checked for range between 1 to 5. A scatterplot, zscore or checking if data is within limits of 1.5*interquartile range are some of the methods to check for outliers.

Data Story and Exploratory Data Analysis

Description of Data selected

I am using 'Patio, Lawn and Garden' category dataset for building recommender system. It could be extended to other category datasets of Amazon, provided enough memory is available on your machine or one can use AWS EC2 or other cloud computing solutions.

Dataframe patio_data created from 'reviews_Patio_Lawn_and_Garden_5.json.gz' file contains data for attributes { reviewrID - userid, asin - productid, reviewerName, heplful - how helpful review is, reviewText, overall - rating given for a product, summary, unixReviewTime and ReviewTime} Dataframe patio_meta created from 'meta_Patio_Lawn_and_Garden.json.gz' file contains data for attributes {asin - productid, description- product description, title -Title of product, imUrl - image url of product, related - list of also viewed products , salesRank, categories- list of categories to which product belongs, price- price of product, brand- brandname to which product belongs} As I am going to build a recommender system based on Ratings given to products, I have to create subset of data by merging both dataframes and choose required attributes only.

Dataset comprises of 12732 observations and 12 attributes.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12732 entries, 0 to 13271
Data columns (total 12 columns):
reviewerID        12732 non-null object
asin              12732 non-null object
reviewerName      12732 non-null object
helpful           12732 non-null object
reviewText        12732 non-null object
overall           12732 non-null float64
summary           12732 non-null object
unixReviewTime    12732 non-null datetime64[ns]
reviewTime        12732 non-null object
title             12732 non-null object
categories        12732 non-null object
price             12732 non-null float64
dtypes: datetime64[ns](1), float64(2), object(9)
memory usage: 1.3+ MB
```

It is also a good practice to know the columns and their corresponding data types, along with finding whether they contain null values or not.

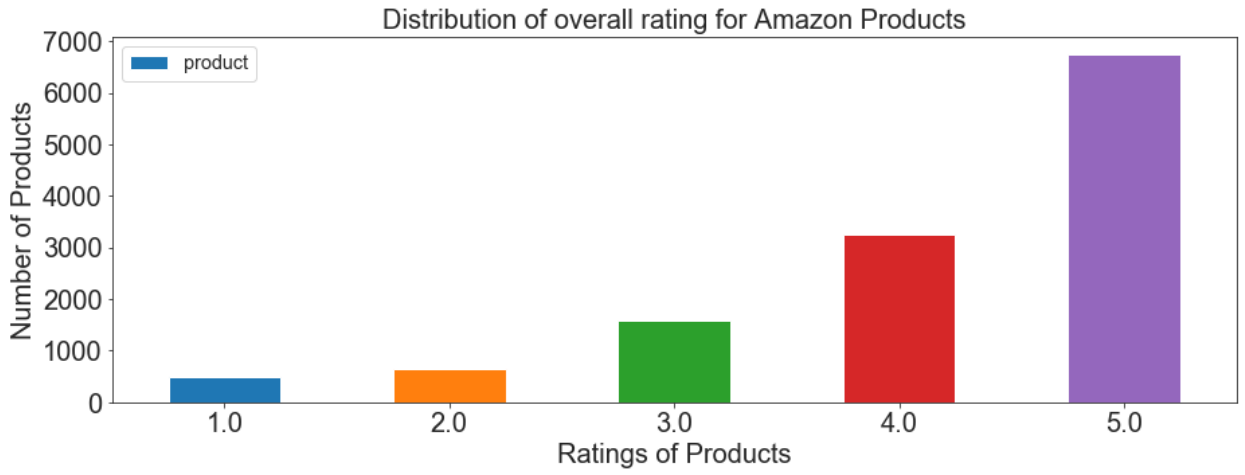
Data has 2 float columns, 1 dateTime and 9 object type columns. No variable column has null/missing values.

Number of unique products = 919

Number of unique users = 1685

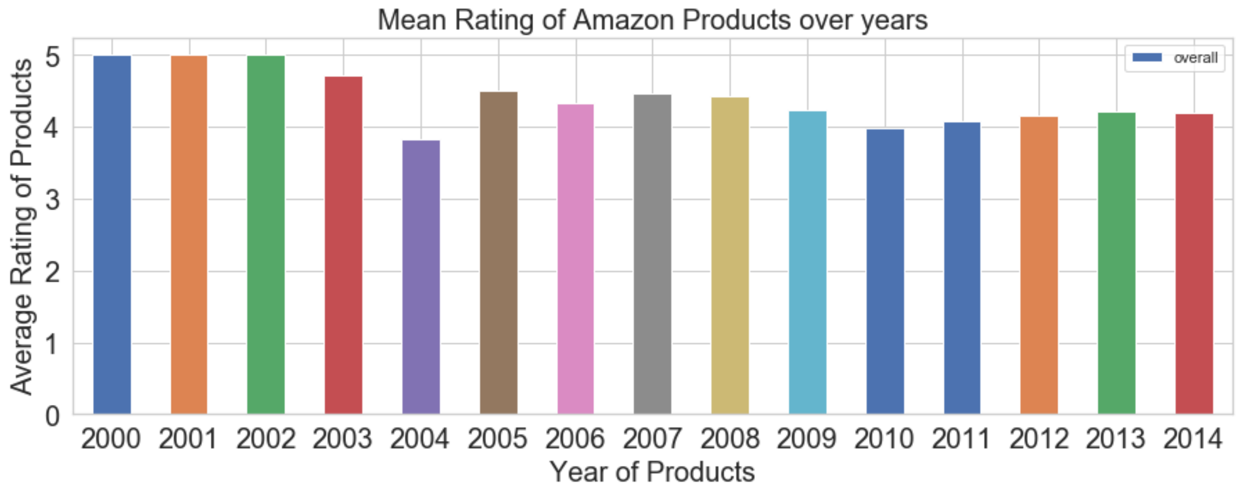
Number of ratings = 12732

1. Distribution of overall rating for Amazon Products



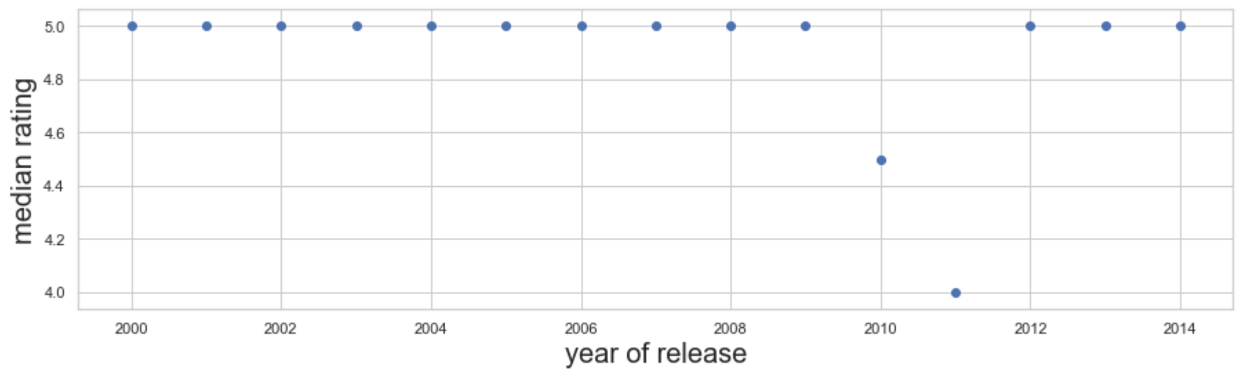
Above plot shows that many users have given 5 rating to prodcuts followed by 4 and 3 whereas very few users have given a low rating of 1 or 2.

2. Mean rating of Amazon Products over years



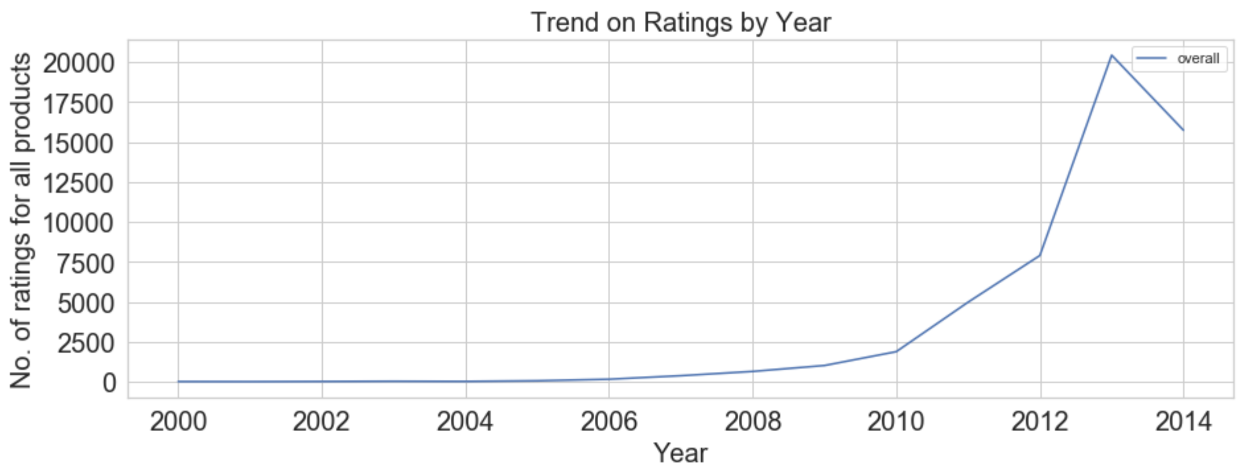
Looking at above plot, we can infer that over the years from 2000 to 2014, the mean rating of the products has reduced.

3. Median of ratings given to products



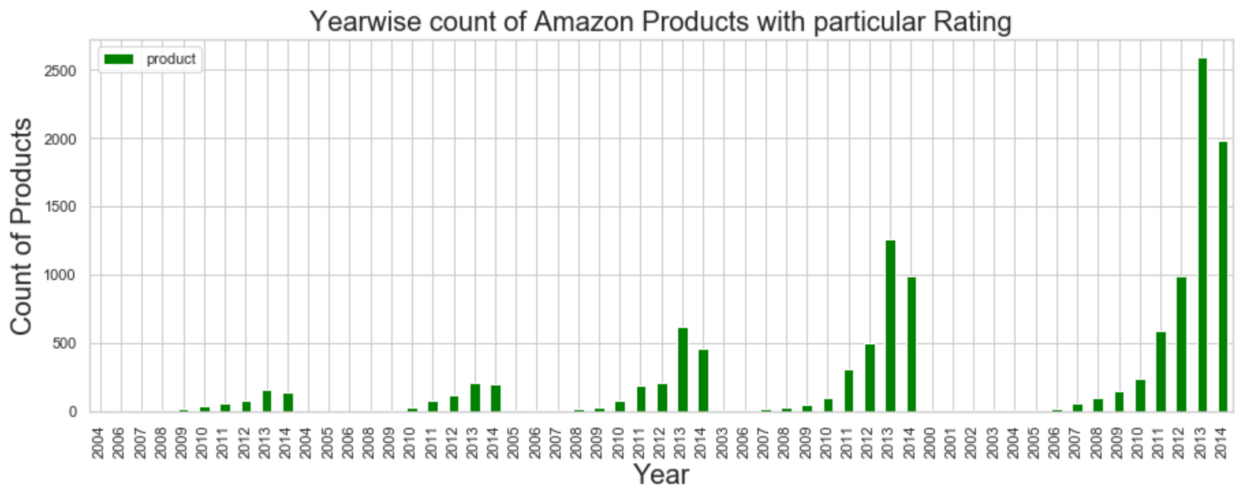
Median of ratings given to products remain at 5 from 2000 to 2014 except for years 2010 and 2011

4. Trend on ratings by year



There is an increasing trend for number of ratings given by the users to products on Amazon which indicates that more number of users started using Amazon ecommerce site for online shopping and more number of users started giving feedback on the products purchased from 2000 to 2014. There is a significant increase in number of ratings given by users from 2012 to 2014.

5. Year wise count of Amazon products with particular rating



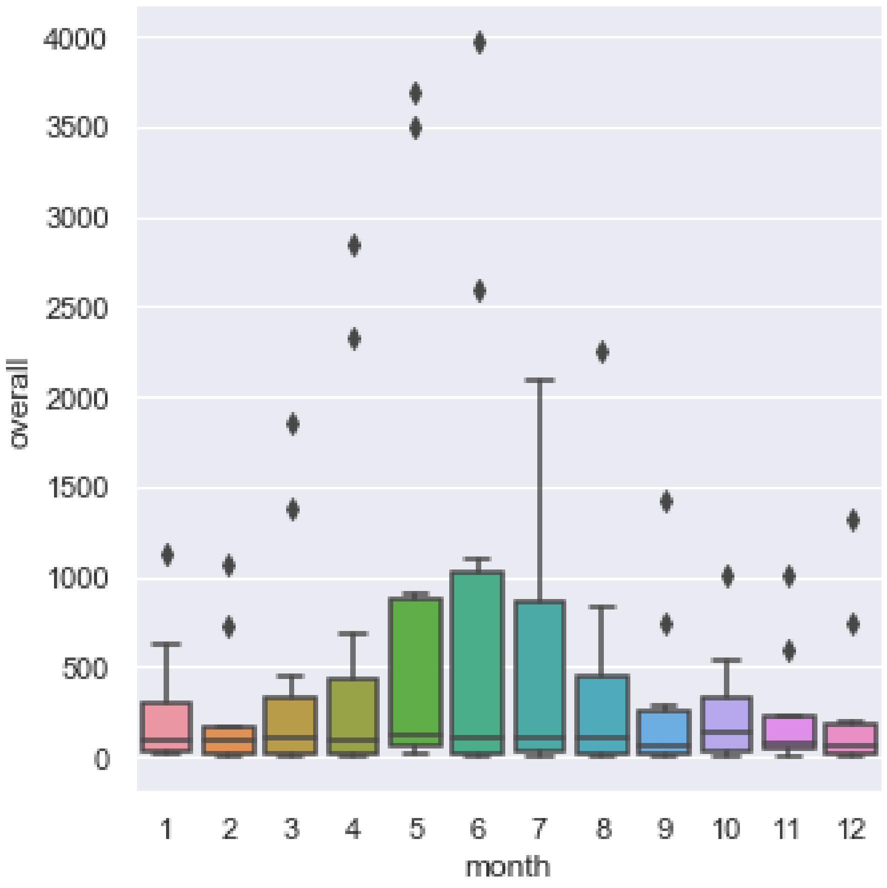
We can infer from above plot that

1) Users do not give bad ratings unless there is some real reason as we can observe that number products with rating '1' or rating '2'also rating '3' is significantly less over the years as compared to ratings '3' and '4'

2) From 2006 to 2013, there is increasing trend in total number of ratings, total number of good ratings (4,5).

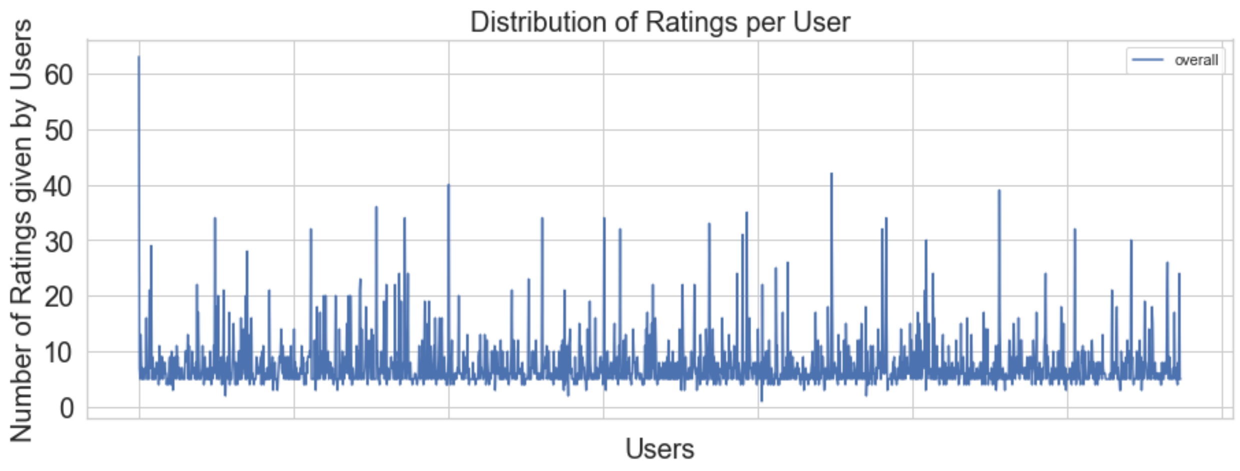
3) In 2013, there is a sudden surge of positive ratings and in 2014 it has decreased again.

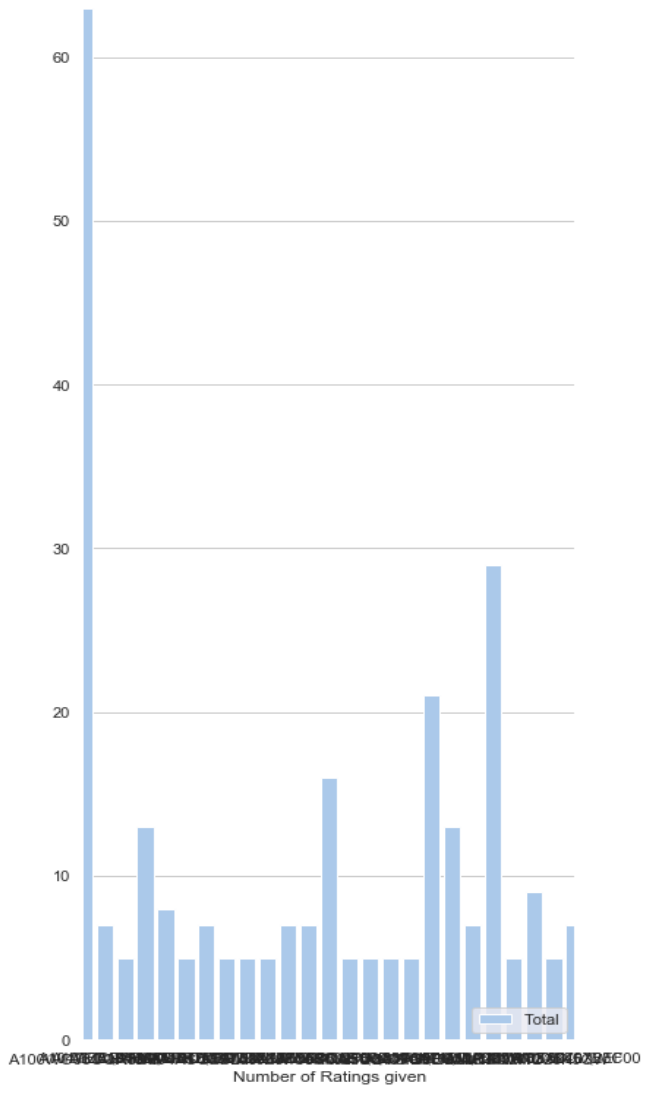
6. Distribution of ratings by month



From the boxplot above, we can say that highest number of ratings are given in the month of June followed by May and July. It indicates that in these three months, there is high volume of purchases done by the users. As against that February shows the lowest number of ratings given by users so on can infer that lowest sales might have happened in the month of February.

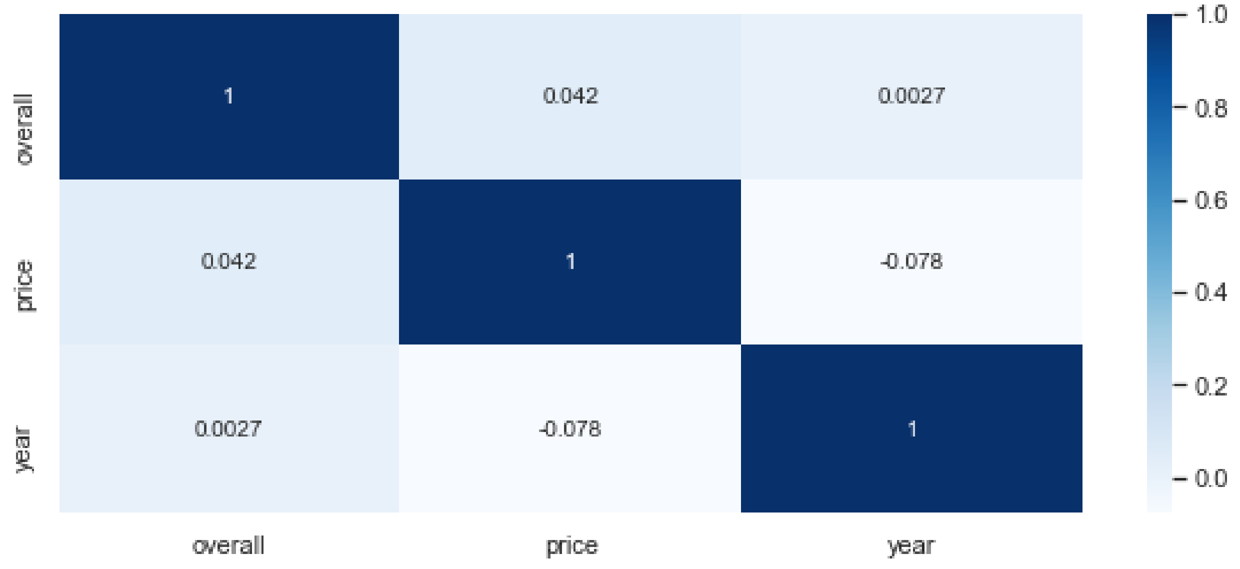
7. Distribution of ratings per user





Total number of users in Garden and Patio dataset is 1685. Maximum number of ratings given by single user is 63 and minimum number of ratings by single user is 1. On an average a user gives 7.55 ratings on Amazon as per the data present (till 2014).

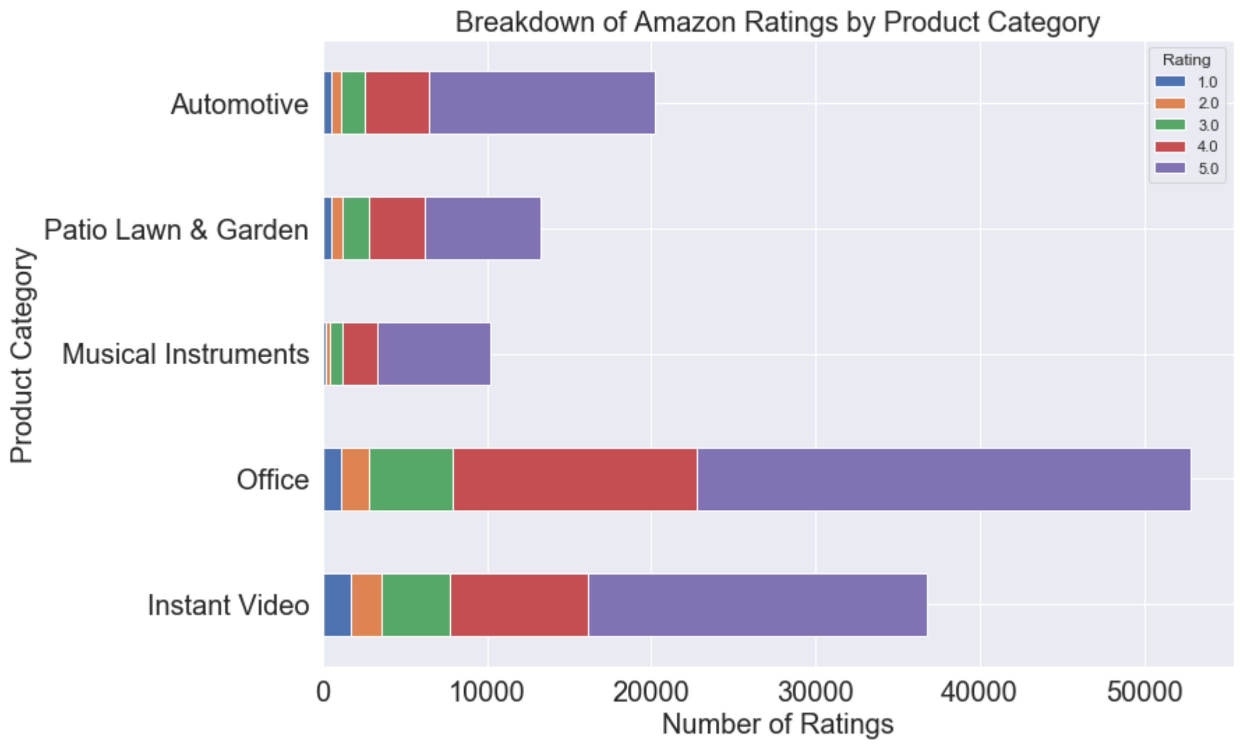
8. Heatmap: Checking for correlation between attributes in Garden and Patio dataset



Dark shades represent positive correlation while lighter shades represents negative correlation. Here correlation for numerical columns of dataframe is plotted as other columns are object type.

Here we can say that year and overall (Rating) have somewhat positive correlation whereas overall (Rating) and price has a positive correlation.

9. Breakdown of Amazon ratings by product category



I have used 5 different datasets from Amazon to plot above chart namely 'Automotive', 'Garden and Patio', 'Musical Instruments', 'Office' and 'Instant Video'. These are datasets for different categories of Amazon products. In general, 'Office category' products and 'Instant Video' seem to be more popular in these 5 categories. In all five categories, number of good ratings is more than number of bad ratings.

References

1) Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering

R. He, J. McAuley *WWW*, 2016 pdf

2) Image-based recommendations on styles and substitutes

J. McAuley, C. Targett, J. Shi, A. van den Hengel *SIGIR*, 2015, pdf

3) The Good, Bad and Ugly: Apache Spark for Data Science Work

https://thenewstack.io/the-good-bad-and-ugly-apache-spark-for-data-science-work/

4) Pythonic Data Cleaning With NumPy and Pandas

https://realpython.com/python-data-cleaning-numpy-pandas/

5) Data Cleaning with Python and Pandas: Detecting Missing Values

https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

6) Apache Spark

https://spark.apache.org/

7) Apache PySpark

https://spark.apache.org/docs/0.9.1/python-programming-guide.html

8) Recommender systems survey in Knowledge-Based Systems

Volume 46, July 2013, Pages 109-132

J.Bobadilla, F.OrtegaA., HernandoA.Gutiérrez

9) What is Collaborative Filtering?

https://dzone.com/articles/building-sales-recommendation-engine-with-apache-s

10) Collaborative Filtering Final Report

Tianyi Li, Pranav Nakate, Ziqian Song, Dr. Edward A. Fox Department of Computer Science, Virginia Tech

11) Amazon.com Recommendations: Item-to-Item Collaborative Filtering, IEEE Internet Computing, v.7 n.1, p.76-80, January 2003

Greg Linden, Brent Smith, Jeremy York