Product Recommender for online stores.

Capstone Project
Data Science Career Track, Springboard

Thanks to mentor Julian Jenkins III

How online stores can help customers to buy the right product from their millions of products,so that online retailers can retain the customer from going elsewhere and see yearly increase in their average order value. Build a recommendation system so that customers can get the right product with personalized info.

Online stores show millions of products to the customer from their catalog. Choosing the correct product for their needs is becoming difficult because of so much information.Since customers are more likely to buy based on personalized recommendations, management has decided to go for a recommendation system so that they can retain the customer and hence increase yearly product sales.

Data Source

In this project recommendation model is built based on electronics products of Amazon, based on the ratings.

The dataset here is taken from the below website.

Source - Amazon Reviews data (http://jmcauley.ucsd.edu/data/amazon/links.html) The repository has several datasets. For this case study, I am using the Electronics dataset.

Sample review dataset:
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). And has 1689188 records.

"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"

Data set had 1m+ rows and with no null values. out of 17 features only 4 features viz.,'reviewerId', 'productId', 'ratings','timestamp' are considered.Various types of distribution are looked at to get a better understanding of data.

Summary of data set

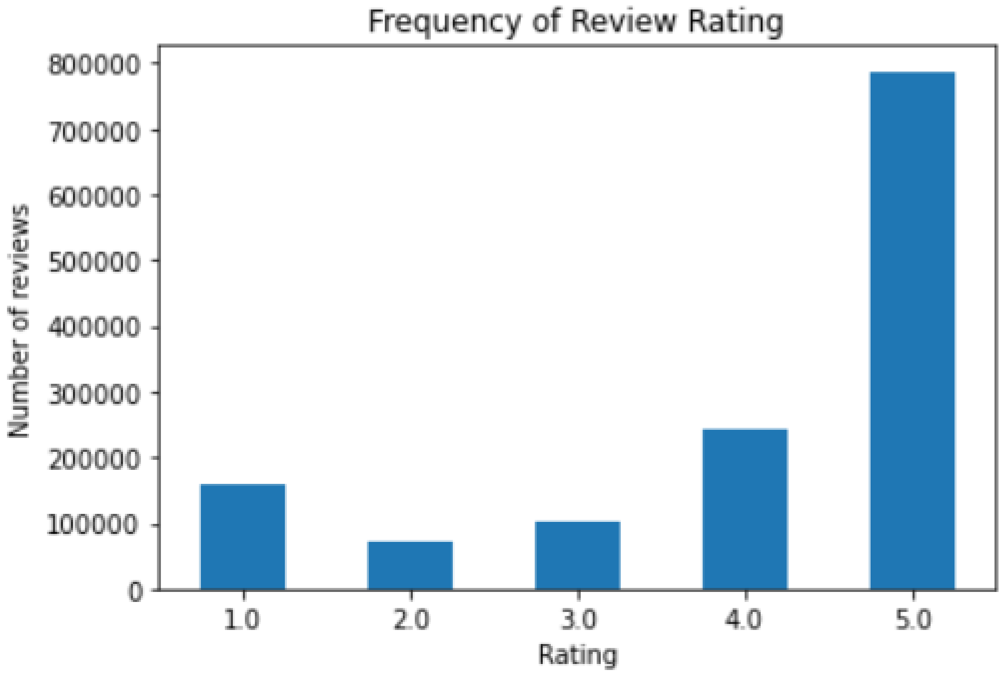
Number of reviews: 1365131
Number of unique reviewers = 1036895
Number of unique products = 35192
Average rating score: 4.046

Insights

Product Ratings:

Products were rated with discrete numbers 1,2,3,4,5. Most of the reviews rated 5. Here is the actual numbers

```
ratings
1.0    158473
2.0     72715
3.0    103776
4.0    242761
5.0    787406
```



Single Reviewer rated more than 50 times. This shows customers are buying many products from this store. Here is the top 5 number of times the reviewer has given the ratings.

```
reviewerId
A3OXHLG6DIBRW8    79
A2AY4YUOX2N1BQ    74
A680RUE1FDO8B     67
ADLVFFE4VBT8      57
A2NOW4U7W3F7RI    55
```

Popular products

The top 5 most popular product are -

B007WTAJTO with ratings 14162

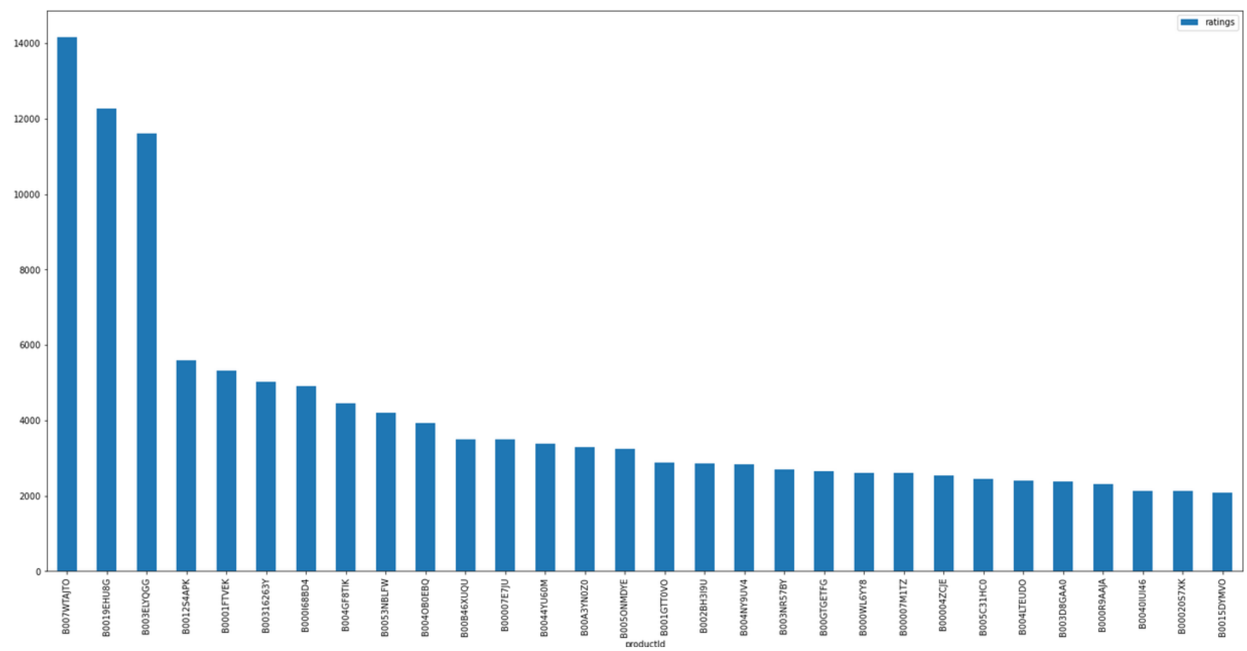
B0019EHU8G with ratings 12274

B003ELYQGG with ratings 11611

B0012S4APK with ratings 5606

B0001FTVEK with ratings 5324

Here is the distribution



Total Review Number based on years -

Number of reviews was maximum in the year 2013.Looks there were hardly ratings done on products till 2002 and slowly started from 2003.

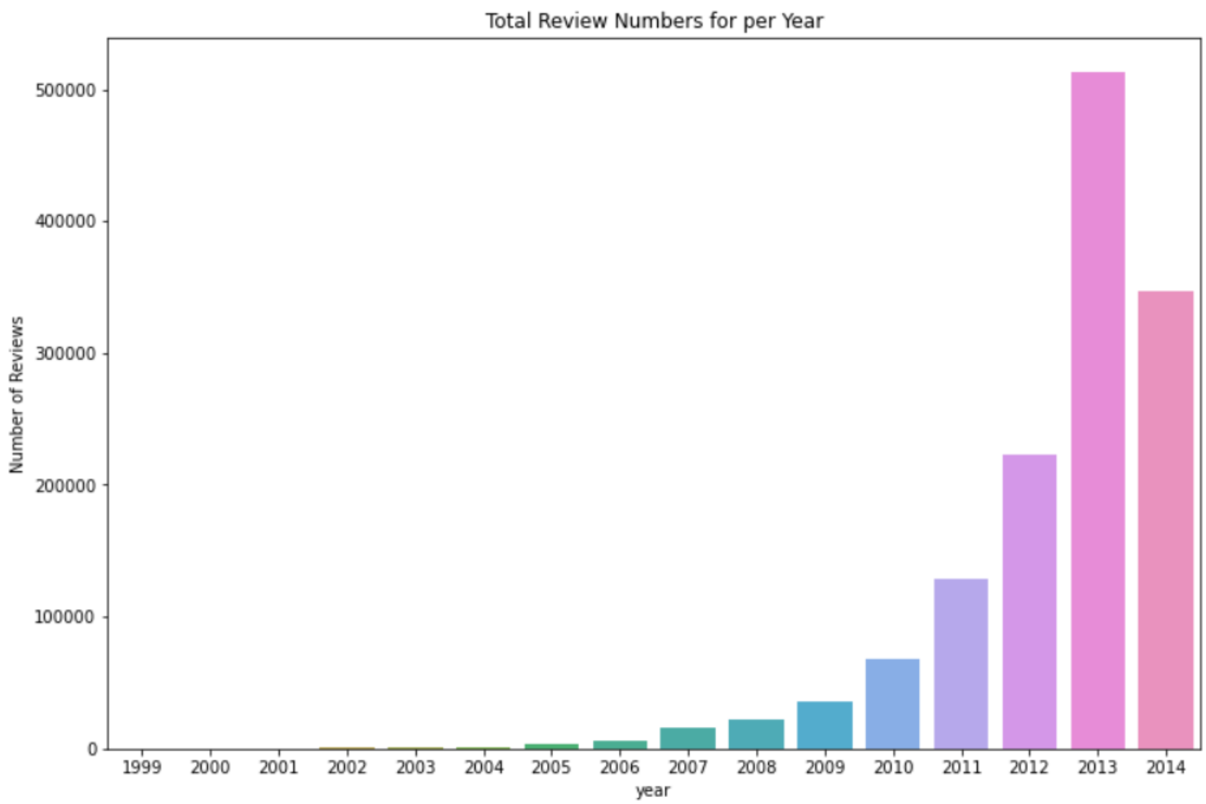
2013 with ratings 513513

2014 with ratings 346501

2012 with ratings 223490

2011 with ratings 128834

2010 with ratings 67371

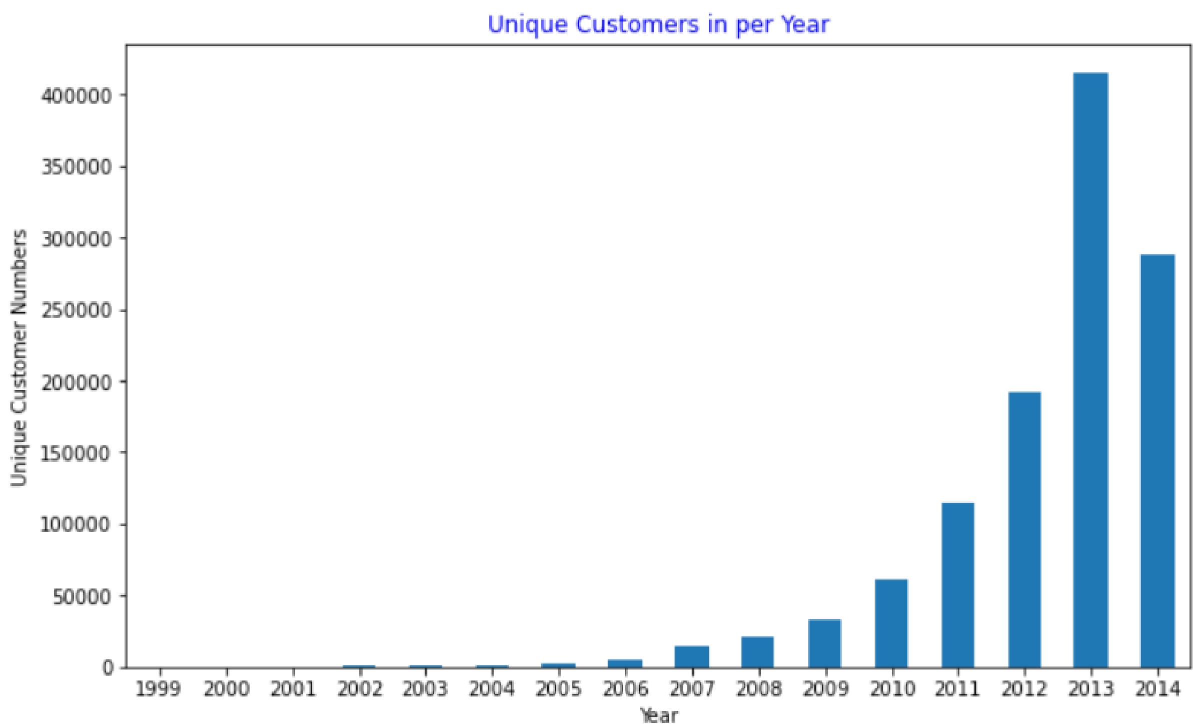


Unique customer per year:

2010 with customer 61782
2011 with customer 115009
2012 with customer 192277
2013 with customer 414734

2014 with customer 28818

There were a total of 1151618 unique customers visited between 1999 to 2014. Year 2013 has seen the maximum number. Here is the visual distribution



Unique Products per year

2010 has seen 7002 products

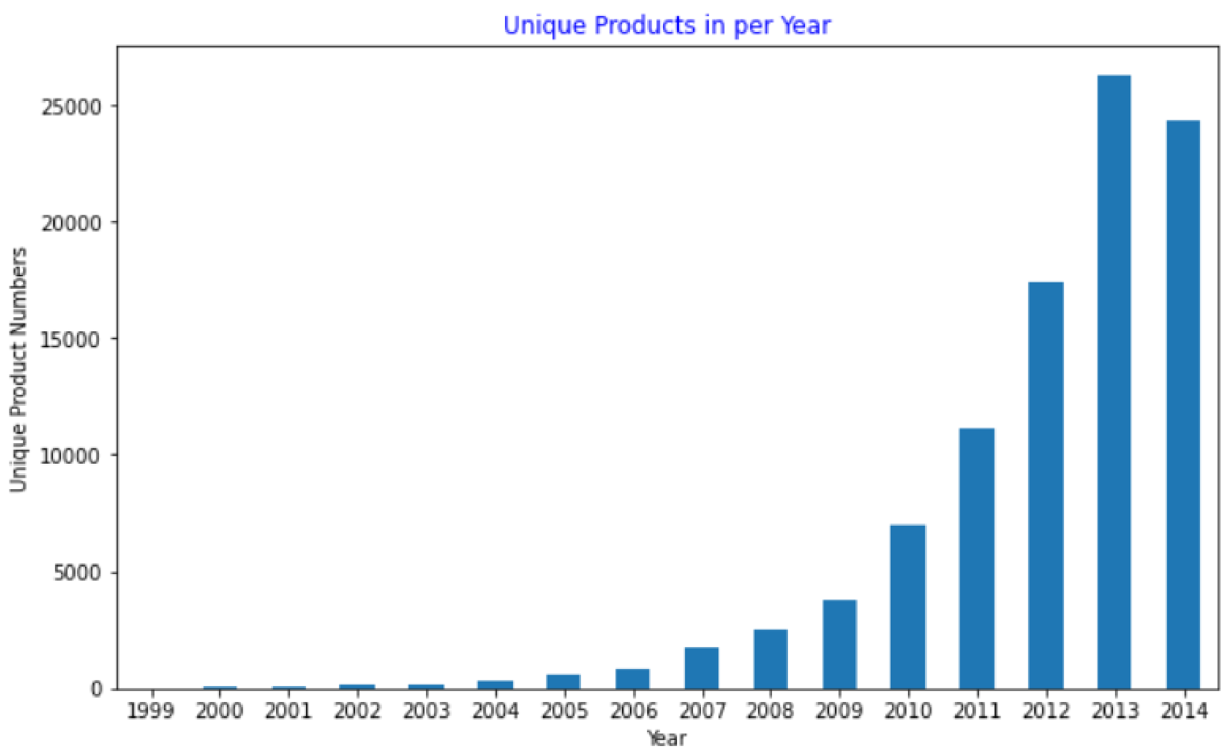
2011 has seen 11177 products

2012 has seen 17387 products

2013 has seen 26257 products

2014 has seen 24322 products

2013 has seen the more products.From the plot looks there were not many products till 2005.



From 1999 to 2004, not many reviews were given, maybe customers were buying less product, or reviews and ratings were not that significant for product purchase.

Model Selection :-

Surprise Package :http://surpriselib.com/

References

Documentation :https://surprise.readthedocs.io/en/latest/

Installation: http://surpriselib.com/
Git hub : https://github.com/NicolasHug/Surprise

There are a lot of different packages available to build a recommender system. For this one, I'm using the Surprise package. Surprise has many different algorithms built in.Itprovides various ready-to-use prediction algorithms such as baseline algorithms, neighborhood methods, matrix factorization-based (SVD, PMF, SVD++, NMF), and many others. Also, various similarity measures (cosine, MSD, pearson…) are built-in.

In this case, I need to load in a custom dataset to use with Surprise. According to the documentation, we need to make sure our data frame has three columns: the user ids, the item ids, and the ratings. Additionally, we'll need to specify the rating scale. In our case, users have used the ratings discretely from 1 to 5.

I'm also going to split the data into training and testing data using the Surprise package

With the Surprise library, I used below algorithms

BaselineOnly: Algorithm predicting the baseline estimate for a given user and item.

KNNBaseline:A basic collaborative filtering algorithm taking into account a *baseline* rating.
NMF:A collaborative filtering algorithm based on Non-negative Matrix Factorization.
Co-clustering:A collaborative filtering algorithm based on co-clustering.
SVD:When baselines are not used, this is equivalent to Probabilistic Matrix Factorization, it is as popularized by Simon Funk during the Netflix Prize

Metrics used are rmse and mae. And here is the result.

Algorithm	test_rmse	test_mae	fit_time	test_time
BaselineOnly()	1.291117	1.018094	5.764257	2.878079
SVD()	1.295390	1.018963	52.802452	3.046659
CoClustering()	1.445493	1.115695	83.513256	2.960003

BaselineOnly algorithm has given the best rmse. So I used this model to get the product recommendation based on ratings.

Top 10 best predictions

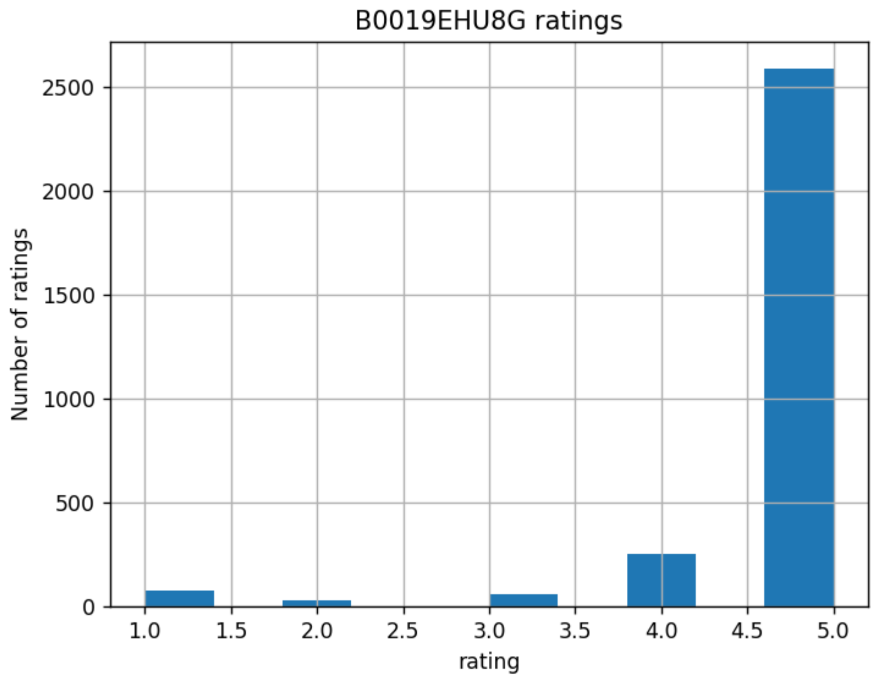
	reviewerId	productId	ratings	predicted Ratings	itemsCounts_byReviewer	reviwerCounts_forItem	err
107286	AUQKQ2Q0N5N79	B005DB6NG6	3.0	2.999966	1	27	0.000034
277341	A1JFEP3BI48Q33	B0094Z6N9Y	3.0	3.000041	0	176	0.000041
242103	A2AKFOQKKC1M2G	B0094Z6N9Y	3.0	3.000041	0	176	0.000041
307506	A1WQ0YSLJ06KFT	B0094Z6N9Y	3.0	3.000041	0	176	0.000041
202183	AOAVQTWTYP8C0	B0094Z6N9Y	3.0	3.000041	0	176	0.000041
40597	A2TQCR6GN8XSP6	B0094Z6N9Y	3.0	3.000041	0	176	0.000041
302422	A1CKWE4BW40LLG	B0094Z6N9Y	3.0	3.000041	0	176	0.000041
175043	AC5GLGVJXDEFV	B000IF51UQ	4.0	4.000056	2	512	0.000056
9586	AMIWD0PA1H1YB	B000HGHMF8	4.0	4.000077	5	26	0.000077
66049	A2SZZB6Y5X9T8M	B0028E1E2Y	4.0	4.000124	1	41	0.000124

The above are the top 10 best Predictions. The product 'B000HGHMF8','B000IF51UQ' and 'B0028E1E2Y' are rated best with ratings 4.

Top 10 worst predictions

	reviewerId	productId	ratings	predicted Ratings	itemsCounts_byReviewer	reviwerCounts_forItem	err
119632	A195EZSQDW3E21	B004J3V90Y	1.0	4.870298	19	1221	3.870298
8116	A227IMN9BB71BO	B0019EHU8G	1.0	4.880704	2	9265	3.880704
299570	AGR1V15L6FLMA	B007WTAJTO	1.0	4.894745	16	10683	3.894745
293699	A2QIX73WKLLDRV	B004Z4FBE2	1.0	4.902412	3	1154	3.902412
167309	ARXYNHFIXEYO2	B00BEW8MVC	1.0	4.909606	3	172	3.909606
333029	A1D9V11QUHXENQ	B004LSNF04	1.0	4.926395	6	1448	3.926395
241191	AC8YM4BB5LVOM	B0045JCFLY	1.0	4.954445	6	109	3.954445
168002	A6KL17KKN0A5L	B000JE7GPY	1.0	4.970054	11	1462	3.970054
311480	A3S3R88HA0HZG3	B008Q3CCZE	1.0	4.974388	7	152	3.974388
64010	A1Z16630QMH8Q6	B000E8BGCE	1.0	5.058453	13	164	4.058453

Cross checking with productId B007WTAJTO, this has predicted rating 4.89 and 10k reviewers rated this.



Conclusion:

There are many ways to build and improve this recommender system. I couldn't use k-NN algorithms from the Surprise package, as it needs more memory, nor could I complete it on google colab. Using more tuned parameters I did some basic cross validation to select the best parameters. To improve the recommendations, tune these parameters and make sure that the algorithm is serving the best recommendations possible with the data available.