TOBB University of Economics and Technology
BIL 401 – Introduction to Big Data
Baran Özgenç
bozgenc@etu.edu.tr
bozgenc.b@gmail.com

Amazon Book Reviews Project

Abstract

The main goal of this project is to make interpretations from a raw dataset which provided by Amazon and containing book reviews which are made between years 1996 – 2018.

Introduction

This project report is divided into several categories. Problem explanation, dataset resource and characteristics, technologies used during development and project structure will be detailed in the upcoming chapters.

A Coarse Outline

This project aims concluding meaningful results via processing raw chunk of data which is accumulated over the years. Since the data is '*Big*' it is crucial to handle large amount of data efficiently. To do so it is necessary to move with the guidelines of fundamentals of big data processing. The results can be interpreted in many ways according the perspective of reviewing. In the upcoming parts you may find results such as most controversial books, top rated books, top disliked books etc.

Dataset Resource

Amazon review datasets can be downloaded from GitHub page of Jianmo Ni (Ph.D. in Computer Science Department University of California San Diego). You may find different categories of datasets related with Amazon reviews such as Electronics, Fashion etc. Also there is an option to download dense subsets which called k-core datasets where k (an integer) corresponds to limit of review number made by one customer.

In this project raw version of Book Reviews dataset is used. Format of the dataset is JSON and it is well structured. File size is approximately 37 GB and contains 51 millions records in it.
You may find more details about datasets by going to the address which is provided in the references section[1]. Please mind that it is compulsory to fill out a form to download datasets.

Technologies Used

Main programming language chosen was Scala due to high integration with Apache Spark. Apache Spark used for distributed data processing events and some of results are visualized by using Python3 and matplotlib library. IntelliJ is used as a development environment.

Configurations:

- Scala version: 2.12.15
- Spark version: 3.3.1

Hardware and OS:

- Project is built and run in a single device with
- 16 GB of ram (Spark driver memory is set to 12 GB.)
- Intel i7 processor.
- MacOS

Contents of the Dataset

The project has been developed on a single dataframe which is converted from raw dataset. The tables were fixed. There was no real-time processing. Whole project is based on batch processing. In the image below, you can see the one JSON object which corresponds to one row of the dataframe. The field '*asin'* is the terminological word used by Amazon and it corresponds to the unique *'id'* number.


```
{
  "reviewerID": "A2SUAM1J3GNN3B",
  "asin": "0000013714",
  "reviewerName": "J. McDonald",
  "vote": 5,
  "style": {
    "Format:": "Hardcover"
  },
  "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"
}
```

Details of the Computation

Computations divided into two categories: All times statistics (1996 - 2018) and yearly statistics. Total review count, user who made most reviews, most liked books, most disliked books, most controversial books, books which received most five stars, books which received most one star is calculated for both categories.

First of all, *drop-early* rule is applied to the initial dataframe to drop unnecessary columns and obtain a lightweight dataframe to speed-up computation process. Then all the computations has been made using this lightweight dataframe.

To achieve results, records are mostly grouped by *asin* column and then other calculations are made such as aggregating number of points received, or number of reviews received. To find some results such as book ratings or controversial books, dataframe is divided into several smaller dataframes then calculated results for each dataframe and results are joined.
When calculating yearly statistics, an additional year based filtering applied into *reviewTime* column then continued with the regular filtering which used in all times calculations.

A *getProductUrl(asin: String): String* function is used for obtaining product url from *asin* number.
Most of the calculation functions used were built-in *spark.sql* functions such as *desc*, *orderBy*, *groupBy* etc.

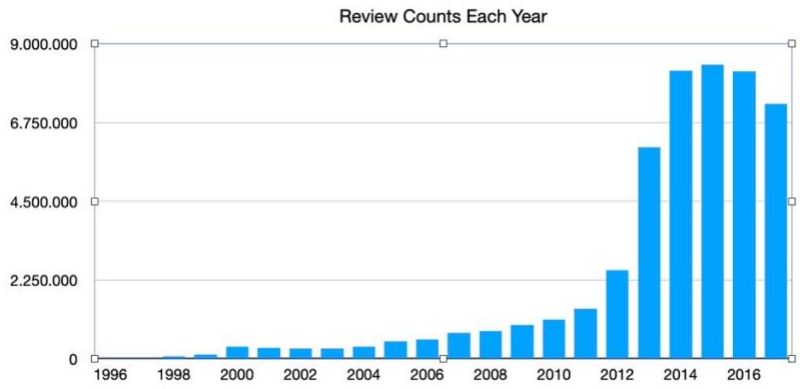
Results

In this chapter, for every different statistics, firstly the code part will be shown to obtain results, then the results will be displayed. You may find the project inside my personal GitHub repository. Link is provided in the references section[2].

Total review number made between 1996 – 2018:

```
val totalReviewNumber = lightweightDF.count();
println("All times total review number is: " + totalReviewNumber)
```

```
All times total review number is: 51311621
```



Finding top 10 users, who made most reviews between 1996-2018:

```
val userWhoReviewedMost = lightweightDF.groupBy(col("reviewerId"))
 .count()
 .withColumnRenamed("count", "users_review_number")
 .orderBy(desc("users_review_number"))
userWhoReviewedMost.show()
```

```
All times top ten users who made most reviews:
(Reviewer id: ,A2OJW07GQRNJUT, Review Count: 9684)
(Reviewer id: ,A2F6N60Z96CAJI, Review Count: 9074)
(Reviewer id: ,A328S9RN3U5M68, Review Count: 7077)
(Reviewer id: ,AHUT55E980RDR, Review Count: 5842)
(Reviewer id: ,A1X8VZWTOG8IS6, Review Count: 4437)
(Reviewer id: ,A2TX179XAT5GRP, Review Count: 4342)
(Reviewer id: ,A1K1JW1C5CUSUZ, Review Count: 4160)
(Reviewer id: ,A1D2C0WDCSHUWZ, Review Count: 4159)
(Reviewer id: ,A2VKWLCNZF4ZVB, Review Count: 4134)
(Reviewer id: ,A320TMDV6KCFU, Review Count: 3960)
```

And the customer name who made 9684 book reviews is:

```
+--------------+
|  reviewerName|
+--------------+
|Steven H Propp|
+--------------+
```

Finding the books which has been reviewed most between 1996 – 2018

```
val topBooksReviewed = lightweightDF.groupBy((col("asin")))
 .count()
 .withColumnRenamed("count", "products_review_count")
```

```
  .orderBy(desc("products_review_count"))
topBooksReviewed.show()
```

```
All times most reviewed ten books:
Review Count: 58150 Book Link: https://www.amazon.com/dp/038568231X
Review Count: 44956 Book Link: https://www.amazon.com/dp/0297859382
Review Count: 44381 Book Link: https://www.amazon.com/dp/0007420412
Review Count: 37783 Book Link: https://www.amazon.com/dp/0141353678
Review Count: 36620 Book Link: https://www.amazon.com/dp/0312577222
Review Count: 33676 Book Link: https://www.amazon.com/dp/0099911701
Review Count: 32495 Book Link: https://www.amazon.com/dp/B000X1MX7E
Review Count: 30297 Book Link: https://www.amazon.com/dp/0553418025
Review Count: 27954 Book Link: https://www.amazon.com/dp/0007548672
Review Count: 25713 Book Link: https://www.amazon.com/dp/0316055433
```

Finding the top 10 books which has most five star reviews:

```
   val bookWhoHasMostFiveStar = lightweightDF.filter(col("overall") === 5d)
  .groupBy(col("asin"))
  .count()
  .withColumnRenamed("count", "number_of_five_stars")
  .orderBy(desc("number_of_five_stars"))
bookWhoHasMostFiveStar.show(10)
```

```
All times top 10 books which reviewed 5 stars
https://www.amazon.com/dp/0312577222, Count:31426
https://www.amazon.com/dp/0007420412, Count:30158
https://www.amazon.com/dp/0141353678, Count:29818
https://www.amazon.com/dp/038568231X, Count:27330
https://www.amazon.com/dp/0099911701, Count:26160
https://www.amazon.com/dp/0553418025, Count:22639
https://www.amazon.com/dp/0007548672, Count:20031
https://www.amazon.com/dp/0297859382, Count:19212
https://www.amazon.com/dp/B017WJ5PR4, Count:18224
https://www.amazon.com/dp/B000X1MX7E, Count:18210
```

Finding the top 10 books which has most one star reviews:

```
  val bookWhoHasMostOneStar = lightweightDF.filter(col("overall") === 1d)
 .groupBy(col("asin"))
 .count()
 .withColumnRenamed("count", "number_of_one_stars")
 .orderBy(desc("number_of_one_stars"))
bookWhoHasMostOneStar.show(10)
```

```
All times top 10 books which received 1 star
https://www.amazon.com/dp/0297859382, Count:3547
https://www.amazon.com/dp/0007444117, Count:3456
https://www.amazon.com/dp/038568231X, Count:3264
https://www.amazon.com/dp/B000X1MX7E, Count:2752
https://www.amazon.com/dp/0316055433, Count:2453
https://www.amazon.com/dp/1616440724, Count:1907
https://www.amazon.com/dp/0751565350, Count:1784
https://www.amazon.com/dp/0345543254, Count:1598
https://www.amazon.com/dp/0062409883, Count:1541
https://www.amazon.com/dp/0545582881, Count:1428
```

Finding the top rated 10 books between 1996-2018:

```
  val booksAndTotalPoints = lightweightDF.groupBy(col("asin"))
 .agg(sum("overall").as("total_points"))
 .orderBy(desc("total_points"))
 .limit(100)

val booksAndReviewCounts = lightweightDF.groupBy(col("asin"))
 .count()
 .withColumnRenamed("count", "review_numbers")
 .orderBy(desc("review_numbers"))
 .limit(100)

val pointsAndCountsJoined = booksAndTotalPoints
```

```
 .join(booksAndReviewCounts, booksAndTotalPoints("asin") === booksAndReviewCounts("asin"), "inner")

val bookRatings = pointsAndCountsJoined
 .withColumn("ratings", $"total_points" / $"review_numbers")
 .orderBy(desc("ratings"))
bookRatings.show()
```

```
All times top ten books which has the highest scores:
Book Link: https://www.amazon.com/dp/0529120887 , Rating: 4.853354134165366
Book Link: https://www.amazon.com/dp/0141378247 , Rating: 4.849280404871106
Book Link: https://www.amazon.com/dp/0375969020 , Rating: 4.848265554943758
Book Link: https://www.amazon.com/dp/0007378033 , Rating: 4.789280806229959
Book Link: https://www.amazon.com/dp/0312577222 , Rating: 4.783670125614418
Book Link: https://www.amazon.com/dp/1683247353 , Rating: 4.7676488962932115
Book Link: https://www.amazon.com/dp/1503943372 , Rating: 4.767552602436323
Book Link: https://www.amazon.com/dp/0802407692 , Rating: 4.748671583789357
Book Link: https://www.amazon.com/dp/0143125478 , Rating: 4.747396968376193
Book Link: https://www.amazon.com/dp/0099664313 , Rating: 4.7194154160446296
```

Finding the top 10 most controversial books:

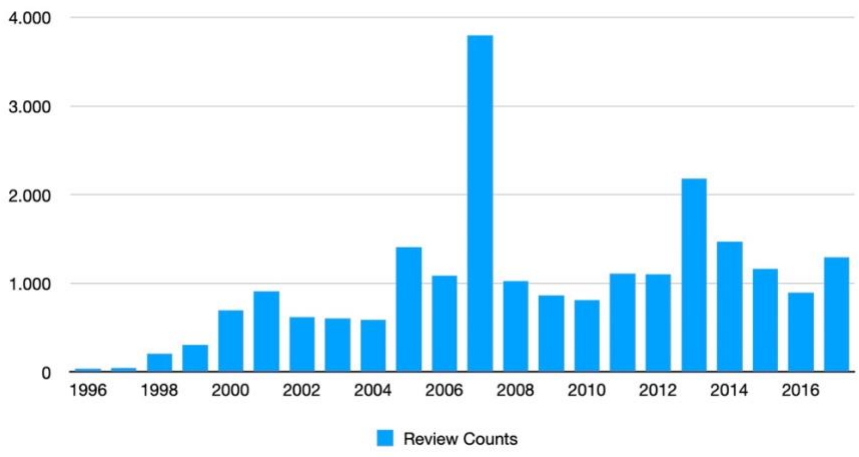
```
  val five = lightweightDF.filter(col("overall") === 5d).
 groupBy(col("asin"))
 .count()
 .withColumnRenamed("count", "five_stars")
```

```
val one = lightweightDF.filter(col("overall") === 1d)
 .groupBy(col("asin"))
 .count()
 .withColumnRenamed("count", "one_stars")

val others = lightweightDF.filter(col("overall") =!= 5d)
 .filter(col("overall") =!= 1d)
 .groupBy(col("asin"))
 .count()
 .withColumnRenamed("count", "other_points")

val tempDf = five.alias("five").join(one.alias("one"), five("asin") === one("asin"), "inner").drop(one("asin"))
val joined = tempDf.join(others.alias("others"), tempDf("asin") === others("asin"),
"inner").drop(others("asin"))

val expr = ($"five_stars" + $"one_stars") / $"other_points"
val controversials = joined.filter(col("five_stars") > col("other_points"))
 .filter(col("one_stars") > col("other_points"))
 .withColumn("ratio", expr)
 .orderBy(desc("ratio"))
```

```
Book Link: https://www.amazon.com/dp/1947814982,  Controversy Ratio : 310.0
Book Link: https://www.amazon.com/dp/0692873953,  Controversy Ratio : 157.5
Book Link: https://www.amazon.com/dp/0741426277,  Controversy Ratio : 132.0
Book Link: https://www.amazon.com/dp/1545109230,  Controversy Ratio : 120.5
Book Link: https://www.amazon.com/dp/0316387932,  Controversy Ratio : 119.0
Book Link: https://www.amazon.com/dp/0692480986,  Controversy Ratio : 110.5
Book Link: https://www.amazon.com/dp/9866895149,  Controversy Ratio : 109.0
Book Link: https://www.amazon.com/dp/1519140193,  Controversy Ratio : 109.0
Book Link: https://www.amazon.com/dp/0906138191,  Controversy Ratio : 102.0
Book Link: https://www.amazon.com/dp/0692795359,  Controversy Ratio : 98.0
```

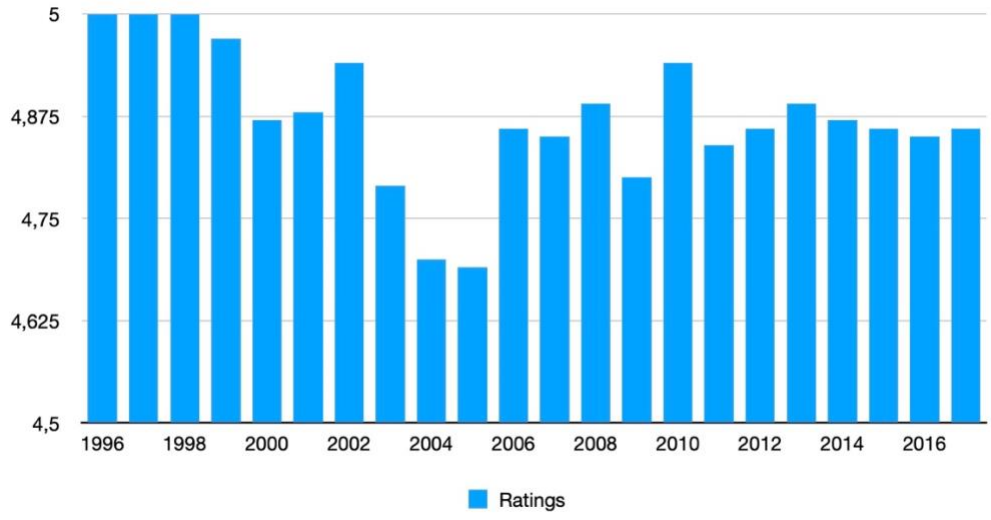
Customers who made most reviews each year

Reviewer ID	Review Counts	Reviewer ID	
1996	39	A3PPXVR56U2JD	
1997	46	A1PS66N855A04D,	
1998	205	AGHIVOVOON7MO	
1999	309	A3MQXFVZ9DCE8M	
2000	699	A1K1JW1C5CUSUZ	
2001	912	A1K1JW1C5CUSUZ	
2002	617	A222LQEPE707BV	
2003	601	A20DBHT4URXVXQ	
2004	593	AMXOPJKV4PPNJ	
2005	1.407	A1M8PP7MLHNBQB	
2006	1.086	A1M8PP7MLHNBQB	
2007	3.794	A1X8VZWTOG8IS6	
2008	1.023	A2TX179XAT5GRP	
2009	868	A2TX179XAT5GRP	
2010	815	A2TX179XAT5GRP	
2011	1.109	A20JW07GQRNJUT	
2012	1.103	A20JW07GQRNJUT	
2013	2.183	A20JW07GQRNJUT	
2014	1.468	A20JW07GQRNJUT	
2015	1.166	AHUT55E980RDR	
2016	897	AHUT55E980RDR	
2017	1.294	A2S1A56W8WG26T	



Top rated books each year

Reviewer ID	Ratings	ASIN	
1996	5	0045300232	
1997	5	034542705X	
1998	5	0151015392	
1999	4,97	0385333218	
2000	4,87	0385730640	
2001	4,88	0345465083	
2002	4,94	BOOOXUBF3I	
2003	4,79	0613171373	
2004	4,7	044652252X	
2005	4,69	0001844423	
2006	4,86	0739474464	
2007	4,85	0141034262	
2008	4,89	0446537527	
2009	4,8	0446549177	
2010	4,94	0061964549	
2011	4,84	0755361881	
2012	4,86	147674355X	
2013	4,89	0529120887	
2014	4,87	0989450252	
2015	4,86	006017322X	
2016	4,85	1501110365	
2017	4,86	0141378247	
2018	4,97	1979830282	



Controversy ratio is calculated by summing number of five stars and number of one stars and divided the sum to number of two, three and four stars. In other words, controversial books are the books which has most one and five star reviews than inbetween values.

Yearly statistics are calculated similar to all times statistics with one exception, which is applying year based filtering before regular calculations. Since there almost 20 times more output in the yearly statistics and since I do not want to make report unnecessarily longer, results of yearly statistics will not be displayed in the report. Instead of there is a slideshow inside the GitHub repository which contains all times and yearly statistics results. You may find the link which is provided in the references section[3].

References

[1] https://nijianmo.github.io/amazon/index.html
[2] https://github.com/bozgenc/amazon-book-reviews
[3] https://github.com/bozgenc/amazon-book-reviews/blob/master/Baran_Ozgenc_BIL_401_Project_Slideshow.key