CIS 4130 CMWA[17198]

Analysis and Prediction using Amazon Reviews Datasets

Ahmed Elsayegh: ahmed.elsayegh@baruchmail.cuny.eduy.edu

This dataset contains detailed customer reviews for various products sold on Amazon. It's a rich source of information, encompassing not just the text of the reviews but also a range of metadata. The data, which can be accessed from https://www.kaggle.com/datasets/cynthiarempel/amazon-us-customer-reviews-dataset, includes aspects such as customer ratings, votes for reviews, and verification status of purchases.

The dataset contains the following attributes:

Customer ID
Review ID
Product ID
Product Title
Product Category
Star Rating
Helpful Votes
Total Votes
Verified Purchase Status
Review Headline and Body
Review Date

With this dataset, our primary goals are as follows:

Trend Analysis: We aim to uncover patterns in customer reviews, exploring how ratings and review content vary across different products and over time. This analysis will shed light on consumer preferences and the evolution of product reception.

Popularity Prediction: Developing a model to predict the popularity of products based on review characteristics. Understanding the factors that lead to higher customer ratings can guide manufacturers and sellers in improving product quality and marketing strategies.

Insightful Review Analysis: By delving deep into the review text, we plan to extract insights that go beyond mere ratings. This will involve sentiment analysis and trend identification, providing a nuanced understanding of customer opinions.

Recommendation System Development: Leveraging the rich data available, our goal is to build a recommendation system. This system would suggest products to users based on their preferences and past review patterns.

Expected Outcomes

Visualization of Trends: Clear and insightful visualizations that map out customer preferences and review trends over time.
Predictive Model for Product Popularity: A machine learning model capable of accurately predicting the popularity of products based on review data.
Actionable Insights for Sellers and Manufacturers: Providing valuable information that can help in optimizing product features and marketing tactics.
Enhanced Recommendation System: A robust system capable of making personalized product recommendations to enhance the customer shopping experience.

Data Acquisition

During the stages of our project our main focus was, on utilizing the Spotify data. Although this dataset provided insights and a wide range of data points it presented challenges that needed to be reconsidered. The big size and complexity of the dataset add some hurdles in terms of manipulating the data processing speed and storage capacity. Moreover the nature of the dataset required preprocessing, which had the potential to extend our project timeline. Keeping these limitations in mind and aiming to maintain efficiency and feasibility I made a decision to shift the attention towards the Amazon US customer reviews dataset. This alternative dataset is still comprehensive but more streamlined and manageable allowing for progress, in our tasks within a reasonable timeframe.

1. **Setting Up Prerequisites**

Kaggle API Key:

Description:

- To utilize the Kaggle command-line interface on the EC2 instance, you need the Kaggle API key.

Instructions:

- Obtain the key, which is typically a kaggle.json file, from the Kaggle website under the account settings.
- Upload this file to the EC2 instance.

AWS CLI:

Description:

- Facilitates file transfers to Amazon S3.

Instructions:

- Ensure AWS CLI is installed and configured.
- The CLI should have the necessary permissions for S3 uploads.

Necessary Tools:

Description:

- Tools required for data extraction and processing.

Instructions:

- Ensure unzip is installed on the system.

2. Downloading the Kaggle Dataset

Instructions:

- Use the command below to fetch the dataset:

“kaggle datasets download -d cynthiarempel/amazon-us-customer-reviews-dataset”

3. Listing Filenames Inside the Zip

Instructions:

- To view the files within the zipped dataset, use: “unzip -l amazon-us-customer-reviews-dataset.zip”

4. Data Extraction and Upload

Unzipping Files:

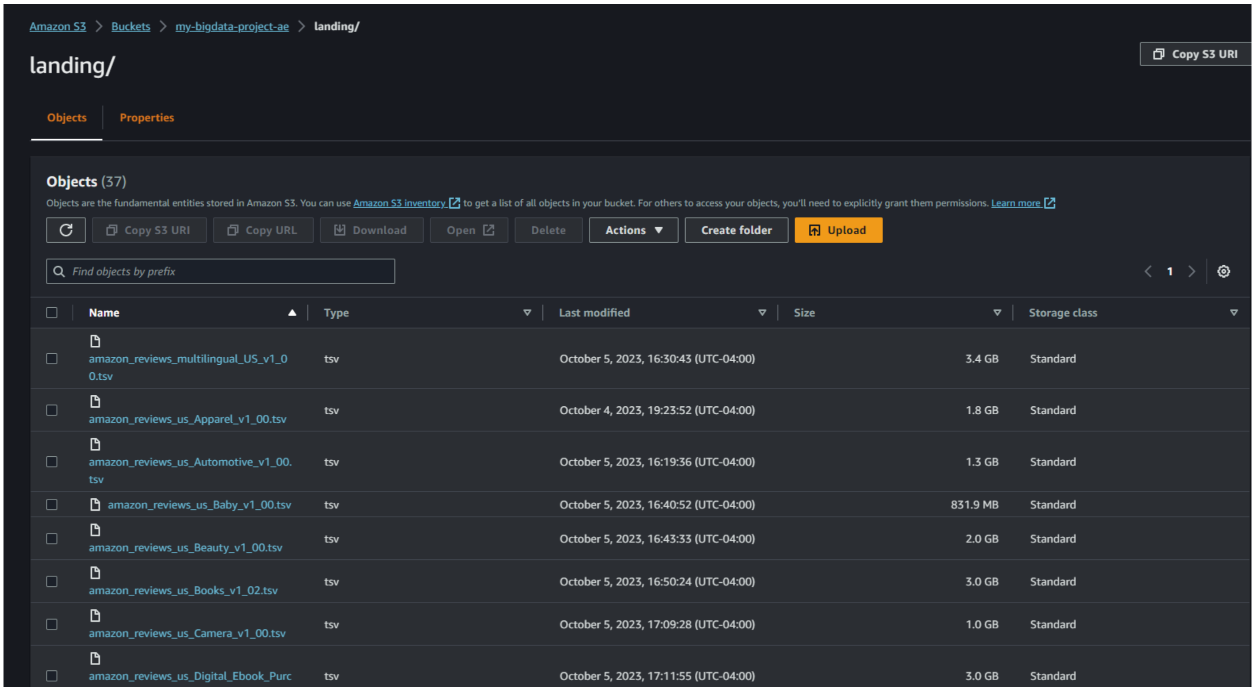
Description:

- Extract individual data files from the zipped dataset. For amazon_reviews_us_Apparel_v1_00.tsv, use: “unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Apparel_v1_00.tsv”

Copying to S3:

Description:

- Transfer the extracted dataset to Amazon S3. “aws s3 cp amazon_reviews_us_Apparel_v1_00.tsv s3://amazon-reviews-ae/landing/amazon_reviews_us_Apparel_v1_00.tsv”

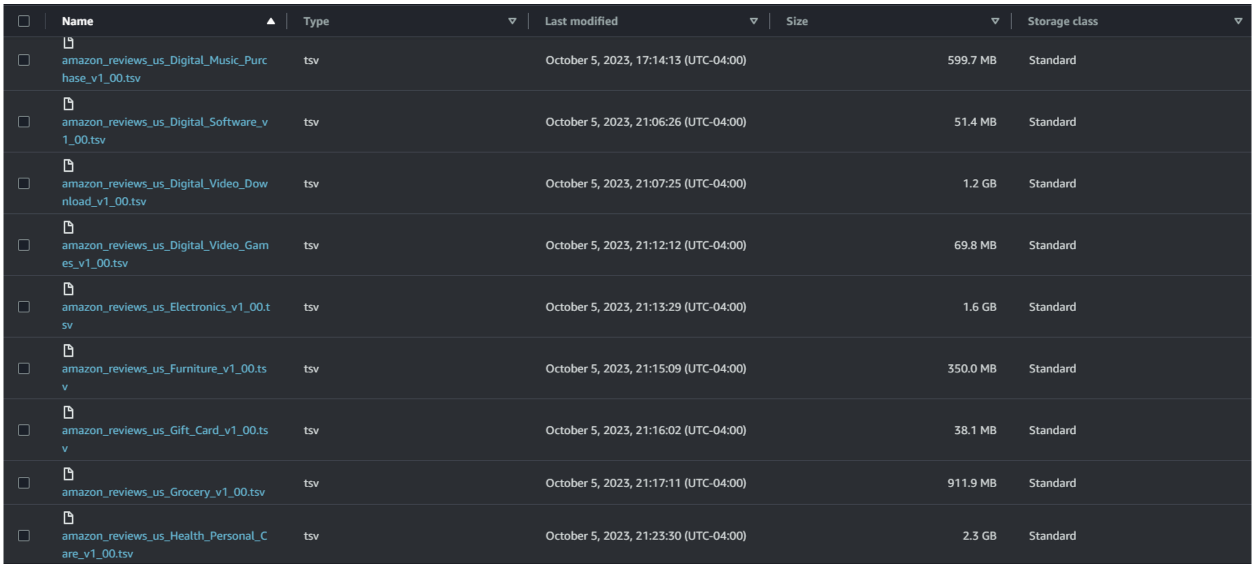


landing/

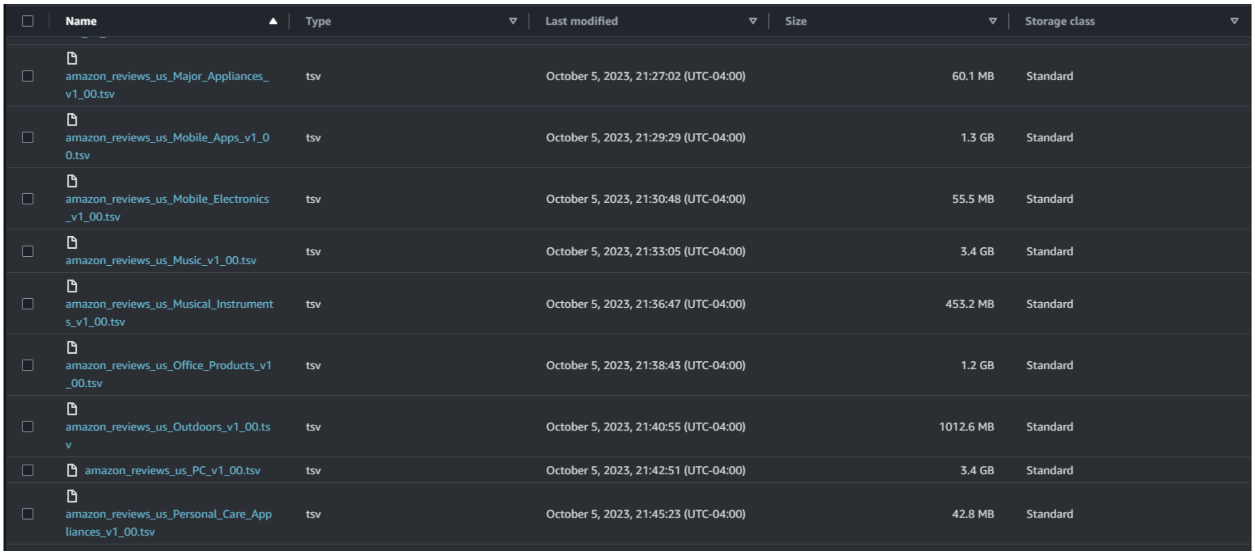
Objects (37)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

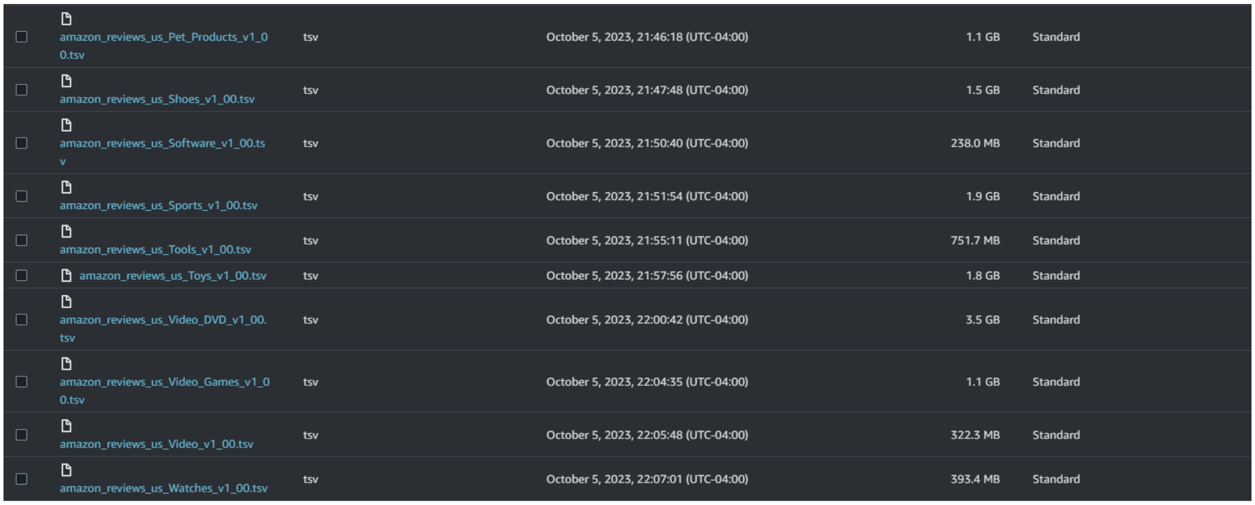
□	Name	Type	Last modified	Size	Storage class
□	amazon_reviews_multilingual_US_v1_00.tsv	tsv	October 5, 2023, 16:30:43 (UTC-04:00)	3.4 GB	Standard
□	amazon_reviews_us_Apparel_v1_00.tsv	tsv	October 4, 2023, 19:23:52 (UTC-04:00)	1.8 GB	Standard
□	amazon_reviews_us_Automotive_v1_00.tsv	tsv	October 5, 2023, 16:19:36 (UTC-04:00)	1.3 GB	Standard
□	amazon_reviews_us_Baby_v1_00.tsv	tsv	October 5, 2023, 16:40:52 (UTC-04:00)	831.9 MB	Standard
□	amazon_reviews_us_Beauty_v1_00.tsv	tsv	October 5, 2023, 16:43:33 (UTC-04:00)	2.0 GB	Standard
□	amazon_reviews_us_Books_v1_02.tsv	tsv	October 5, 2023, 16:50:24 (UTC-04:00)	3.0 GB	Standard
□	amazon_reviews_us_Camera_v1_00.tsv	tsv	October 5, 2023, 17:09:28 (UTC-04:00)	1.0 GB	Standard
□	amazon_reviews_us_Digital_Ebook_Purc	tsv	October 5, 2023, 17:11:55 (UTC-04:00)	3.0 GB	Standard



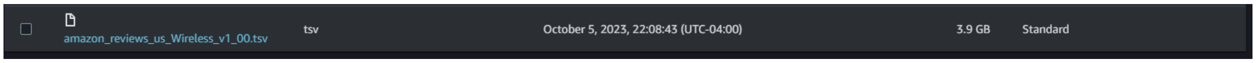
□	Name ▲	Type ▽	Last modified ▽	Size ▽	Storage class ▽
□	amazon_reviews_us_Digital_Music_Purchase_v1_00.tsv	tsv	October 5, 2023, 17:14:13 (UTC-04:00)	599.7 MB	Standard
□	amazon_reviews_us_Digital_Software_v1_00.tsv	tsv	October 5, 2023, 21:06:26 (UTC-04:00)	51.4 MB	Standard
□	amazon_reviews_us_Digital_Video_Download_v1_00.tsv	tsv	October 5, 2023, 21:07:25 (UTC-04:00)	1.2 GB	Standard
□	amazon_reviews_us_Digital_Video_Games_v1_00.tsv	tsv	October 5, 2023, 21:12:12 (UTC-04:00)	69.8 MB	Standard
□	amazon_reviews_us_Electronics_v1_00.tsv	tsv	October 5, 2023, 21:13:29 (UTC-04:00)	1.6 GB	Standard
□	amazon_reviews_us_Furniture_v1_00.tsv	tsv	October 5, 2023, 21:15:09 (UTC-04:00)	350.0 MB	Standard
□	amazon_reviews_us_Gift_Card_v1_00.tsv	tsv	October 5, 2023, 21:16:02 (UTC-04:00)	38.1 MB	Standard
□	amazon_reviews_us_Grocery_v1_00.tsv	tsv	October 5, 2023, 21:17:11 (UTC-04:00)	911.9 MB	Standard
□	amazon_reviews_us_Health_Personal_Care_v1_00.tsv	tsv	October 5, 2023, 21:23:30 (UTC-04:00)	2.3 GB	Standard



□	Name ▲	Type ▽	Last modified ▽	Size ▽	Storage class ▽
□	amazon_reviews_us_Major_Appliances_v1_00.tsv	tsv	October 5, 2023, 21:27:02 (UTC-04:00)	60.1 MB	Standard
□	amazon_reviews_us_Mobile_Apps_v1_00.tsv	tsv	October 5, 2023, 21:29:29 (UTC-04:00)	1.3 GB	Standard
□	amazon_reviews_us_Mobile_Electronics_v1_00.tsv	tsv	October 5, 2023, 21:30:48 (UTC-04:00)	55.5 MB	Standard
□	amazon_reviews_us_Music_v1_00.tsv	tsv	October 5, 2023, 21:33:05 (UTC-04:00)	3.4 GB	Standard
□	amazon_reviews_us_Musical_Instruments_v1_00.tsv	tsv	October 5, 2023, 21:36:47 (UTC-04:00)	453.2 MB	Standard
□	amazon_reviews_us_Office_Products_v1_00.tsv	tsv	October 5, 2023, 21:38:43 (UTC-04:00)	1.2 GB	Standard
□	amazon_reviews_us_Outdoors_v1_00.tsv	tsv	October 5, 2023, 21:40:55 (UTC-04:00)	1012.6 MB	Standard
□	amazon_reviews_us_PC_v1_00.tsv	tsv	October 5, 2023, 21:42:51 (UTC-04:00)	3.4 GB	Standard
□	amazon_reviews_us_Personal_Care_Appliances_v1_00.tsv	tsv	October 5, 2023, 21:45:23 (UTC-04:00)	42.8 MB	Standard



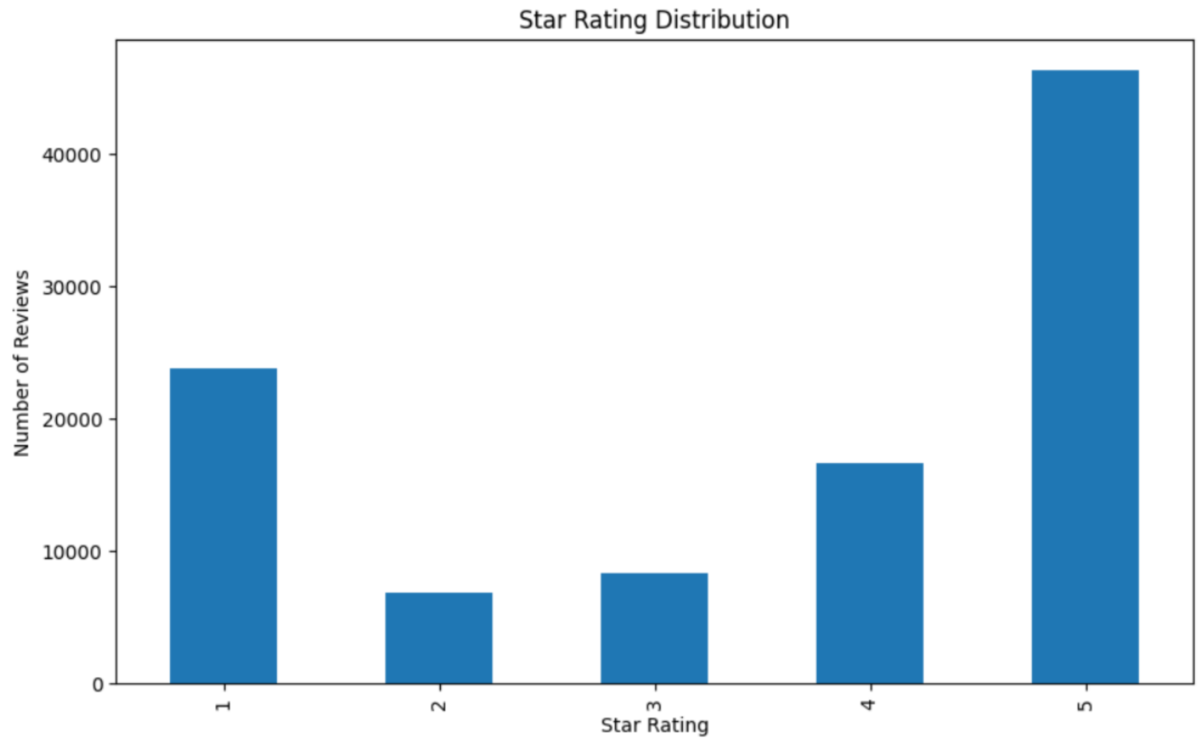
□	amazon_reviews_us_Pet_Products_v1_00.tsv	tsv	October 5, 2023, 21:46:18 (UTC-04:00)	1.1 GB	Standard
□	amazon_reviews_us_Shoes_v1_00.tsv	tsv	October 5, 2023, 21:47:48 (UTC-04:00)	1.5 GB	Standard
□	amazon_reviews_us_Software_v1_00.tsv	tsv	October 5, 2023, 21:50:40 (UTC-04:00)	238.0 MB	Standard
□	amazon_reviews_us_Sports_v1_00.tsv	tsv	October 5, 2023, 21:51:54 (UTC-04:00)	1.9 GB	Standard
□	amazon_reviews_us_Tools_v1_00.tsv	tsv	October 5, 2023, 21:55:11 (UTC-04:00)	751.7 MB	Standard
□	amazon_reviews_us_Toys_v1_00.tsv	tsv	October 5, 2023, 21:57:56 (UTC-04:00)	1.8 GB	Standard
□	amazon_reviews_us_Video_DVD_v1_00.tsv	tsv	October 5, 2023, 22:00:42 (UTC-04:00)	3.5 GB	Standard
□	amazon_reviews_us_Video_Games_v1_00.tsv	tsv	October 5, 2023, 22:04:35 (UTC-04:00)	1.1 GB	Standard
□	amazon_reviews_us_Video_v1_00.tsv	tsv	October 5, 2023, 22:05:48 (UTC-04:00)	322.3 MB	Standard
□	amazon_reviews_us_Watches_v1_00.tsv	tsv	October 5, 2023, 22:07:01 (UTC-04:00)	393.4 MB	Standard



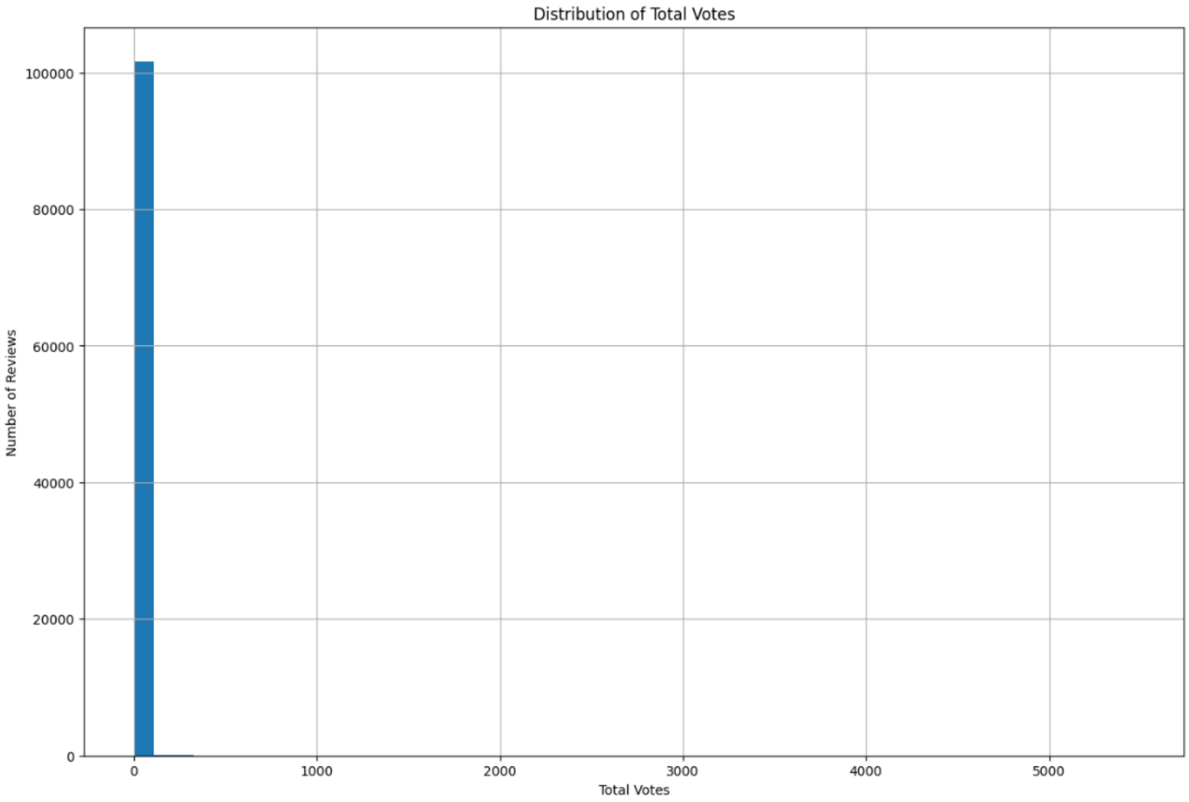
□	amazon_reviews_us_Wireless_v1_00.tsv	tsv	October 5, 2023, 22:08:43 (UTC-04:00)	3.9 GB	Standard

Exploratory Data Analysis (EDA)

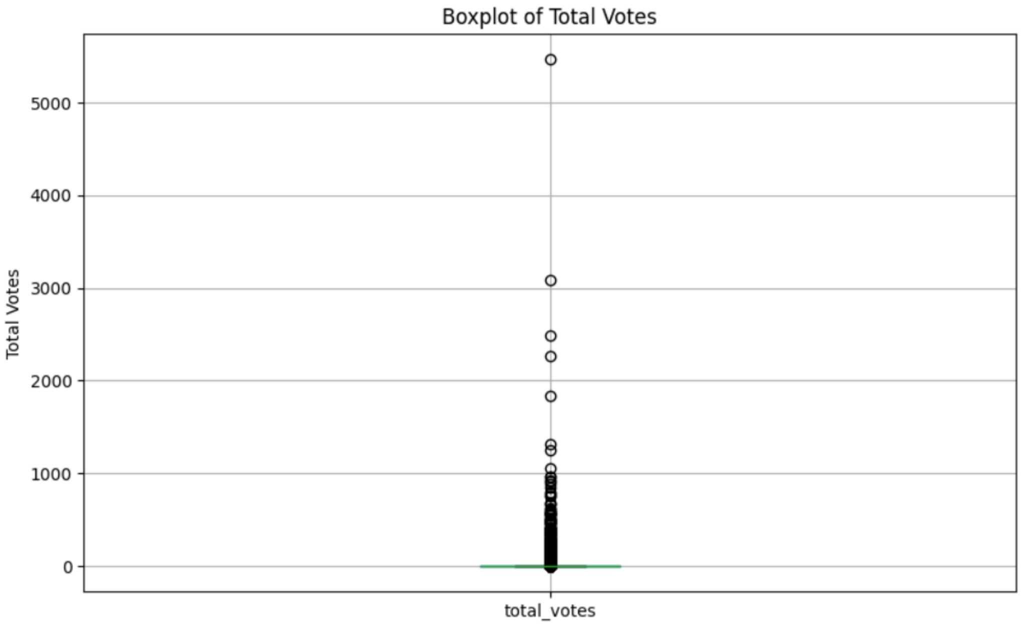
I picked a random data file from the Amazon Reviews dataset to analyze. At first, I chose a dataset that was about 2 GB so it took so long to run the code to load the data in Jupyter Notebook and it ended up crashing so I decided to go with a smaller file " amazon_reviews_us_Digital_Software_V1_00.tsv"; This data file contains reviews of digital software products from amazon such as start rating, total votes and product category and review content that is text heavy
The reviews rating from 1 to 5 was leaning towards positive ratings (4 or 5 stars). In the datafile, lot of products got only a few votes; ;70% of the products got less than 10 votes. However the top 5% received over 500 votes. I could not find many gaps in the data but I noticed that 2% of the reviews did not have a headline; I will need to figure out if i need to fill these gaps or just leave them; i will most likely remove them. The data inlcludes different data types such as text reviews, numerical rating and categories like product types which means I will need to use different tools and strategies to understand each part. In short, the intial findings of this smal datafile will help guide our next steps as we prepare of feature engineering and remodeling the data.



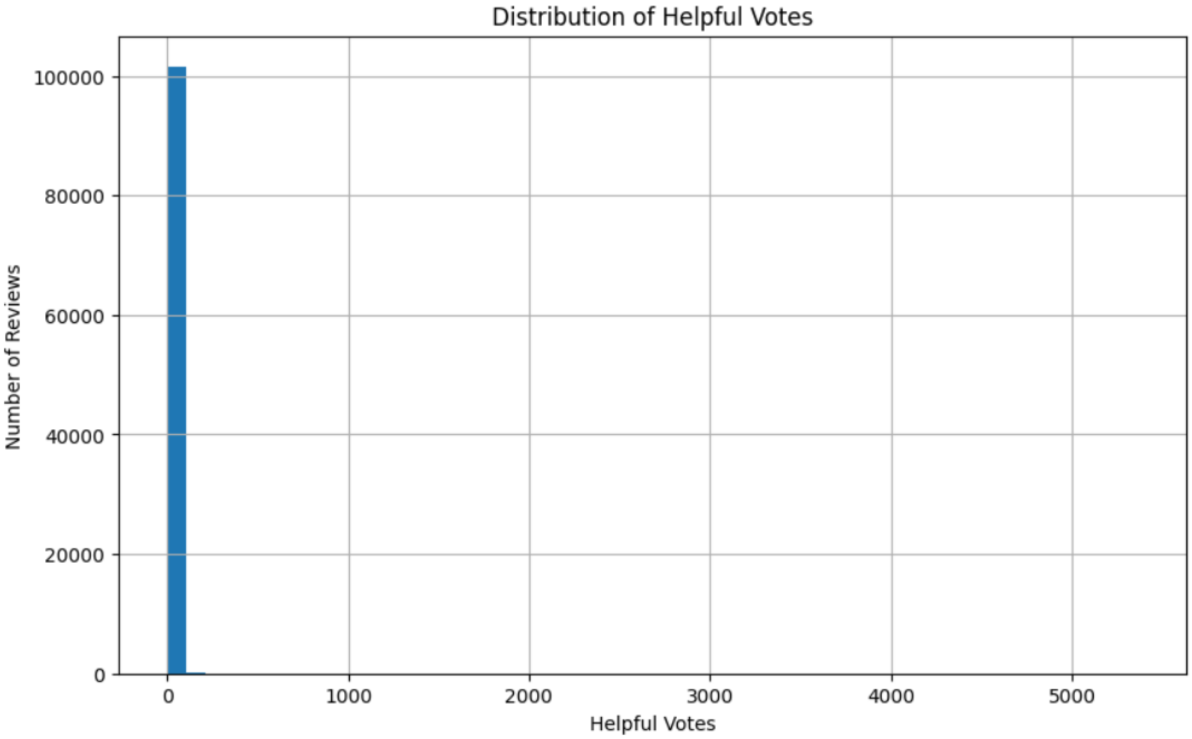
The start Rtaing Distripution graph shows a significant number of reviews, around 40,000 have given a 5-star rating. This indicates a high customer satisfaction of the product that is being reviewed. Around 25,000 reviews have a 1-star rating which shows areas of potential concern. Overall, most users have a strong and highly favorable reviews for the product.



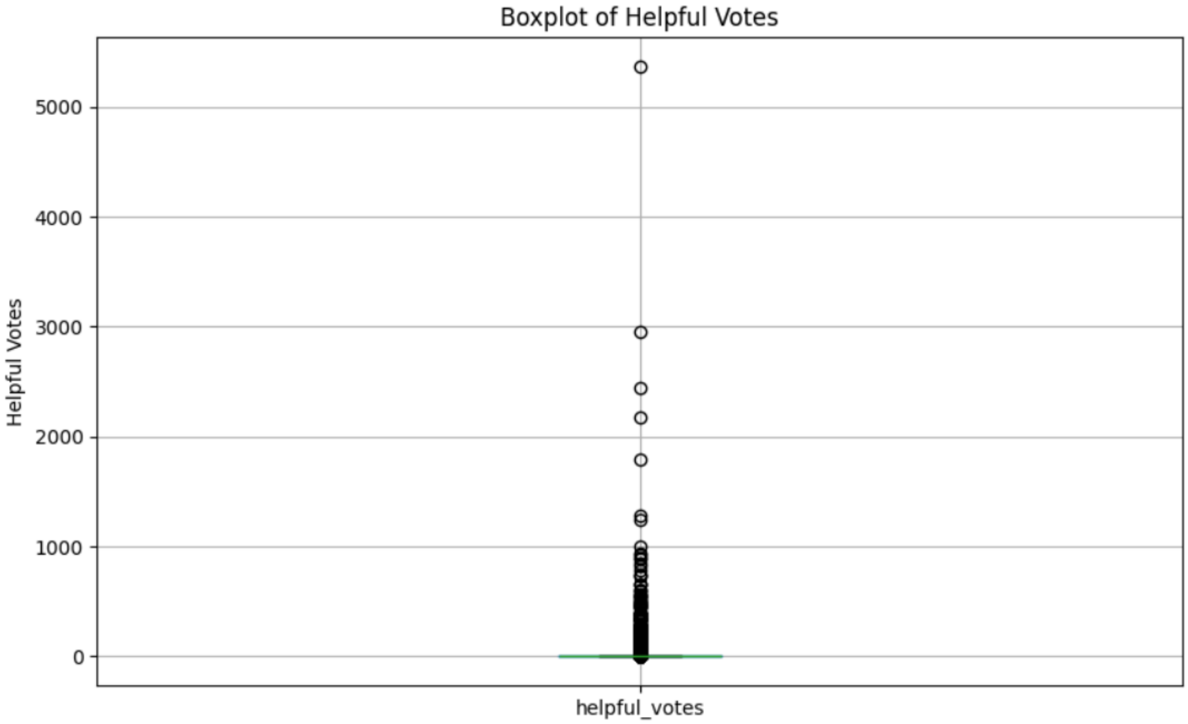
The Distribusion of total votes shows the large majority of the reviews about 90,000 have received between 0 to 100 votes which suggests that many reviews might not have been seen or interacted with. Because of the skweness of the graph, it was hard to see the comparison. Therefore i decide to have a boxplot to show the skewenss of the votes.



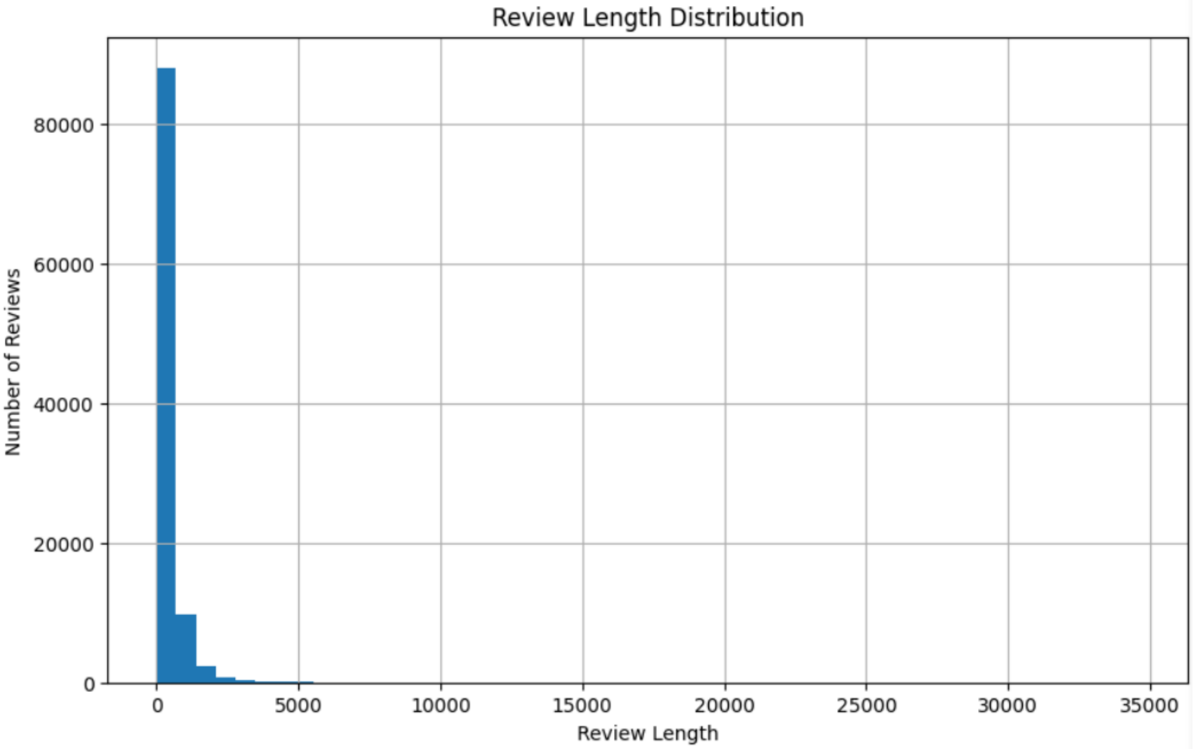
The Median is very close to zero which suggests that more than half of the data points have received a low number of total votes.



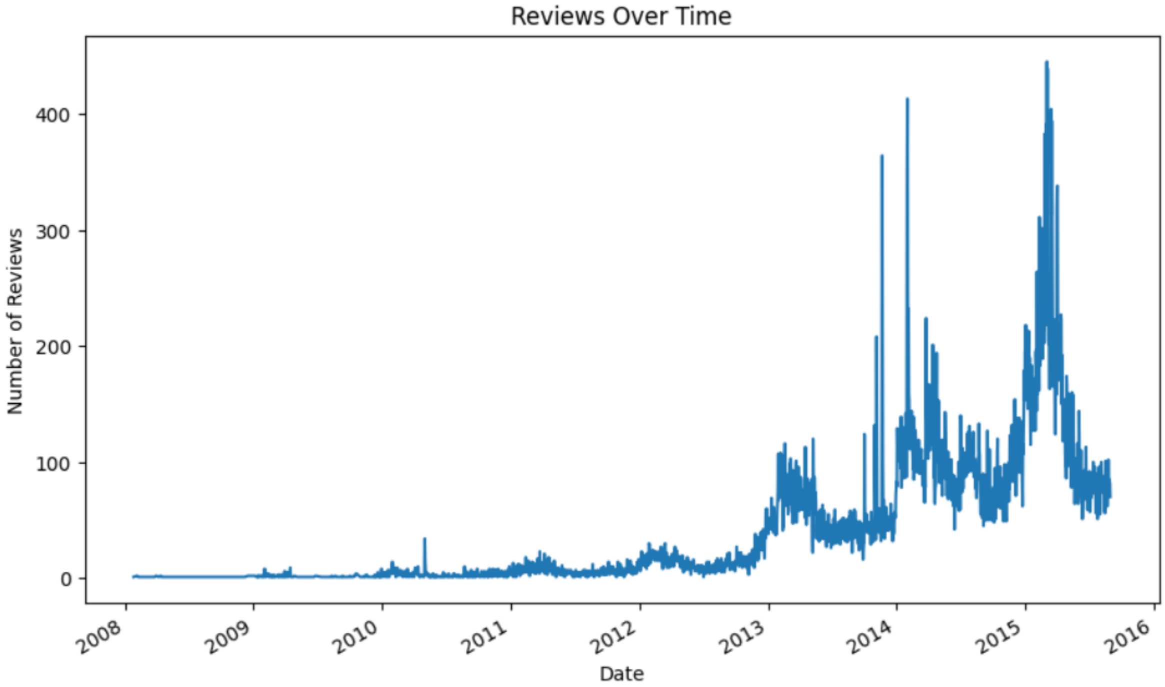
The Ditribution of the helpful votes suggests that the majority of the reviewers received few votes. Essentially most reviews did not get many helpful reviews.



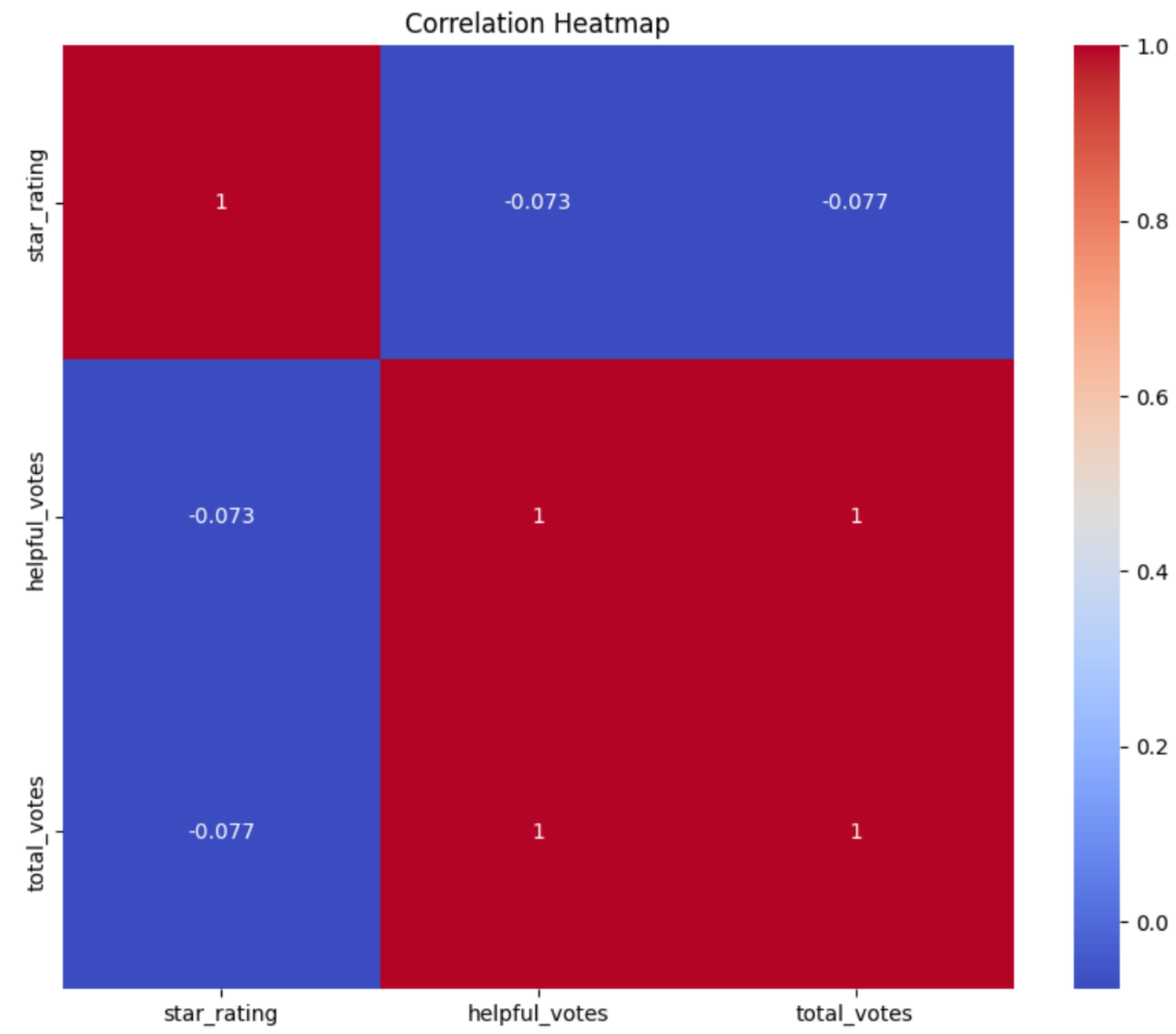
The Boxplot provides more insight about the spread of the data. We reached the same conclusion but visualized in more details.



The Review length distribution graph shows that the majority of the reviews are very short, with significant spike near 0. There is a rapid decline in the number of reviews as their length increases.



The review overtime graph shows the trend of reviews from 2008 to 2016 starting with a very low number in the early years, experiencing a few significant peaks around 2012 and 2014.



The heatmap displays the correction values between ‘star-rating’, ‘helpful_votes’, and ‘total_votes’. There is a weak negative correlation between ‘star_rating’ and both ‘helpful_votes’ and total_votes’. Hoever, ‘helpful_votes’ and ‘total_votes’ show a perfect positive correlation of 1.

Feature Engineering and Modeling:

When I work with the Amazon Reviews dataset, the first thing I need to do is clean and prepare the data very carefully. This dataset has a lot of different kinds of data in it, such as textual customer comments, categorical product information, and numerical ratings. Our goal wasn't just to clean up the data; it was also to make it work with complex machine learning techniques.

I started this process by getting the data from the /landing folder, which holds raw data that hasn't been handled yet. This step was very important because it set the stage for everything else. Next, I got rid of any empty numbers in key columns like customer_id, product_id, and star_rating. This step of cleaning was very important to make sure that our information stayed reliable. To make our predictive modeling goals easier to understand and achieve, I chose to leave out some fields, such as marketplace and vine, that I didn't think were important. This cleaning process wasn't just a way to get rid of data; it was also an effort to get to the goal of the information.

By saving the cleaned data in the /raw folder, the cleaning process was complete, and the data is now ready for the next stage of transformation.

As I worked on improving features, I first came across the product_category column. The variable was a categorical one that needed to be turned into a numerical one so that our machine learning programs could understand it. The StringIndexer did a good job with this version. Next, I had to make the star_rating easier to understand. To do this, I used the Binarizer to cleverly turn the complexity of the different scores into a binary format. This put feelings into two groups: high and low. The helpful_votes field was especially hard, so I used the Bucketizer to separate the votes into clear categories, which made it easier to analyze. When I got to review_headline and review_body, I didn't ignore the subtleties of their text; instead, I used the FeatureHasher method to turn them into feature vectors. This made a big difference in lowering the number of dimensions in our text data. Finally, VectorAssembler was used to smoothly combine all of these engineered parts into a single whole. With this last step, the data was finally ready to be used for predictive models. The improved and expanded form of the data was then saved in the /trusted folder to show that it is now ready for further examination.

I carefully cleaned up and improved the Amazon Reviews information. By pulling out important information, our goal was to make it more useful for machine learning algorithms. Every step was planned out so that the final dataset would be clean, well-organized, and have traits that can help with predictive modeling.

The StringIndexer portion of PySpark's ML package caused troubles during feature engineering. Machine learning models prefer number indices, which the StringIndexer converts string-type category data into. However, I made blunders that indicated data processing issues.

The fundamental issue with StringIndexer was that it couldn't support non-string data types. In our dataset, certain indexing fields were numbers or other non-string types, causing an IllegalArgumentException. This error suggested the DataFrame's raw data type did not match the anticipated one.

Handling Null Values: StringIndexer also struggled with null values in fields. Null values may create data processing errors or unexpected behavior, making feature changes less stable and trustworthy.

I changed the code to make it compliant and dependable throughout feature engineering to avoid these issues.

Before using StringIndexer, I updated the code to explicitly convert field data types to strings. This patch ensured that StringIndexer always received the correct string format, fixing the data type compatibility issue.
PySpark's withColumn and cast methods converted product_category to string.

I designed steps to fill null values with a placeholder word (like "unknown") or delete entries with null values in fields that need to be stored.
This prevented StringIndexer from encountering nulls, preventing errors.
These adjustments were crucial for feature engineering success. Fixing StringIndexer allowed the data to easily convert category data to machine learning model format. This was crucial to the Amazon Reviews analysis.

While training and testing a predictive model using the Amazon Reviews dataset on DaraBricks cluster, I encountered many issues that made it difficult to proceed. These issues were largely related to the Cluster and data access.

What Followed Cluster Termination
The biggest issue was the cluster shutting down unexpectedly during model training.

Data Inaccessibility: Training and validation data requests failed after the cluster was shut down. I encountered a FileReadException while trying to read a Parquet file from the S3 bucket. This error suggested the file's existence or stability was compromised, maybe due to its abrupt conclusion.

After the unexpected shutdown, Spark's cache may have been inconsistent, making updated data files hard to access. Even after refreshing the cache and restarting the cluster, these issues persisted, making it difficult to access model training data.

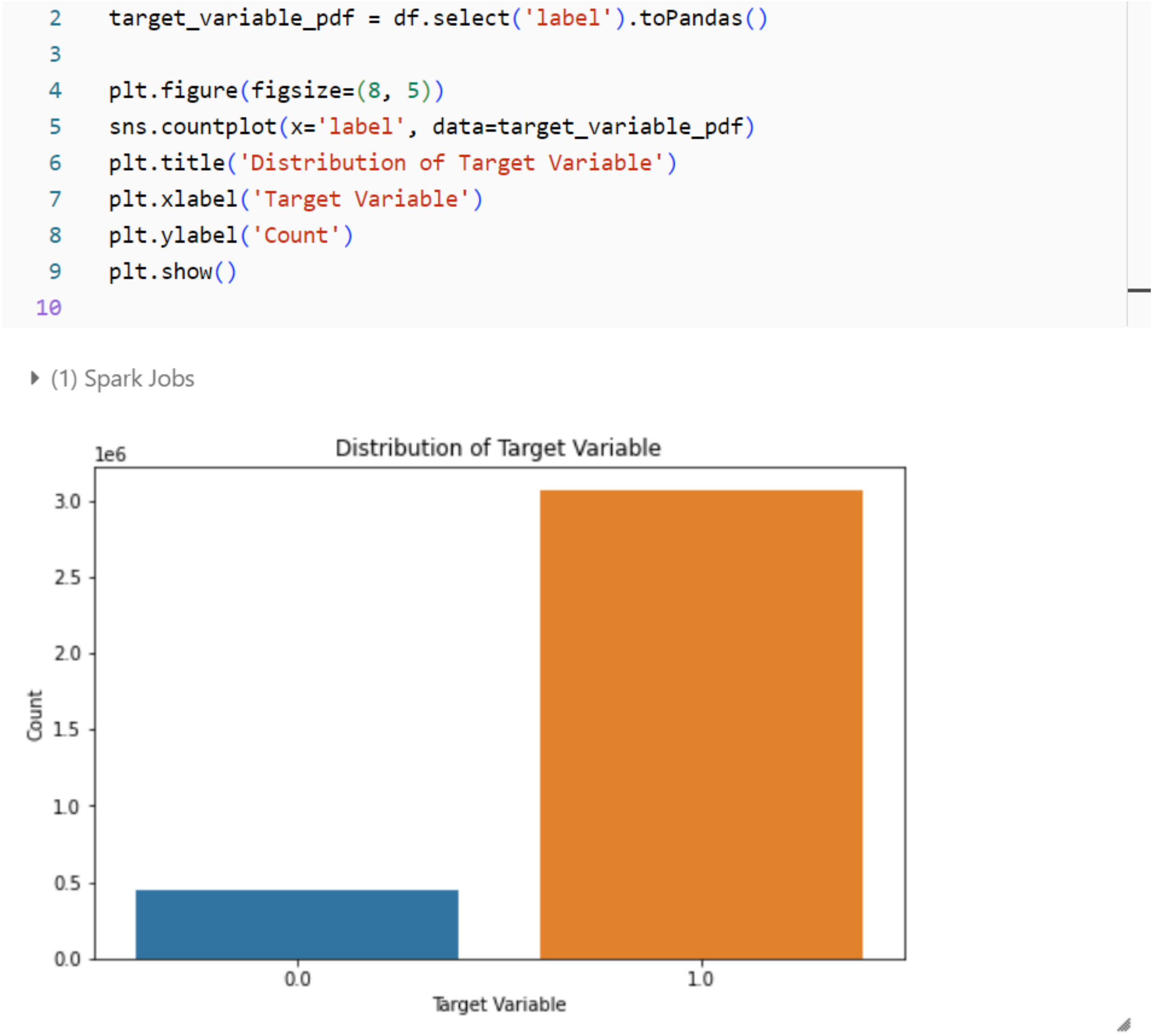
Network and configuration issues: The cluster's termination and restart raised concerns about network or configuration issues impacting the cluster-AWS S3 data access.
These issues slowed the data processing, which affected the following:

The model could not be trained without stable access to the provided data. I couldn't feed machine learning algorithms data without reading and analyzing it.

To address the cluster termination issue and the timing issue, I picked a smaller sample of the dataset for initial training and testing. I also replaced the Feature Hasher with TF-IDF to process the text data which is more memory-efficient. I also excluded an extra variable hiRating to prevent data leakage. The trained model was saved to the models folders in S3 for future use.

Data Visualization

After evaluating the professor’s feedback, I redid the visualizations using different insights such as ROC curve and Correlaton heatmap inaddition to my original visualization.

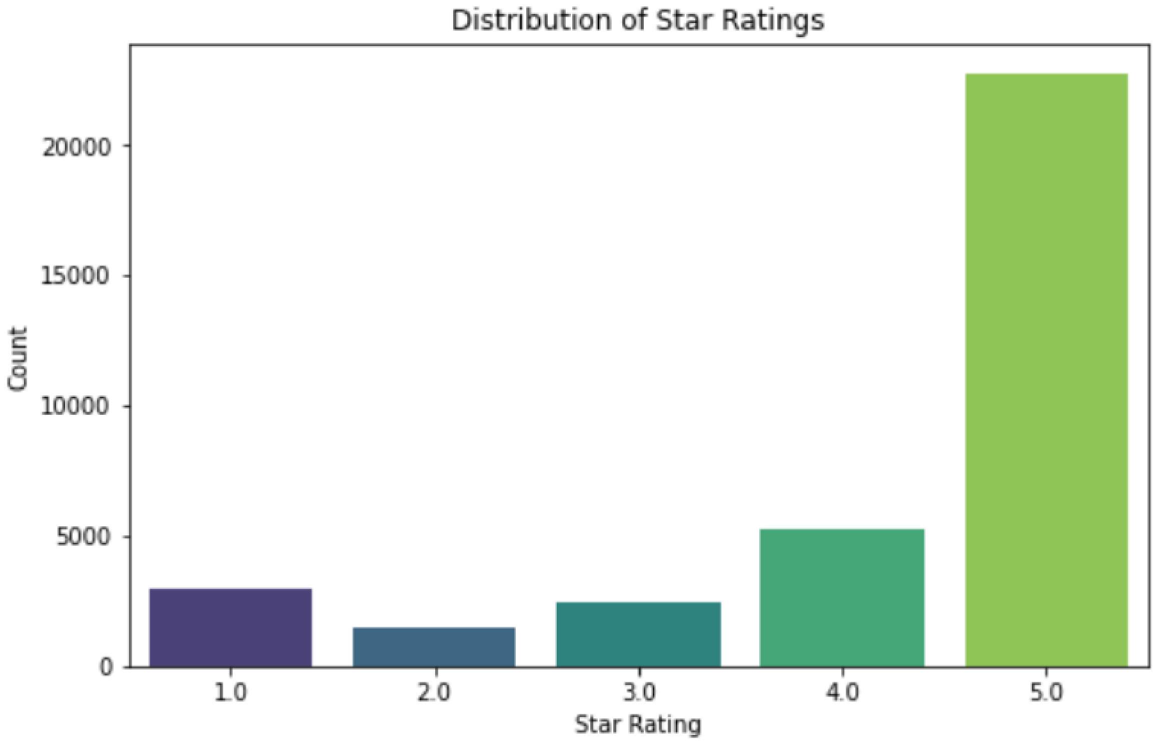


```
target_variable_pdf = df.select('label').toPandas()

plt.figure(figsize=(8, 5))
sns.countplot(x='label', data=target_variable_pdf)
plt.title('Distribution of Target Variable')
plt.xlabel('Target Variable')
plt.ylabel('Count')
plt.show()

```

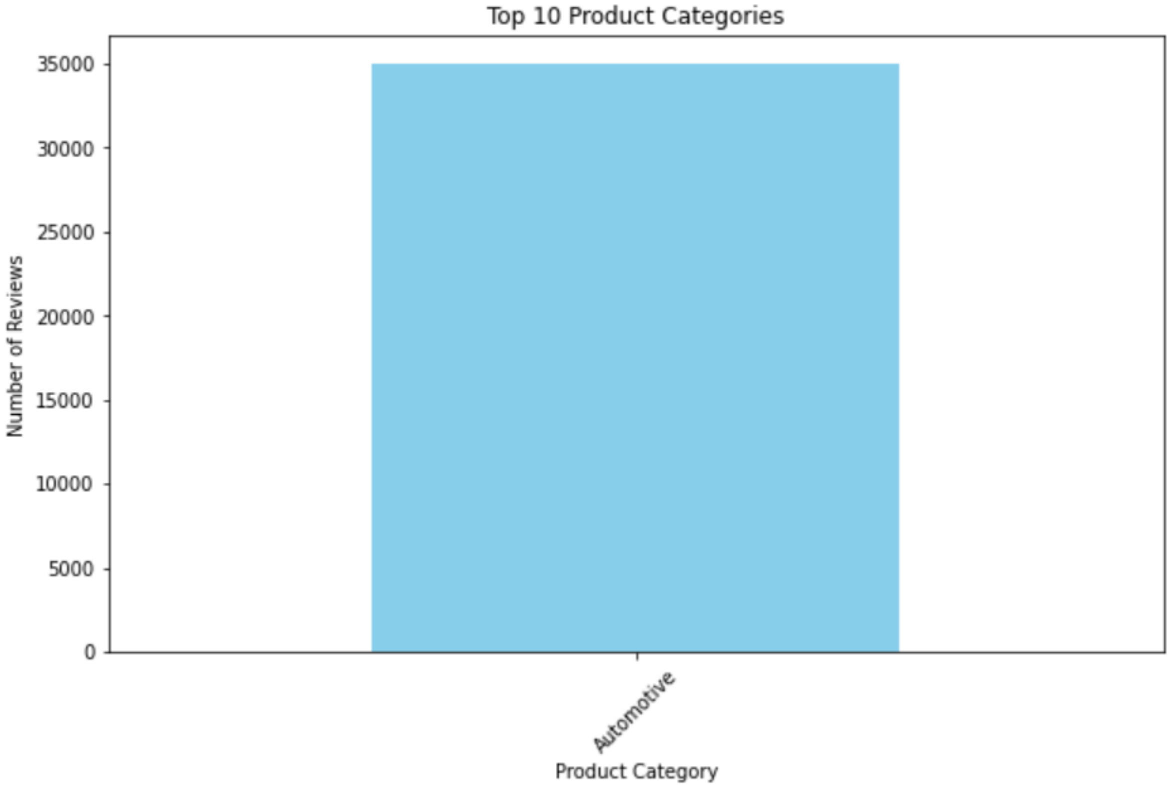
(1) Spark Jobs

This graph examines the distribution of the target variable to understand the dataset’s balance.

▸ (1) Spark Jobs



This visualization shows the distribution of star ratings in your dataset, which is useful for understanding the overall sentiment of the reviews.



This chart displays the frequency of reviews in different product categories, giving an idea of which categories are most reviewed.

I attempted to work on other two visualizations to show Comparison of Actual vs. Predicted Ratings and Feature Importance in the RandomForest Model. However, I encountered errors that I could not bypass. 1 had to do with the length of the arrays which i could not figure out how to correct it and the second had to do with the ussuportive type in conversion to arrow: VectorUDT().

Summary and Conclution

In a world where data reigns supreme our project embarked on a mission to unlock the potential of data analytics. Specifically, focused on a dataset consisting of Amazon product reviews. By utilizing Apache Spark, one of the leading tools, in the field of data the goal was to delve into the details hidden within these reviews and transform vast amounts of data into meaningful insights.

The foundation of the project relied on a constructed pipeline for processing data. This pipeline was designed to navigate through the complexities of data with efficiency and accuracy. The initial step involved cleaning and preprocessing of the data. removed columns like customer_id. Addressed any null values in crucial fields such as product_id and star_rating. This was not a process; it was a strategic decision aimed at improving the quality and relevance of the data.

Moving forward into feature engineering. It took an approach by incorporating the TF IDF technique for processing information. This allowed to extract nuanced features from the reviews while ensuring that the predictive modeling maintained integrity by isolating the target variable, highRating thus preventing any leakage or bias in our analysis.

At the heart of our analysis lay the Random Forest Classifier—a machine learning model that played a role in extracting valuable insights, from the dataset.
After acknowledging the limitations of processing power and memory it took an approach by training the model on a sample of just 1% of the data. This not sped up the training process. Also showcased our dedication to using resources efficiently. Following data practices utilized AWS S3 to store our models ensuring their accessibility, for use.

The project went beyond data processing. I went deeper into creating a range of visualizations that brought our data to life. From examining the distribution of the target variable to uncovering the importance of features in our Random Forest model each visualization provided insights into underlying patterns within customer feedback.

Navigating through this complex pipeline of data led to several significant discoveries. The critical role played by feature relevance in predicting customer satisfaction became abundantly clear. Additionally this project shed light on the importance of maintaining data integrity. Highlighted how effective our chosen machine learning model was at making predictions.
It emphasized the need, for resource management especially when handling datasets.

To concludeThis project went beyond analyzing Amazon reviews. It showcased the potential of utilizing data tools and techniques illustrating the process of transforming data into valuable insights. As concluded the project, GitHub repository served as an archive allowing others to explore the work in depth and potentially expand upon it.

References

Apache Spark. (n.d.). Apache Spark Documentation. https://spark.apache.org/docs/latest/.

Amazon Web Services, Inc. (n.d.). AWS Documentation. https://docs.aws.amazon.com/.

Databricks Inc. (n.d.). Databricks Community. https://community.databricks.com/.

Stack Overflow. (n.d.). https://stackoverflow.com/.

Amazon Web Services, Inc. (n.d.). Amazon EMR Management Guide. https://docs.aws.amazon.com/emr/latest/ManagementGuide/.

Apache Parquet. (n.d.). Apache Parquet Documentation. https://parquet.apache.org/documentation/latest/.

Ryza, S., Laserson, U., Owen, S., & Wills, J. (2015). Big Data Analytics with Spark: A Practitioner's Guide to Using Spark for Large Scale Data Analysis. O'Reilly Media.

Appedix A:

Downloading the Kaggle Dataset

Instructions:

- Use the command below to fetch the dataset:

```
“kaggle datasets download -d cynthiarempel/amazon-us-customer-reviews-dataset”
```

Listing Filenames Inside the Zip

Instructions:

- To view the files within the zipped dataset, use:

  ```
  “unzip -l amazon-us-customer-reviews-dataset.zip”
  ```

Data Extraction and Upload

Unzipping Files:

Description:

- Extract individual data files from the zipped dataset.
  For amazon_reviews_us_Apparel_v1_00.tsv, use:

Copying to S3:

Description:

- Transfer the extracted dataset to Amazon S3.

```
unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_multilingual_US_v1_00.tsv
aws s3 cp amazon_reviews_multilingual_US_v1_00.tsv  s3://
s3://my-bigdata-project-ae/landing/amazon_reviews_multilingual_US_v1_00.tsv
rm amazon_reviews_multilingual_US_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Apparel_v1_00.tsv
aws s3 cp amazon_reviews_us_Apparel_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Apparel_v1_00.tsv
rm amazon_reviews_us_Apparel_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Automotive_v1_00.tsv
aws s3 cp amazon_reviews_us_Automotive_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Automotive_v1_00.tsv
rm amazon_reviews_us_Automotive_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Baby_v1_00.tsv
```

```
aws s3 cp amazon_reviews_us_Baby_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Baby_v1_00.tsv
rm amazon_reviews_us_Baby_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Watches_v1_00.tsv
aws s3 cp amazon_reviews_us_Watches_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Watches_v1_00.tsv
rm amazon_reviews_us_Watches_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Wireless_v1_00.tsv
aws s3 cp amazon_reviews_us_Wireless_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Wireless_v1_00.tsv
rm amazon_reviews_us_Wireless_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_multilingual_US_v1_00.tsv
aws s3 cp amazon_reviews_multilingual_US_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_multilingual_US_v1_00.tsv
rm amazon_reviews_multilingual_US_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Apparel_v1_00.tsv
aws s3 cp amazon_reviews_us_Apparel_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Apparel_v1_00.tsv
rm amazon_reviews_us_Apparel_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Automotive_v1_00.tsv
aws s3 cp amazon_reviews_us_Automotive_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Automotive_v1_00.tsv
rm amazon_reviews_us_Automotive_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Baby_v1_00.tsv
aws s3 cp amazon_reviews_us_Baby_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Baby_v1_00.tsv
rm amazon_reviews_us_Baby_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Beauty_v1_00.tsv
aws s3 cp amazon_reviews_us_Beauty_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Beauty_v1_00.tsv
rm amazon_reviews_us_Beauty_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Watches_v1_00.tsv
```

```
aws s3 cp amazon_reviews_us_Watches_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Watches_v1_00.tsv
rm amazon_reviews_us_Watches_v1_00.tsv



unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Wireless_v1_00.tsv
aws s3 cp amazon_reviews_us_Wireless_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Wireless_v1_00.tsv
rm amazon_reviews_us_Wireless_v1_00.tsv


unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Books_v1_02.tsv
aws s3 cp amazon_reviews_us_Books_v1_02.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Books_v1_02.tsv
rm amazon_reviews_us_Books_v1_02.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Camera_v1_00.tsv
aws s3 cp amazon_reviews_us_Camera_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Camera_v1_00.tsv
rm amazon_reviews_us_Camera_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Digital_Ebook_Purchase_v1_01.tsv
aws s3 cp amazon_reviews_us_Digital_Ebook_Purchase_v1_01.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Digital_Ebook_Purchase_v1_01.tsv
rm amazon_reviews_us_Digital_Ebook_Purchase_v1_01.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Digital_Music_Purchase_v1_00.tsv
aws s3 cp amazon_reviews_us_Digital_Music_Purchase_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Digital_Music_Purchase_v1_00.tsv
rm amazon_reviews_us_Digital_Music_Purchase_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Watches_v1_00.tsv
aws s3 cp amazon_reviews_us_Watches_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Watches_v1_00.tsv
rm amazon_reviews_us_Watches_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Wireless_v1_00.tsv
```

```
aws s3 cp amazon_reviews_us_Wireless_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Wireless_v1_00.tsv
rm amazon_reviews_us_Wireless_v1_00.tsv


unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Digital_Software_v1_00.tsv
aws s3 cp amazon_reviews_us_Digital_Software_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Digital_Software_v1_00.tsv
rm amazon_reviews_us_Digital_Software_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Digital_Video_Download_v1_00.tsv
aws s3 cp amazon_reviews_us_Digital_Video_Download_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Digital_Video_Download_v1_00.tsv
rm amazon_reviews_us_Digital_Video_Download_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Digital_Video_Games_v1_00.tsv
aws s3 cp amazon_reviews_us_Digital_Video_Games_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Digital_Video_Games_v1_00.tsv
rm amazon_reviews_us_Digital_Video_Games_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Electronics_v1_00.tsv
aws s3 cp amazon_reviews_us_Electronics_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Electronics_v1_00.tsv
rm amazon_reviews_us_Electronics_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Furniture_v1_00.tsv
aws s3 cp amazon_reviews_us_Furniture_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Furniture_v1_00.tsv
rm amazon_reviews_us_Furniture_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Gift_Card_v1_00.tsv
aws s3 cp amazon_reviews_us_Gift_Card_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Gift_Card_v1_00.tsv
rm amazon_reviews_us_Gift_Card_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Grocery_v1_00.tsv
```

```
aws s3 cp amazon_reviews_us_Grocery_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Grocery_v1_00.tsv
rm amazon_reviews_us_Grocery_v1_00.tsv
```

```
unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Wireless_v1_00.tsv
aws s3 cp amazon_reviews_us_Wireless_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Wireless_v1_00.tsv
rm amazon_reviews_us_Wireless_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Health_Personal_Care_v1_00.tsv
aws s3 cp amazon_reviews_us_Health_Personal_Care_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Health_Personal_Care_v1_00.tsv
rm amazon_reviews_us_Health_Personal_Care_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Major_Appliances_v1_00.tsv
aws s3 cp amazon_reviews_us_Major_Appliances_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Major_Appliances_v1_00.tsv
rm amazon_reviews_us_Major_Appliances_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Mobile_Apps_v1_00.tsv
aws s3 cp amazon_reviews_us_Mobile_Apps_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Mobile_Apps_v1_00.tsv
rm amazon_reviews_us_Mobile_Apps_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Mobile_Electronics_v1_00.tsv
aws s3 cp amazon_reviews_us_Mobile_Electronics_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Mobile_Electronics_v1_00.tsv
rm amazon_reviews_us_Mobile_Electronics_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Music_v1_00.tsv
aws s3 cp amazon_reviews_us_Music_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Music_v1_00.tsv
rm amazon_reviews_us_Music_v1_00.tsv
```

```
unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Musical_Instruments_v1_00.tsv
aws s3 cp amazon_reviews_us_Musical_Instruments_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Musical_Instruments_v1_00.tsv
rm amazon_reviews_us_Musical_Instruments_v1_00.tsv



unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Office_Products_v1_00.tsv
aws s3 cp amazon_reviews_us_Office_Products_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Office_Products_v1_00.tsv
rm amazon_reviews_us_Office_Products_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Outdoors_v1_00.tsv
aws s3 cp amazon_reviews_us_Outdoors_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Outdoors_v1_00.tsv
rm amazon_reviews_us_Outdoors_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_PC_v1_00.tsv
aws s3 cp amazon_reviews_us_PC_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_PC_v1_00.tsv
rm amazon_reviews_us_PC_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip
amazon_reviews_us_Personal_Care_Appliances_v1_00.tsv
aws s3 cp amazon_reviews_us_Personal_Care_Appliances_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Personal_Care_Appliances_v1_00.tsv
rm amazon_reviews_us_Personal_Care_Appliances_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Pet_Products_v1_00.tsv
aws s3 cp amazon_reviews_us_Pet_Products_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Pet_Products_v1_00.tsv
rm amazon_reviews_us_Pet_Products_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Shoes_v1_00.tsv
aws s3 cp amazon_reviews_us_Shoes_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Shoes_v1_00.tsv
rm amazon_reviews_us_Shoes_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Software_v1_00.tsv
```

```
aws s3 cp amazon_reviews_us_Software_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Software_v1_00.tsv
rm amazon_reviews_us_Software_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Sports_v1_00.tsv
aws s3 cp amazon_reviews_us_Sports_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Sports_v1_00.tsv
rm amazon_reviews_us_Sports_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Tools_v1_00.tsv
aws s3 cp amazon_reviews_us_Tools_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Tools_v1_00.tsv
rm amazon_reviews_us_Tools_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Toys_v1_00.tsv
aws s3 cp amazon_reviews_us_Toys_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Toys_v1_00.tsv
rm amazon_reviews_us_Toys_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Video_DVD_v1_00.tsv
aws s3 cp amazon_reviews_us_Video_DVD_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Video_DVD_v1_00.tsv
rm amazon_reviews_us_Video_DVD_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Video_Games_v1_00.tsv
aws s3 cp amazon_reviews_us_Video_Games_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Video_Games_v1_00.tsv
rm amazon_reviews_us_Video_Games_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Video_v1_00.tsv
aws s3 cp amazon_reviews_us_Video_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Video_v1_00.tsv
rm amazon_reviews_us_Video_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Watches_v1_00.tsv
aws s3 cp amazon_reviews_us_Watches_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Watches_v1_00.tsv
rm amazon_reviews_us_Watches_v1_00.tsv

unzip amazon-us-customer-reviews-dataset.zip amazon_reviews_us_Wireless_v1_00.tsv
```

```
aws s3 cp amazon_reviews_us_Wireless_v1_00.tsv
s3://my-bigdata-project-ae/landing/amazon_reviews_us_Wireless_v1_00.tsv
rm amazon_reviews_us_Wireless_v1_00.tsv
```

Appedix B:

```
# Load Data

import boto3
import pandas as pd

bucket_name = 'my-bigdata-project-ae'
file_name = 'landing/amazon_reviews_us_Digital_Software_v1_00.tsv'

obj = s3.get_object(Bucket=bucket_name, Key=file_name)
data = pd.read_csv(obj['Body'], sep='\t', on_bad_lines='skip')

# View Sample Data

print(data.head())

# Descriptive Statistics

print(f"Number of observations: {data.shape[0]}")

print(f"Variables: {data.columns.tolist()}")

print(data.isnull().sum())

numeric_data = data.select_dtypes(include=['number'])
print(numeric_data.describe())

date_cols = ['review_date']
# since the review_date contains mix of data types, I Converted 'review_date' column to
dateTime
data['review_date'] = pd.to_datetime(data['review_date'], errors='coerce')
for col in date_cols:
   print(f"Min date in {col}: {data[col].min()}")
   print(f"Max date in {col}: {data[col].max()}")
```

```
text_columns = ['review_body', 'review_headline', 'product_title', 'product_category']

for col in text_columns:
    word_count_col = col + '_word_count'
    data[word_count_col] = data[col].apply(lambda x: len(str(x).split()))

    print(f"Column: {col}")
    print(f"Average word count: {data[word_count_col].mean()}")
    print(f"Minimum word count: {data[word_count_col].min()}")
    print(f"Maximum word count: {data[word_count_col].max()}")
    print("------------------------------")


# Graphs and Charts

import seaborn as sns
import matplotlib.pyplot as plt

pip install seaborn wordcloud

import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud

# Star Rating Distribution
plt.figure(figsize=(10,6))
data['star_rating'].value_counts().sort_index().plot(kind='bar')
plt.title('Star Rating Distribution')
plt.xlabel('Star Rating')
plt.ylabel('Number of Reviews')
plt.show()

# Total Votes Distribution
plt.figure(figsize=(15,10))
data['total_votes'].hist(bins=50)
plt.title('Distribution of Total Votes')
plt.xlabel('Total Votes')
plt.ylabel('Number of Reviews')
plt.show()
```

```
#Since the data is so skewed to 0, I used Voxplot to get a better idea of the distripution.
plt.figure(figsize=(10,6))
data.boxplot(column=['total_votes'])
plt.title('Boxplot of Total Votes')
plt.ylabel('Total Votes')
plt.show()

# Helpful Votes
plt.figure(figsize=(10,6))
data['helpful_votes'].hist(bins=50)
plt.title('Distribution of Helpful Votes')
plt.xlabel('Helpful Votes')
plt.ylabel('Number of Reviews')
plt.show()

#Since the data is so skewed to 0, I used Voxplot to get a better idea of the distripution.
plt.figure(figsize=(10,6))
data.boxplot(column=['helpful_votes'])
plt.title('Boxplot of Helpful Votes')
plt.ylabel('Helpful Votes')
plt.show()


# Review Length
data['review_length'] = data['review_body'].apply(lambda x: len(str(x)))
plt.figure(figsize=(10,6))
data['review_length'].hist(bins=50)
plt.title('Review Length Distribution')
plt.xlabel('Review Length')
plt.ylabel('Number of Reviews')
plt.show()


# Reviews Over Time
data['review_date'] = pd.to_datetime(data['review_date'])
data.groupby('review_date').size().plot(figsize=(10,6))
plt.title('Reviews Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Reviews')
plt.show()
```

```
# Correlation Heatmap
import seaborn as sns
correlation = data[['star_rating', 'helpful_votes', 'total_votes']].corr()
plt.figure(figsize=(10,8))
sns.heatmap(correlation, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

Appedix C:

```
import os
from pyspark.sql import SparkSession
from pyspark.sql.functions import col


# AWS credentials and configuration
access_key = ''
secret_key = ''
os.environ['AWS_ACCESS_KEY_ID'] = access_key
os.environ['AWS_SECRET_ACCESS_KEY'] = secret_key
aws_region = "us-east-2"

# Initialize Spark Session
spark = SparkSession.builder.appName("DataCleaning").getOrCreate()
spark.conf.set("fs.s3a.access.key", access_key)
spark.conf.set("fs.s3a.secret.key", secret_key)
spark.conf.set("fs.s3a.endpoint", "s3." + aws_region + ".amazonaws.com")

# Paths for data
landing_path = "s3a://my-bigdata-project-ae/landing/"
raw_path = "s3a://my-bigdata-project-ae/raw/"

# List of files to process
file_list = [
    'amazon_reviews_us_Apparel_v1_00.tsv',
    'amazon_reviews_us_Automotive_v1_00.tsv'
]

for file_name in file_list:
    file_path = os.path.join(landing_path, file_name)

   # Read data with inferred schema
    df = spark.read.csv(file_path, sep='\t', inferSchema=True, header=True)

# Displaying raw data
print("Raw Data:")
df.show(5)
```

Appendix D:

Data Cleaning:

```
from pyspark.sql import functions as F


columns_to_drop = ["customer_id", "marketplace", "vine"]
df_cleaned = df.drop(*columns_to_drop)

# 2. Remove records with nulls in critical columns
df_cleaned = df_cleaned.dropna(subset=["product_id", "star_rating"])

# Filter out rows where review_body is null
df = df.filter(df.review_body.isNotNull())

# cleaned data to raw folder as Parquet
df_cleaned.write.mode("overwrite").parquet("s3a://my-bigdata-project-ae/raw/cleaned_data.parquet")

# Displaying cleaned data
print("Cleaned Data:")
df_cleaned.show(5)



from pyspark.ml.feature import IDF, Tokenizer, CountVectorizer
from pyspark.ml import Pipeline

# Read cleaned data from the raw folder
df = spark.read.parquet("s3a://my-bigdata-project-ae/raw/cleaned_data.parquet")
```

Appendix E:

Feature Engineering:

```
from pyspark.sql.functions import col
from pyspark.sql.types import StringType

df = df.withColumn("review_body", col("review_body").cast(StringType()))


from pyspark.sql.functions import col
from pyspark.sql.types import StringType

df = df.withColumn("review_body", col("review_body").cast(StringType()))



tokenizer = Tokenizer(inputCol="review_body", outputCol="words")
df = tokenizer.transform(df)
df.show(5)


from pyspark.ml.feature import CountVectorizer, IDF

cv = CountVectorizer(inputCol="words", outputCol="rawFeatures")
df = cv.fit(df).transform(df)

df.select("review_body", "words", "rawFeatures").show(5, truncate=False)

# Apply IDF
idf = IDF(inputCol="rawFeatures", outputCol="textFeatures")
df = idf.fit(df).transform(df)


df.select("review_body", "words", "rawFeatures", "textFeatures").show(5, truncate=False)


from pyspark.ml.feature import StringIndexer, Binarizer, Bucketizer, VectorAssembler
from pyspark.sql.types import DoubleType
```

```
# StringIndexer for categorical features
category_indexer = StringIndexer(inputCol="product_category", outputCol="categoryIndex")
df = category_indexer.fit(df).transform(df)
from pyspark.ml import PipelineModel


df.select("product_category", "categoryIndex").show(5)

# Binarizer for the target variable
df = df.withColumn("star_rating_double", df["star_rating"].cast(DoubleType()))
binarizer = Binarizer(threshold=2.5, inputCol="star_rating_double", outputCol="label")
df = binarizer.transform(df)

df.select("star_rating", "label").show(5)


assembler = VectorAssembler(inputCols=["categoryIndex", "textFeatures"],
outputCol="features")
df = assembler.transform(df)

df.select("features").show(5, truncate=False)

# transformed data to the trusted folder
df.write.mode("overwrite").parquet("s3a://my-bigdata-project-ae/trusted/transformed_data.parqu
et")

train_df, test_df = df.randomSplit([0.7, 0.3], seed=42)

train_df.show(5)
test_df.show(5)


sampled_train_df = train_df.sample(withReplacement=False, fraction=0.01, seed=42)
sampled_test_df = test_df.sample(withReplacement=False, fraction=0.01, seed=42)


# Define the classifier
rf = RandomForestClassifier(labelCol="label", featuresCol="features", numTrees=10)
```

```
# Train the model on the sampled training data
rf_model = rf.fit(sampled_train_df)

# Make predictions on the sampled test data
predictions = rf_model.transform(sampled_test_df)


# Evaluate the model
evaluator = BinaryClassificationEvaluator(labelCol="label")
accuracy = evaluator.evaluate(predictions)
print(f"Model Accuracy: {accuracy}")
```

Appendix F:

```
import matplotlib.pyplot as plt
import seaborn as sns

pdf = df.toPandas()  # Convert to Pandas DataFrame for visualization

plt.figure(figsize=(8, 5))
sns.countplot(x='star_rating', data=pdf, palette='viridis')
plt.title('Distribution of Star Ratings')
plt.xlabel('Star Rating')
plt.ylabel('Count')
plt.show()

plt.figure(figsize=(10, 6))
pdf['product_category'].value_counts().head(10).plot(kind='bar', color='skyblue')
plt.title('Top 10 Product Categories')
plt.xlabel('Product Category')
plt.ylabel('Number of Reviews')
plt.xticks(rotation=45)
plt.show()

predictions_pdf = predictions.toPandas()

plt.figure(figsize=(8, 5))
sns.countplot(x='highRating', hue='prediction', data=predictions_pdf, palette='Set2')
plt.title('Comparison of Actual and Predicted High Ratings')
plt.xlabel('High Rating')
plt.ylabel('Count')
plt.legend(title='Prediction', labels=['False', 'True'])
plt.show()

import pandas as pd

features = ['categoryIndex', 'votesBucket', 'textFeatures']  # replace with your actual features
importances = rf_model.featureImportances.toArray()
importances_df = pd.DataFrame({'Feature': features, 'Importance': importances})

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=importances_df.sort_values(by='Importance',
ascending=False))
plt.title('Feature Importances in Random Forest Model')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.show()
```