Data Wrangling for Capstone-1: Amazon Product Recommender System by Uma Gajendragadkar

Date: May 31st 2019

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 ratings to recommend different products to current user.

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.

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