Analysis

Basic setup

First, we need to import all the necessary libraries of python and spark in order to start with order analysis. The code snippet is attached below.

databricks Project_Amazon_Reviews (Python)

```
# import required libraries and functions
import requests
from pyspark.sql.functions import countDistinct, avg, stddev
from pyspark.sql import SparkSession
import pandas as pd
# plotting
import seaborn as sns
import matplotlib.pyplot as plt
sns.set(style="whitegrid")
from pandas import datetime
```

Fig 1. Importing Python and Spark Libraries

After importing the libraries, we import our data to data brick and transfer all the data to local database file system of data brick.

databricks Project_Amazon_Reviews (Python)

```
# download data from source
# data source: https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt

url = "https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_multilingual_US_v1_00.tsv.gz"
target_path = "amazon_reviews_multilingual_US_v1_00.tsv.gz"
response = requests.get(url, stream=True)
if response.status_code == 200:
    with open(target_path, 'wb') as f:
        f.write(response.raw.read())
else:
  print("Unable to download file")
```

```
%sh
gunzip amazon_reviews_multilingual_US_v1_00.tsv.gz
```

```
# moving data from driver node to dbfs file system
dbutils.fs.mv("file:/databricks/driver/amazon_reviews_multilingual_US_v1_00.tsv", "dbfs:/tmp/amazon_reviews_multilingual_US_v1_00.tsv")

Out[9]: True
```

Fig 2. Importing data to Databrick database file system

Once the data is available in the local file system of databrick, we can load the data into spark cluster easily and use the cluster for further analysis.

databricks   Project_Amazon_Reviews (Python)

```
# load data on spark cluster
df = spark.read.format("csv").option("header","true").option("inferSchema", "true").option("sep", "\t").load("dbfs:/tmp/amazon_reviews_multilingual_US_v1_00.tsv")
```

Fig 3. Transferring data from local database file system to spark

Pre-elementary check of data

Data size

We check the size of the data by using spark function called count. This will give us total number of records. There are 6.93 million rows.

```
# total no. of reviews
df.count()

Out[3]: 6931166
```

Fig 4. Total number of records using Spark count function

Null Value

This provide us insights on how many Null values are there in our datasets. There are 16 Null values in our data sets. Spark function isNull, count and when used.

```
# check for null values
from pyspark.sql.functions import isnull, when, count, col
display(df.select([count(when(isnull(c), c)).alias(c) for c in df.columns]))
```

marketplace	customer_id	review_id	product_id	product_parent	product_title	product_category	star_rating	helpful_votes	total_votes	vine	verified_purchase	review_headline
0	0	0	0	0	0	1	1	1	1	1	1	12

Fig 5. Displaying Null values of each columns

Exploratory data analysis

In this section we will explore different attributes of dataset and analyze them.

- Total number of different products : There are 52380 types of products in our data.

```
# Number of distinct product titles
df.select(countDistinct("product_title").alias("Distinct products")).show()

▸ (1) Spark Jobs
+-----------------+
|Distinct products|
+-----------------+
|            52380|
+-----------------+

Command took 58.95 seconds -- by chauhan.ku@northeastern.edu at 10/18/2019, 11:01:03 PM on test
```

Fig 6

- Total product categories: There are 38 product categories in our data.

```
# Number of different product caetegories
df.select(countDistinct("product_category").alias("Distinct product category")).show()
```

▸ (1) Spark Jobs

```
+-------------------------+
|Distinct product category|
+-------------------------+
|                       38|
+-------------------------+
```

Command took 57.09 seconds -- by chauhan.ku@northeastern.edu at 10/21/2019, 8:48:35 PM on test

Fig 7

- Total number of customers : 4112395

```
# total number of unique cutsomers
df.select(countDistinct("customer_id").alias("Number of Customers ")).show()
```

▸ (1) Spark Jobs

```
+--------------------+
|Number of Customers |
+--------------------+
|             4112395|
+--------------------+
```

Command took 58.66 seconds -- by chauhan.ku@northeastern.edu at 10/18/2019, 11:05:28 PM on test

Fig 8

Metric 1: Ratings

In this metric we have tried to explore the rating section of product through basic analysis explained below.

1. **Distribution of product ratings:** It has helped us to understand the product quality and the customer satisfaction. The maximum proportion of 5 star rating is 64% with a count of 4.44 million, followed by 4 star rating which means customers are highly satisfied by the products.

```
# distribution of star ratings
display(df.groupby('star_rating').count())
```

▸ (5) Spark Jobs

star_rating	count
5	4441940
4	1266311
3	536717
1	406653
2	279544

Fig 14

```
display(df.groupby('star_rating').count().orderBy('star_rating'))
```

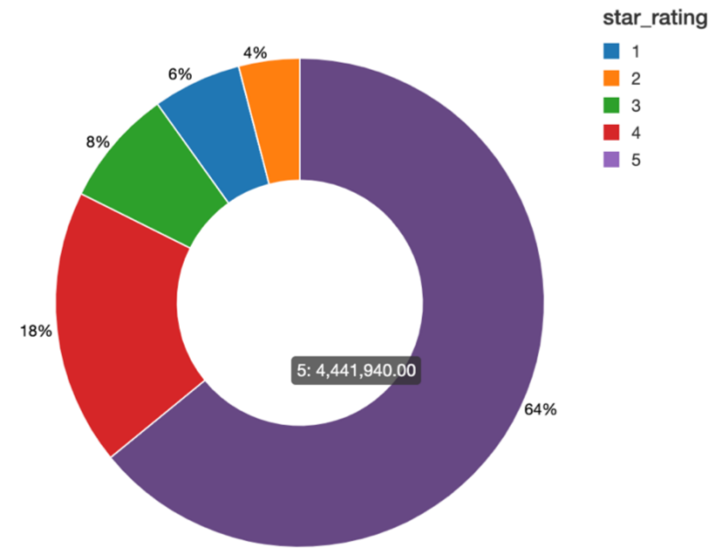

Fig 9. Distribution of rating in Pie chart.

We have used display function of python and count as well order by in spark to the desired result.

2. **Average star rating for category:** We computed the average star rating for all the product categories and sorted them high to low. We observed that “Automotive” products has the highest average rating which is 4.59/5.

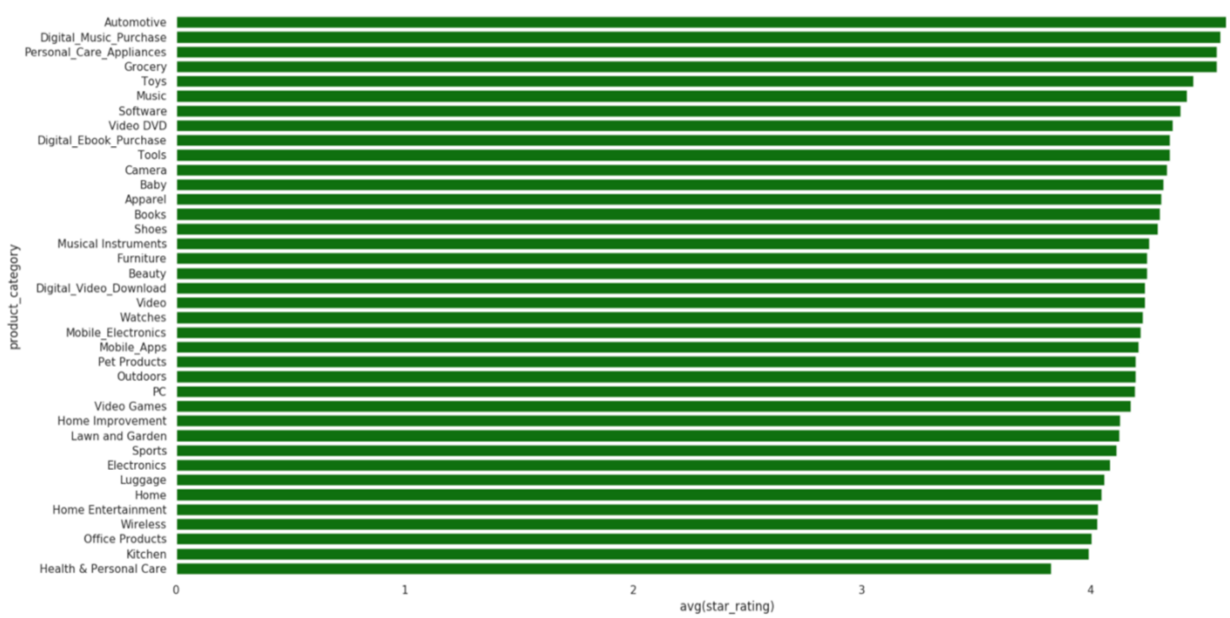

Fig 10. Average star rating of each category in amazon

We also sort the average rating in ascending order and found that Health & personal care rating is the minimum 3.83/5.

```
# average star rating for product category
display(category_avg_rating_df)
```

(2) Spark Jobs

product_category	avg(star_rating)
Health & Personal Care	3.830999066293184
Kitchen	3.9934888768312535
Office Products	4.008646779074795
Wireless	4.031939196907126
Home Entertainment	4.036797722045778

Fig 17

Metric 2: Reviews

This metric helps us to understand total reviews of each product, reviews for each category and most used words in reviews. We have dissected each part of review to analyze its effect on consumer behavior.

1. **Total review in each category:** This provides us insights on how many reviews over past 20 years has been written in each category, which product category is most popular among customers and which is less. We grouped the products category wise and count the reviews. Finally, we sorted them in ascending and descending order.
   Mobile apps got maximum reviews with a count of over 1 million and **Pet products** got only 5 reviews which is least among all product categories.
   These results could be of some highly popular product got more reviews which leads to increased reviews for product category.

```
# reviews by product_category
display(df.groupby('product_category').count())
```

(5) Spark Jobs

product_category	count
Mobile_Apps	1474583
Digital_Ebook_Purchase	1248890
Video DVD	1096886
Digital_Video_Download	1058097
Books	838800

Fig 18. Top 5 most reviewed product category

product_category	count
Pet Products	5
Furniture	8
Personal_Care_Appliances	9
Grocery	18
Beauty	52

Fig 19. Least reviewed product category

2. **Product with maximum review within a category:** We are interested to know which single product got maximum reviews and to which category does it belong. So we grouped our products with category and product title and sorted top 10 products which received maximum reviews in their category.

```
# Product with maximum reviews and its product category
display(df_product)
```

▸ (1) Spark Jobs

product_category	product_title	count
Books	Breaking Dawn (The Twilight Saga, Book 4)	5225
Books	The God Delusion	2332
Books	Steve Jobs	2126
Books	The Life-Changing Magic of Tidying Up: The Japanese Art of Decluttering and Organizing	1924
Books	Grey: Fifty Shades of Grey as Told by Christian (Fifty Shades of Grey Series)	1585
Books	The Hunger Games (The Hunger Games, Book 1)	1495
Books	Guns, Germs, and Steel: The Fates of Human Societies	1435
Baby	Fisher-Price Rainforest Jumperoo	1330
Books	A Feast for Crows (A Song of Ice and Fire, Book 4)	1186
Books	The Husband's Secret	1119

Fig 20. Product with maximum reviews

We observed that **"Breaking Dawn"** book got maximum reviews with a count of 5225, which is highest for a single product. Also, the top 10 most reviewed products belongs to Books category.
It is expected because Amazon started its business with books and in our data we have some new products which may have less reviews than books category.

3. **Growth in Number of reviews over 20 years:** This shows the **exponential trend** on increase in reviews. It also helps us to understand the growing importance of reviews.

```
# number of reviews over the 20 years
from pyspark.sql.functions import col, year
display(df.withColumn("review_date", year(col("review_date"))).groupBy("review_date").count().orderBy('review_date'))
```

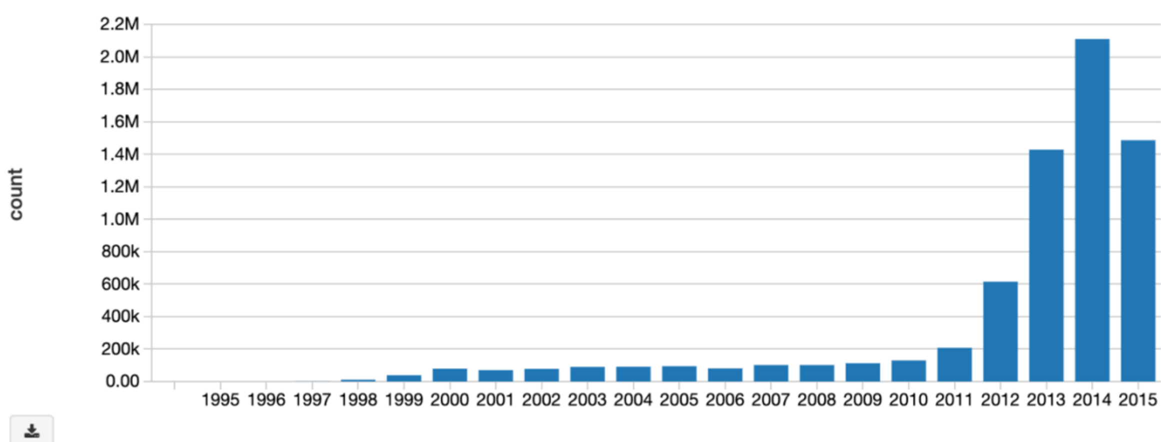

Fig 21: Total number of reviews in each year

We observed that the number of reviews in starting years were very few compared to succeeding years as the bars are missing for year 1995 to 1997 and very short until 2010. In this bar plot, we observed that for the last 5 years the growth in reviews is exponential. Also, amazon got maximum reviews in year 2014.
Also, the number of products are increasing on amazon's website over the years and customers are writing reviews for different products which is causing growth of reviews.

4. **Most frequent word used in writing review headline:** This has helped us to understand what words customer used in headline while writing a review. To know the most frequent words to appreciate or criticize the product in its review. To analyze text we perform text mining using NLP techniques and functions like Tokenizer, stopwordsRemover.
First, we tokenize the words in headline which are separated by space and punctuation. Then we remove the stop words which are common in all reviews otherwise words like "a, the, and" etc. would be the most frequent words which comes in almost all reviews. Then we created a vector in which the word and its frequency is stored and we sort these by the count high to low. We observed that the **stars** is the **most frequent** word with a count of approximately 1 million followed by **five.** This proves that customers are satisfied by the products and writing good reviews that's why the proportion of 5 star rating is high and even the words used in reviews are also five star.

```
#parsed_review_headline_text = df.rdd.map(lambda line: line['review_headline'].lower().split(' ') if line['review_headline'] else [])
#parsed_review_headline_text.take(5)
tokenizer = Tokenizer(inputCol="review_headline", outputCol="headline_words")
tokenized = tokenizer.transform(df.filter(df.review_headline.isNotNull()))
remover = StopWordsRemover(inputCol="headline_words", outputCol="headline_words_filtered")
filtered_set = remover.transform(tokenized)
```

```
counts = filtered_set.rdd.flatMap(lambda line:line['headline_words_filtered']) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
headline_word_counts = counts.collect()
```

```
# top 20 most frequent words in review headline
display(sorted(headline_word_counts, key= lambda x:x[1], reverse=True)[:20])
```

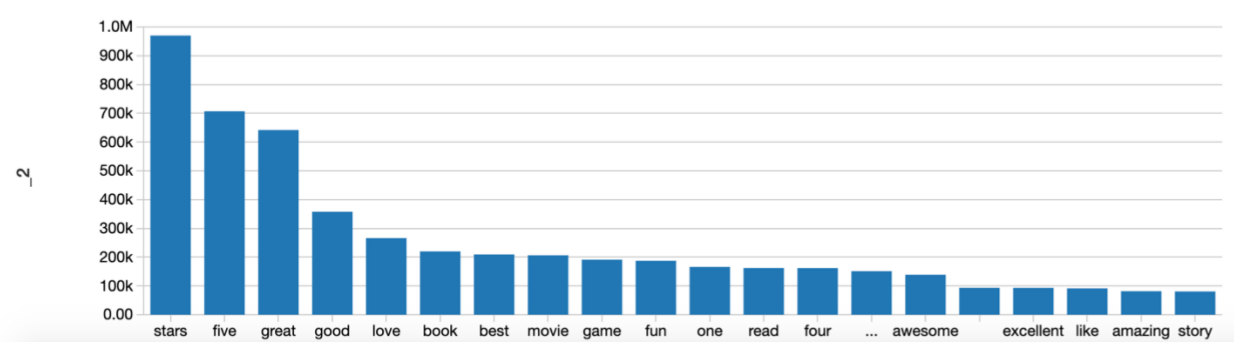

Fig 22. Most used words in review

We have used tokenizer function of spark to create a unique list of all words and then created a frequency table to plot the above graph.

5. **Distribution of length of reviews:** This helps us to understand customer likeliness of long review or short review or medium size review. It provides insights on customer preferences and how many words customer prefer while writing a review. We tokenized the words from the review text body and created a histogram of bin size 10. We plotted the histogram which shows that most of the reviews contains 20 - 30 words. Approximately 2 million reviews were written by customers in 20-30 words.

```
# distribution of length of words used by the customers in reviews
sns.set(rc={'figure.figsize':(16, 8)})
plt.hist(rdd_histogram_data[0][:-1], bins=rdd_histogram_data[0], weights=rdd_histogram_data[1])
display(plt.show())
```

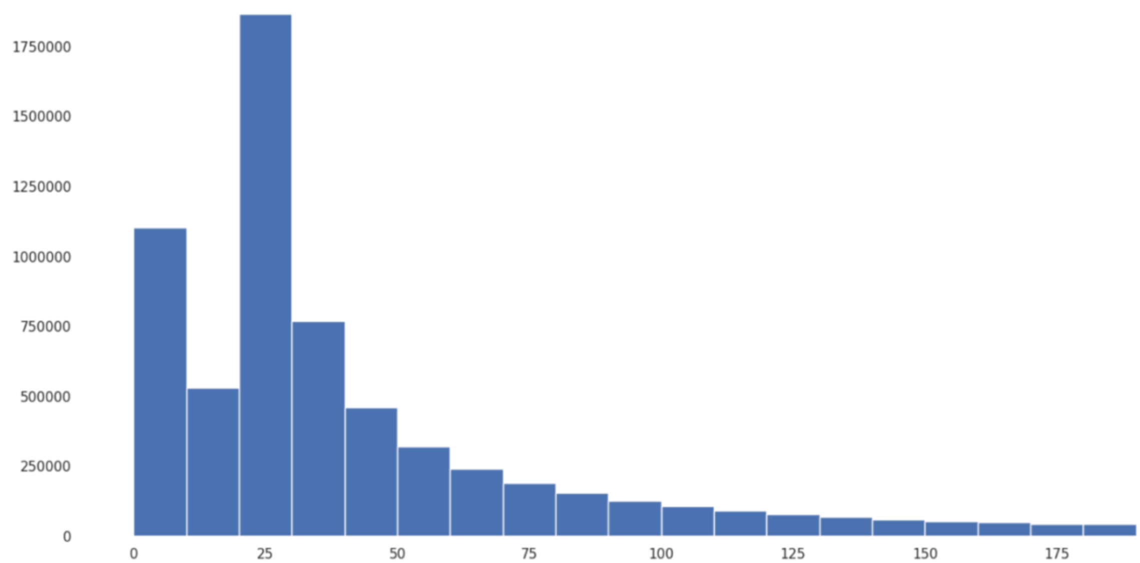

Fig 23. Length of reviews and its frequency

Metric 3: Helpfulness votes

This metric will help us to understand the intensity and impact of each review provided by the user across each product category. This metric is crucial in differentiating between genuine review vs fake review. On the basis of this metric we can blacklist user from the platform.

1. **Average votes for each category:** We have calculated the average of total votes and average helpful vote for each category to understand the which product category got maximum average total votes and maximum average helpful votes.

```
display(prod_cat_votes_sorted_df_20)
```

product_category	avg(total_votes)	avg(helpful_votes)
Mobile_Electronics	9.076086956521738	6.967391304347826
Video	6.967055549609333	3.9151022155624533
Books	6.0892584644730565	4.022758702908917
Video DVD	4.690281396608216	2.541025229604535
Music	4.437005027577838	2.7280228070963974
Office Products	3.8945092952875053	3.4198011240812796
Kitchen	3.252306022788931	2.8616386326641345
Mobile_Apps	3.2435875091466535	2.5243312855227544
Electronics	3.1278471070774065	2.3934025656689064
Health & Personal Care	3.118580765639589	2.6022408963585435

Fig 24. Top 10 helpfulness rating of category in Amazon

2. **Proportion of votes vs helpfulness vote:** This helps us to understand the total proportion of votes given to each category product compared to total helpfulness vote. This helps us to know how many reviews were actually helpful in each category in the platform. If the helpfulness vote is more than the category has product with more genuine reviews then the when compared to other. We chose only top 10 categories to plot the graph.
We observed that the Mobile Electronics category got maximum proportion of helful votes while in our previous analysis (fig 18) the number of count of mobile apps got the maximum number of reviews.

```
# Average votes and helpful votes proportion for top 10 most voted product category
sns.set(rc={'figure.figsize':(16, 8)})
#fig, axs = plt.subplots(1,2,sharex='col', sharey='row')
prod_cat_hfvotes_sorted_df = prod_cat_votes_df.sort_values(['avg(helpful_votes)'], ascending=False).reset_index(drop=True)
prod_cat_hfvotes_sorted_df_10 = prod_cat_hfvotes_sorted_df[:10]
prod_cat_votes_sorted_df_10 = prod_cat_votes_sorted_df[:10]
p1 = sns.barplot(x="avg(total_votes)", y="product_category", data=prod_cat_votes_sorted_df_10, label = "Total",color="blue")
#display(sns.despine())
p2 = sns.barplot(x="avg(helpful_votes)", y="product_category", data=prod_cat_hfvotes_sorted_df_10, label = "Total",color="green")
display(sns.despine())
```

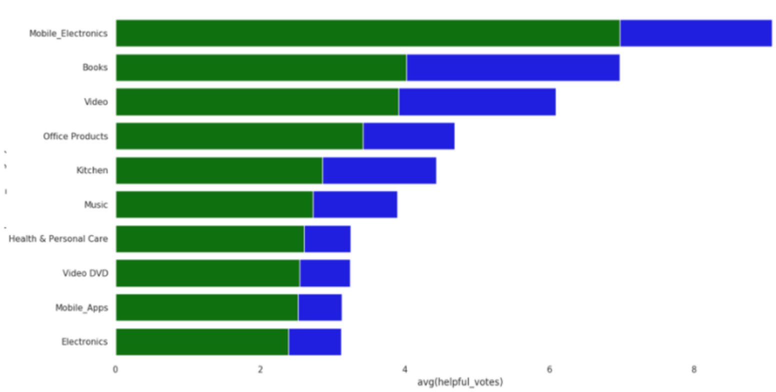

Fig 25. Proportion of helpful votes (green) vs total vote(blue) for top 10 category

Conclusion

In this project we had dealt with both the business as well as the technology problems in our approach. We took Amazon review and ratings data to understand its effect. We found out that user having negative experience with a product writes longer review while user having positive experience provides shorter review of the product. The length of review provides us with overview of user experience. The initial stage of project involved understanding the data, identifying problem statement, asking question around problem statement and creating KPI metrics on the question asked which provided us deep insights on the data which might not have been discovered during routine analysis. Reviews and ratings have gained sufficient traction in customer support segments to improve customer support experience in the organization. We can also perform sentiment analysis on the text written by customer in reviews and can build a classifier that classifies the sentiments into positive, negative and neutral categories.

In the technology aspect, we understood the underlying framework of Azure and its services such as databricks which leverages Spark and python to its full potential. Working on such a massive dataset was not an easy task but with Apache spark we got our results in seconds. We could also use a cluster of high configuration and more CPUs to get quicker results. The easiness in deployment with variety of tracking tool to understand the consumption of resources is useful to any organization who plans to develop strong architecture for the company across all the business unit. Apache Spark made our computation and aggregation easy as it took less time to execute and provided us result by using minimum resources. We explored the batch operation used by spark to run parallel operations in Azure while Python acted as intermediate software providing us with capabilities such as visualization and manipulating of dataset in spark.

In this project, we identified three main metrics which were Ratings, Reviews and Helpfulness votes which gave us insights on customer thinking, quality of products in Amazon and impact of reviews in influencing decision making. The current project explores the various exploratory analysis which leverages the qualitative features of data and exploits various aspect of it to identify new arenas for analysis. The future scope will be do create a model which will predict the behavior pattern of a customer and will show product recommendation accordingly. If the consumer decision is not affected by reviews, then the website can show products with good rating which has no or less reviews. The analysis can be seen as a compliment data for recommendation machine learning algorithms and improve its efficiency to a certain extent.

References

1. Woolf, M. (2017, January 2). Playing with 80 Million Amazon Product Review Ratings Using Apache Spark. Retrieved from https://minimaxir.com/2017/01/amazon-spark/.

2. (n.d.). Retrieved from https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt.

3. Azure Databricks Quickstart. (n.d.). Retrieved from https://docs.azuredatabricks.net/getting-started/quick-start.html.

4. Quick Start. (n.d.). Retrieved from https://spark.apache.org/docs/latest/quick-start.html.