

Introduction

A product recommendation is a suggestion for a product that a customer might want to buy. Recommendation systems play a crucial role in shaping user experiences across e-commerce platforms, streaming services, and online marketplaces. These systems analyze user behavior, preferences, and interactions to provide personalized suggestions.

The aim of this project is to build a model, applying five key unsupervised learning techniques;

1. Clustering - KMeans, DBSCAN, Tsne (Done by Harshita Puthran (B017) & Nupur Divekar (B016))
2. Topic Modelling - LDA, NMF (Done by Anjali Patwa (B010))
3. Anomaly Analysis using Different Models (Done by Vinti Shukla (B026))
4. Anomaly Detection using Different Models (Done by Vinti Shukla (B026))
5. Recommendation System Using Graph Based and KMeans (Done by

Yashika Tirkey (B008))

These help to identify patterns, discover hidden structures, and detect unusual instances in datasets. The objective is to apply these methodologies to our chosen dataset and interpret the results effectively. Further details of our dataset are as follows:

Dataset:

The dataset we have chosen called ‘sample30’, sourced from kaggle, is a dataset with multiple categories and brands of products available in the market, along with its reviews and ratings. There are 15 columns, namely-

user\_id: Unique identifier for each user. product\_id: Unique identifier for each product.

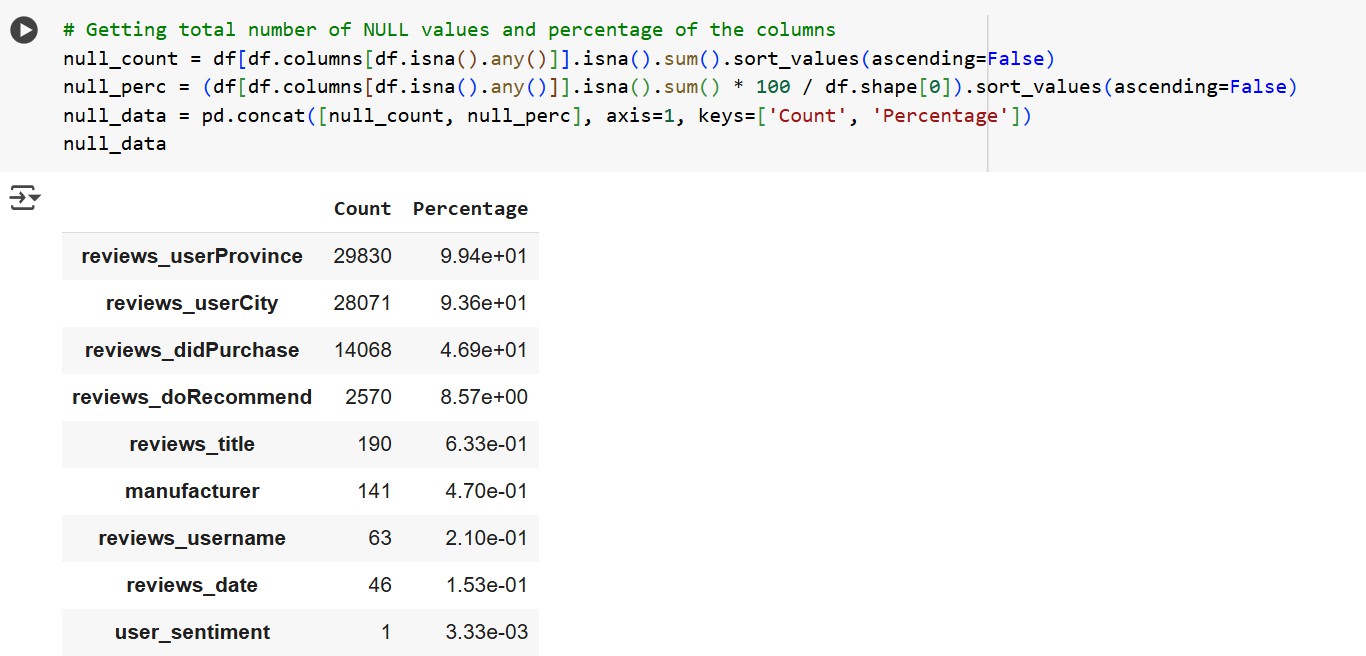
reviews\_rating: User rating (1-5 scale) given to a product. reviews\_didPurchase: Whether the user made an actual purchase. reviews\_doRecommend: Whether the user recommends the product. reviews\_text: User-written review content.

user\_sentiment: Whether the review classifies as positive, negative or neutral.

There are over 30000 rows of data that assist us in creating a model that is well-fitted and balanced.

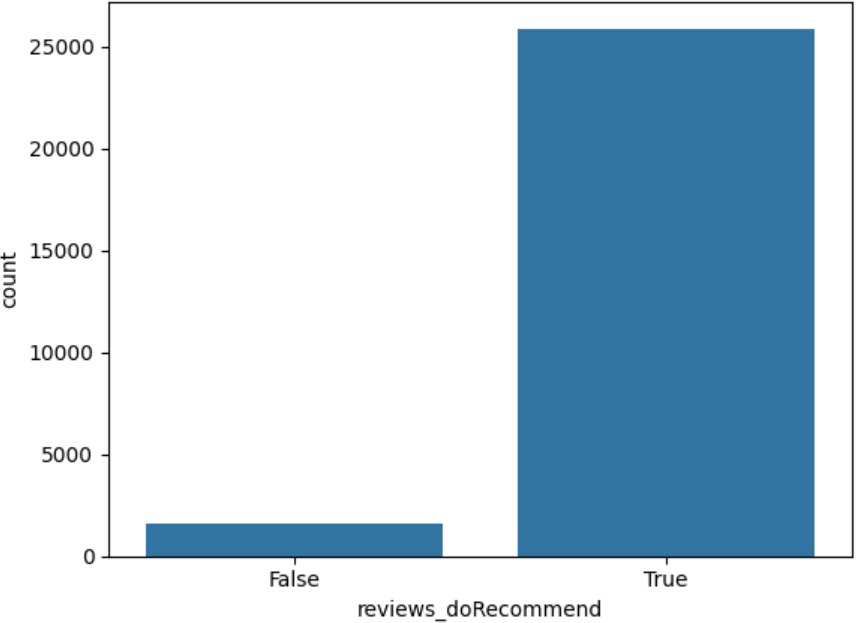
## EDA & Preprocessing

Before building any model, it's essential to understand the data we are working with. EDA allows us to clean, preprocess, and visualize the dataset to uncover patterns and relationships. One of the primary steps of this stage is Data cleaning and missing value handling.

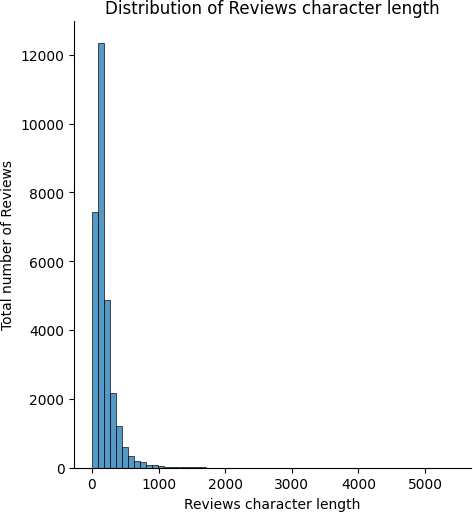
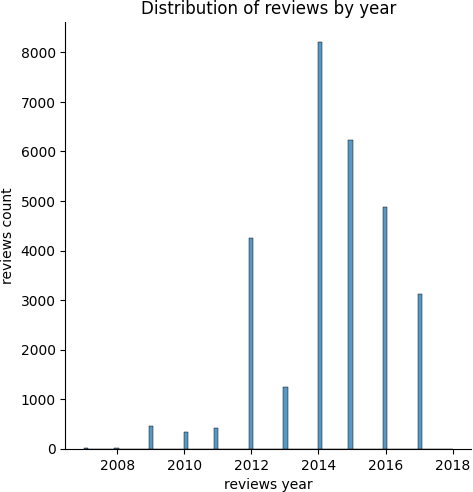
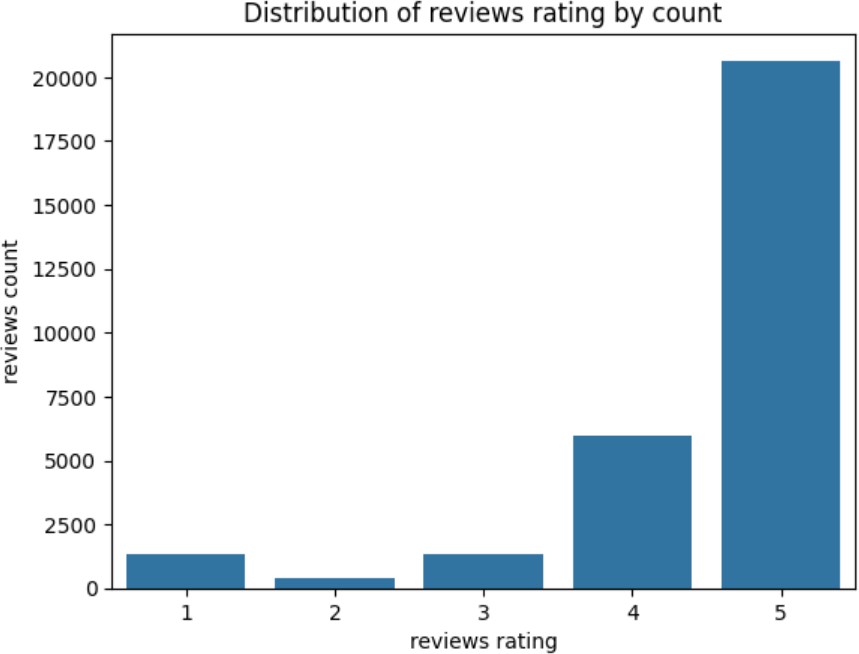


At this stage we removed ‘reviews\_userProvince’, 'reviews\_userCity', 'reviews\_didPurchase', because not only does it have more than 93% missing values, but it also doesn’t have much importance to our project.

Upon checking the value distribution of ‘reviews\_doRecommend’, it is imbalanced and nearly half of the values are True. This will not add any value to our analysis so we will also remove this column.



For the rest of the columns, we will remove rows with missing values. Now some of the plots to better understand the data are:



Lastly, we made sure that all columns were of string data type, to make the next stages easier.

**Clustering**

Clustering is a Unsupervised machine learning technique used to group similar data points based on inherent patterns. In this project, we employ K-Means clustering, which partitions data into K distinct clusters by minimizing the variance within each cluster.

**Data Preprocessing**: Before applying K-Means clustering, data preprocessing is essential to ensure accurate and meaningful results. The preprocessing steps include:

* **Loading the Dataset**: The dataset is loaded into the environment for analysis.
* **Normalization**: Data is normalized to bring all features to a similar scale, preventing any dominant influence from variables with larger values.

## K-Means:

K-Means is an unsupervised learning algorithm that partitions data into K clusters. The steps involved include:

1. **Choosing the Optimal K Value**: The Elbow Method is used to determine the best number of clusters by analyzing the inertia (sum of squared distances of data points from their cluster center).
2. **Fitting the K-Means Algorithm**: The K-Means algorithm is applied to the dataset, and the model assigns each data point to one of the K clusters.
3. **Predicting Cluster Assignments**: After training, the model labels each data point with a corresponding cluster.

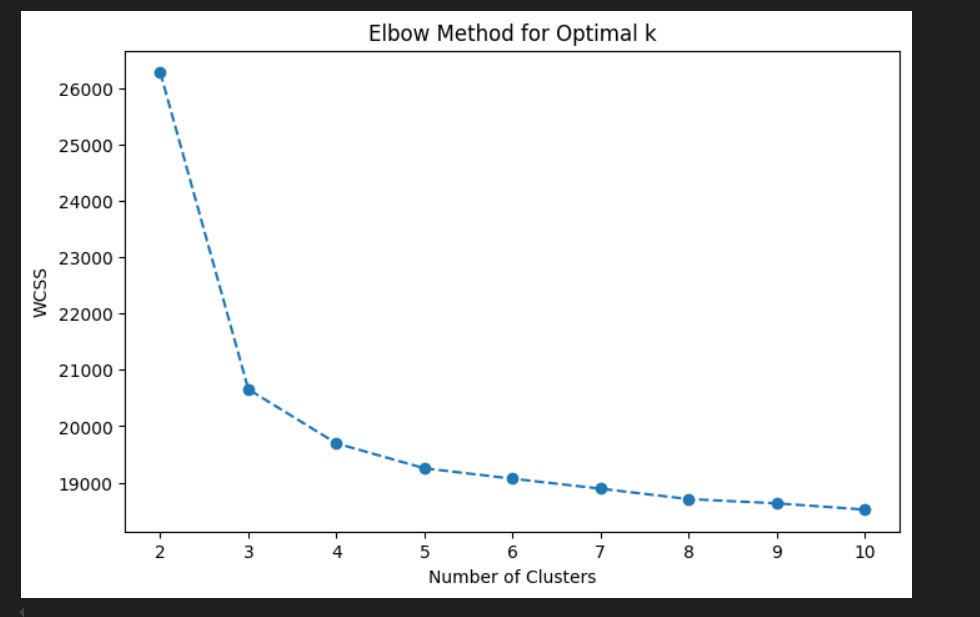
The Elbow Method is a heuristic approach to determine the optimal number of clusters (K). It involves:

* + Running K-Means for a range of cluster values (K).
  + Calculating inertia for each value.
  + Plotting an Elbow Curve, where the point of inflection suggests the optimal number of clusters.

Here the following diagram of Elbow Method:

1. axis: Represents the number of clusters (k), ranging from 2 to 10.

Y-axis: Represents the Within-Cluster Sum of Squares (WCSS), which measures the variance within each cluster.



### Observations:

* + Decrease in WCSS: As ( k ) increases from 2 to 10, the WCSS decreases, indicating that the model is capturing more variance with additional clusters.
  + Elbow Point: The significant drop in WCSS occurs until ( k = 3 ). After this point, the reduction in WCSS becomes less pronounced, suggesting that adding more clusters yields diminishing returns.

### Optimal k:

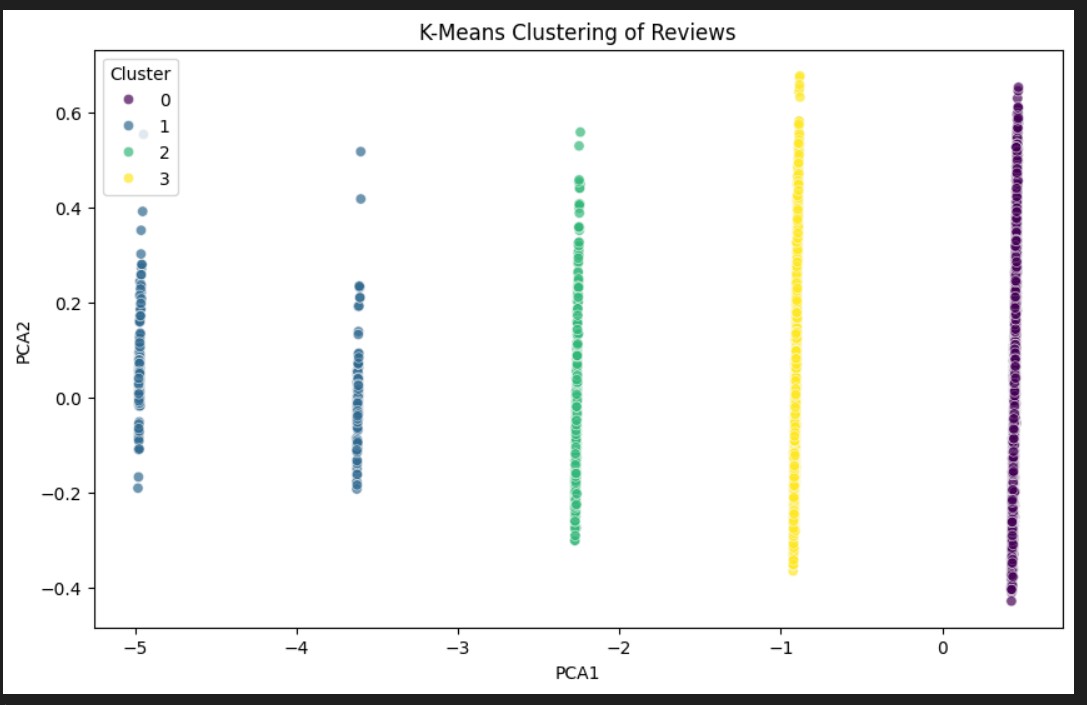
* + Identified Value: The optimal value of ( k ) is at the "elbow" of the curve, which appears to be around k = 3. This balance indicates effective clustering without overfitting the model with unnecessary complexity.

After applying K-means clustering with the optimal number of clusters (3) determined through the elbow method, the data was transformed using PCA to reduce it to two dimensions. This allowed us to visualize the clusters effectively. However, the results indicated a problem:

**Principal Component Analysis (PCA)** is a powerful technique used to reduce the dimensionality of datasets while preserving as much variance as possible. In this analysis, we aimed to condense the information from high-dimensional features into a lower-dimensional space to better visualize and interpret the data.

### Reduced Dimensions and Clustering:

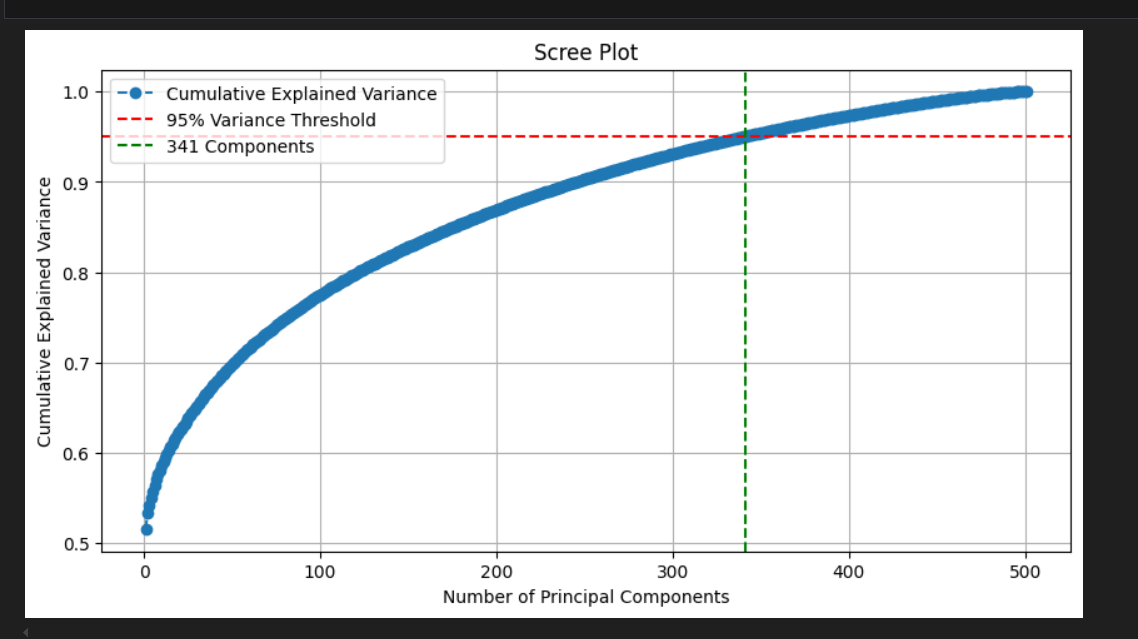
the PCA transformation resulted in the first component (PCA1) capturing approximately 51.5% of the variance while the second component (PCA2) contributed only about 1.87%. This heavily skewed distribution suggested that PCA had limitations in revealing meaningful patterns across the clusters, particularly since the clusters appeared almost vertically aligned.



**Significance of Explained Variance:**

The cumulative explained variance after the PCA indicated that only about 53.4% of the total variance was captured in the two principal components. This is relatively low, suggesting significant information loss, which is a concern for effective analysis. Typically, a good PCA model would capture at least 90% of the variance.

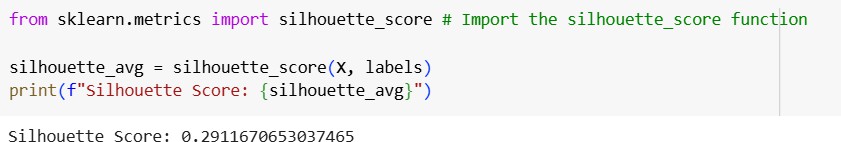
Further analyses showed that even increasing the number of principal components to five only raised the explained variance marginally to about 56%. This gradual increase highlighted that subsequent components did not contribute meaningful insights, raising questions about the suitability of PCA for this dataset.



The **Scree Plot** provides a visual representation of the explained variance versus the number of principal components. In our analysis, the plot illustrated the cumulative explained variance on the y-axis against the number of components on the x-axis. Key observations from the Scree Plot include:

* + **Cumulative Explained Variance**: The plot reveals that the cumulative explained variance steadily rises but flattens out quickly. The first few components collectively explain a significant amount of variance while additional components contribute marginally.
  + **Variance Threshold:** A dashed red line is drawn at the 95% variance threshold, indicating where we ideally want the cumulative variance to reach. However, the plot indicates that even after including a higher number of components, the variance remains below the desired threshold.
  + **341 Components:** A vertical dashed line indicates that 341 components were necessary to reach this threshold. The implication is clear: the original dataset is complex and diverse, requiring more components to capture meaningful variance.

### Clustering Method Evaluation:



The observed **silhouette score of approximately 0.291 suggests weak clustering**.

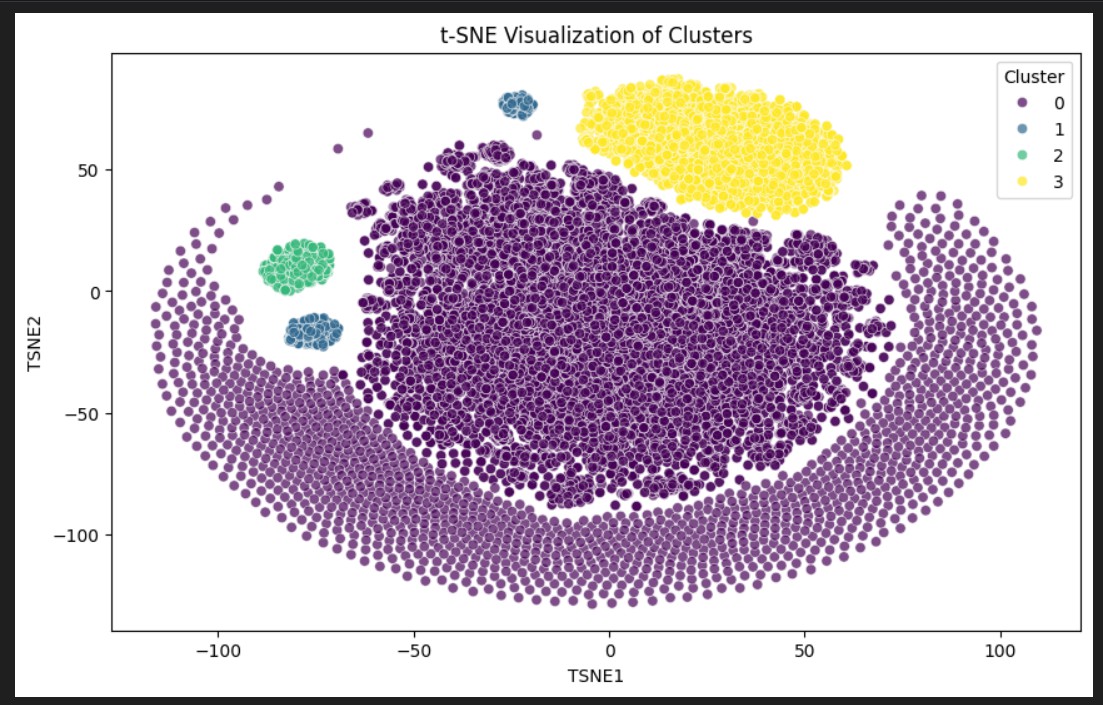
This may indicate that K-means clustering might not be the best approach due to its sensitivity to feature scales, particularly if ratings overpower text-based features.

The requirement for a high number of components to capture a more substantial amount of variance calls for a reconsideration of the dimensionality reduction approach. Alternatives such as t-SNE or UMAP may provide superior insights for visualizing high-dimensional data, and exploring different clustering methods might yield more reliable groups that better represent the underlying data structure.

**T-SNE for Better Visualization:**

t-distributed Stochastic Neighbor Embedding (t-SNE) is a popular data visualization technique primarily used for reducing the dimensionality of high-dimensional datasets while preserving their structure.

It converts similarities between data points to joint probabilities, focusing on minimizing the divergence between these probabilities in high and low-dimensional spaces.



* TSNE1: Represents the first dimension of the t-SNE embedding.
* TSNE2: Represents the second dimension of the t-SNE embedding.

**Cluster Separation**: The plot shows how well-separated the clusters are; distinct groups can indicate that K-means clustering was effective in identifying patterns in the data. Data Distribution: The overall shape and distribution of clusters can provide insights into the properties of the data itself (e.g., density, shape). Outlier Identiﬁcation: If certain points are isolated from the main clusters, they might be outliers or noise in the data.

**Implications**

* The separation of clusters hints at significant distinctions in feature space among the data points.
* The t-SNE method has successfully visualized high-dimensional data into two dimensions, maintaining the structure of clusters for better interpretability.

After modifying our clustering approach by replacing k-Means with t-SNE for dimensionality reduction, we performed a detailed analysis of the cluster characteristics using the sentiment of reviews.

**Cluster Characteristics:**

1. Cluster 0:
   * Top Keywords: This cluster features keywords such as "clorox," "clean," "wipes," and "review," indicating a strong association with cleaning products.
   * Sentiment: It exhibits a highly positive sentiment with an average score of approximately 0.38. This cluster is dominant, comprising a significant number of reviews (15,915).
2. Cluster 1:
   * Top Keywords: This cluster contains keywords like "disappointed," "movie," and "windex," suggesting a focus on products that could lead to customer dissatisfaction.
   * Sentiment: The average sentiment score is low, around 0.05, making it the most negative cluster with only 558 reviews.
3. Cluster 2:
   * Top Keywords: Keywords in this cluster include "better," "ok," and "funny." The presence of positive and neutral terms suggests mixed reviews.
   * Sentiment: It has a moderate sentiment score of about 0.24, with a total of 606 reviews.
4. Cluster 3:
   * Top Keywords: Keywords such as "love," "product," and "movie" signify a positive sentiment towards reviews related to both cleaning products and movies.
   * Sentiment: This cluster's average sentiment score is 0.33, and it includes 3,558 reviews, indicating a generally favorable view despite a smaller number of reviews compared to Cluster 0.

**Sentiment Per Cluster :** Overall sentiment analysis revealed the following average sentiment scores for each cluster:

* Cluster 0: 0.3776 (High positivity)
* Cluster 1: 0.0479 (Low negativity)
* Cluster 2: 0.2366 (Neutral/mixed)
* Cluster 3: 0.3320 (General positivity)

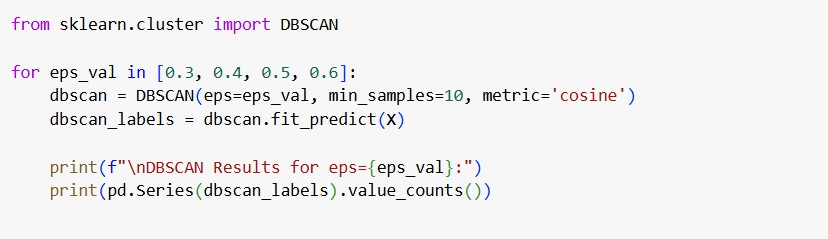
Given that Cluster 0 is overly dominant and primarily focused on cleaning products, we are considering applying DBSCAN for further refinement. This approach may help in identifying more nuanced groupings within the data, allowing for a more balanced representation of clusters.

### DBSCAN (Density-Based Spatial Clustering of Applications with Noise) :

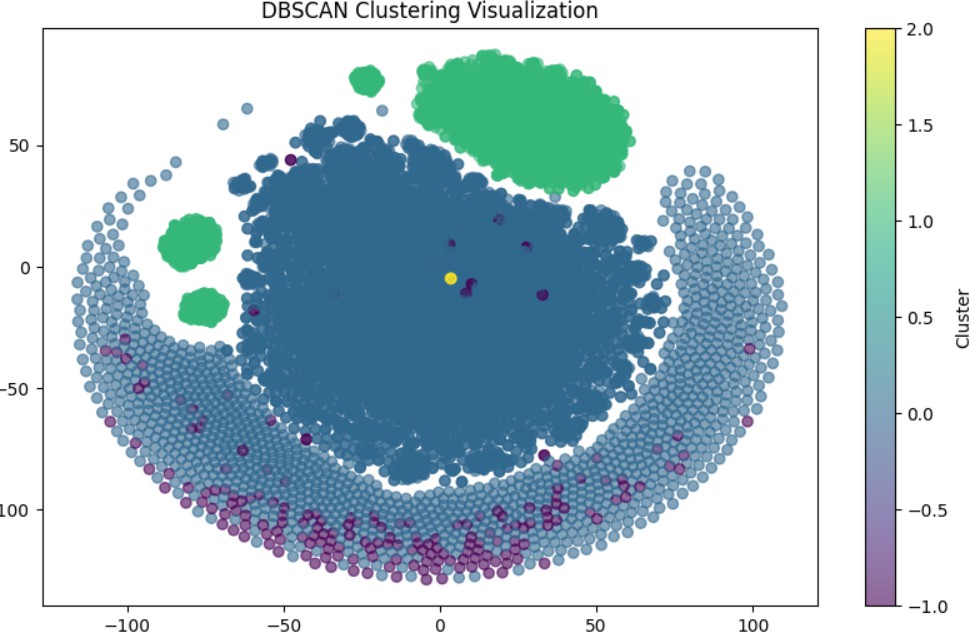
* Unlike K-Means, **DBSCAN** does not require specifying the number of clusters in advance.
* It groups together points that are closely packed while identifying outliers as noise.
* Used to detect anomalies and understand cluster structures better.

Here are the steps taken during the analysis:

1. **Initialization of DBSCAN**: We begin by importing the necessary library, DBSCAN from sklearn.cluster, and prepare to test different values for the eps parameter, which defines the maximum distance between two samples for them to be considered as in the same neighborhood.
2. **Clustering with Different Epsilon Values**: We iterate through a list of eps values: 0.3, 0.4, 0.5, and 0.6. For each value, we initialize the DBSCAN algorithm with min\_samples set to 10 and the distance metric set to 'cosine'. We then fit the model to our dataset X and predict the cluster labels.



1. **Output of Clustering Results**: After fitting the model for each eps value, we print the results, which include the count of points in each cluster. The results show how the number of clusters and the distribution of points change with different eps values:
   * For eps=0.3, we observe multiple clusters with a significant number of outliers.
   * For eps=0.4, the number of clusters decreases, and the number of outliers also reduces.
   * For eps=0.5, we see a more defined clustering with fewer outliers.
   * For eps=0.6, almost all points are assigned to a single cluster, indicating a loss of granularity.
2. **Choosing the Optimal Epsilon**: Based on the results, eps=0.5 is selected as the optimal value, as it provides a good balance between the number of clusters and the identification of noise. We store the cluster labels in a DataFrame for further analysis.
3. **Cluster Distribution Analysis:** We print the distribution of the clusters, which reveals:
   * Cluster 0 contains 15,678 points, primarily consisting of reviews for cleaning products like Clorox wipes.
   * Cluster 1 has 4,722 points, representing a mix of cleaning products and movies.
   * Cluster 2 is a small cluster with only 41 points, related to food reviews.
   * There are 196 outliers classified as noise.
4. **Sentiment Analysis:** We compute the average sentiment for each cluster, revealing that:
   * Cluster 0 has a high average sentiment score of 0.380, indicating positive reviews.
   * Cluster 1 has a neutral-positive sentiment score of 0.286.
   * Cluster 2 shows a low sentiment score of 0.053, suggesting mostly neutral opinions.
   * The noise points have a slightly negative sentiment score of 0.231.
5. **Cross-Tabulation with Existing Clusters:** We create a cross-tabulation to compare DBSCAN clusters with existing clusters, highlighting that Cluster 0 from DBSCAN aligns closely with a specific K-Means cluster, while other DBSCAN clusters represent a mix of K-Means clusters.
6. **Visualization of Clusters:** A visualization is created to depict the clusters and their densities. The central cluster is shown in blue, indicating a high density, while smaller clusters are represented in green. Noise points are highlighted in lighter colors, such as pink, to indicate their classification as outliers.



1. **Interpretation of Results:** The visualization effectively communicates the distribution of clusters, showcasing DBSCAN's ability to identify non-convex shapes and varying densities. The presence of noise points suggests that the algorithm is functioning correctly, as it distinguishes unique reviews that do not fit into any cluster.

# Topic Modelling

Topic modeling is a natural language processing (NLP) technique used to identify topics within a corpus of text data. It helps in uncovering hidden themes in large text datasets. In this project, we use Latent Dirichlet Allocation (LDA), a probabilistic model that assumes each document is a mixture of topics and each topic is a distribution of words.

**Preprocessing Text Data**: Before applying LDA, text data undergoes preprocessing to enhance the accuracy of the model. The key steps include:

* + **Tokenization**: Splitting text into individual words.
  + **Stopword Removal**: Eliminating common words like "the," "is," and "and" that do not contribute to the topic.
  + **Lemmatization**: Converting words to their base form (e.g., "running" to "run").
  + **Document-Term Matrix Conversion**: Using **TF-IDF** (Term

Frequency-Inverse Document Frequency) or **CountVectorizer** to convert text data into a numerical format.

### Applying NMF:

**Non-Negative Matrix Factorization (NMF)** is another topic modeling technique. Unlike LDA, which is probabilistic, NMF is a matrix factorization method that assumes non-negative elements.

* + NMF decomposes the document-term matrix into two matrices: one representing topics as word distributions and another representing documents as topic distributions.
  + It is particularly useful for **shorter texts** and datasets where topics are more distinct.

### Applying LDA:

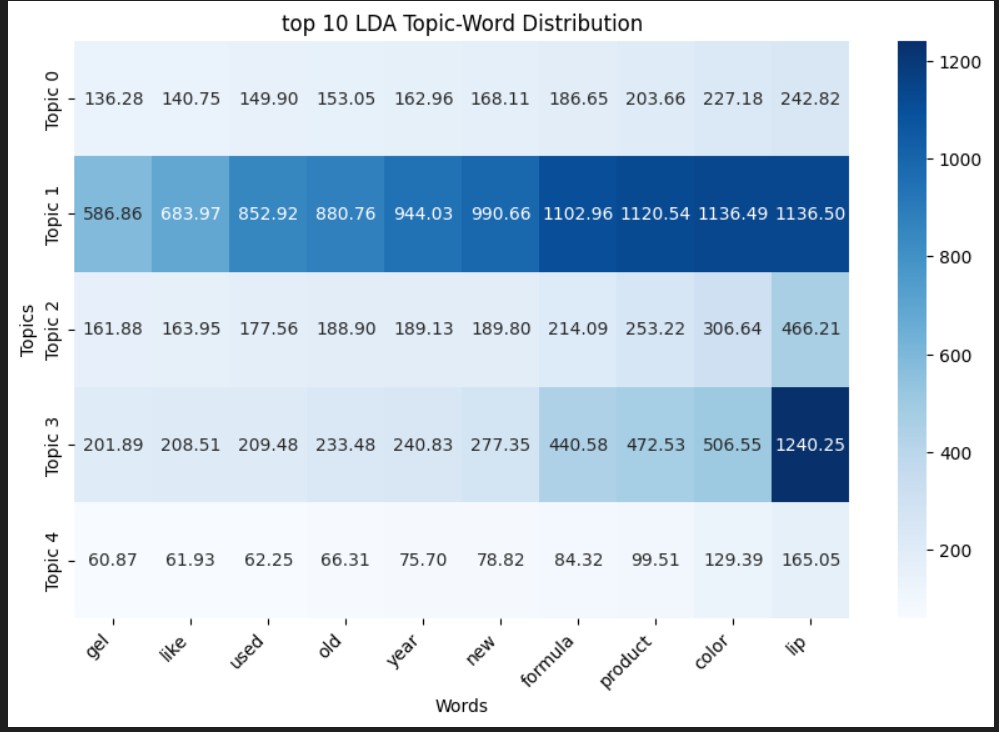
Once preprocessing is complete, LDA is applied to extract topics:

1. **Defining the Number of Topics**: The model assumes a predefined number of topics.
2. **Fitting the Model**: The LDA model is trained on the document-term matrix to learn topic distributions.
3. **Extracting Topic Keywords**: The model identifies the most important words for each topic, which represent the underlying themes.

### Visualizing Topics

Topic modeling results can be visualized to better interpret the discovered topics. Common visualization methods include:

* + **Word Clouds**: Displaying the most significant words in each topic with larger fonts indicating higher importance.
  + **Topic Distributions**: Representing the proportion of different topics within each document.



LDA Topics Analysis Topic 1 (Lipstick & Makeup): Focuses on beauty products, lip colors, and formula.

Topic 2 (Cleaning & Clorox): Talks about Clorox wipes, cleaning, and promotions.

Topic 3 (Hair & Skincare): Discusses hair conditioners, shampoo, skin care, and product feel.

Topic 4 (Movies & Entertainment): Covers movies, entertainment, and action ﬁlms.

Topic 5 (Home & Appliances): Includes vacuums, seasonal ﬂu, and home essentials.

LDA captures broad themes with some mixed topics (e.g., ﬂu + vacuum + movies in Topic 5).



NMF Topics Analysis Topic 1 (Cleaning & Clorox): Talks about Clorox, cleaning, and disinfecting.

Topic 2 (Movies & Entertainment): Focuses entirely on movies, action, and family entertainment.

Topic 3 (Haircare & Shampoo): Covers hair products, conditioners, and how they feel.

Topic 4 (General Product Reviews & Cleaning): Discusses product effectiveness, price, and taste.

Topic 5 (Household Cleaning & Wipes): Focuses speciﬁcally on wipes, kitchen, bathroom, and disinfecting.

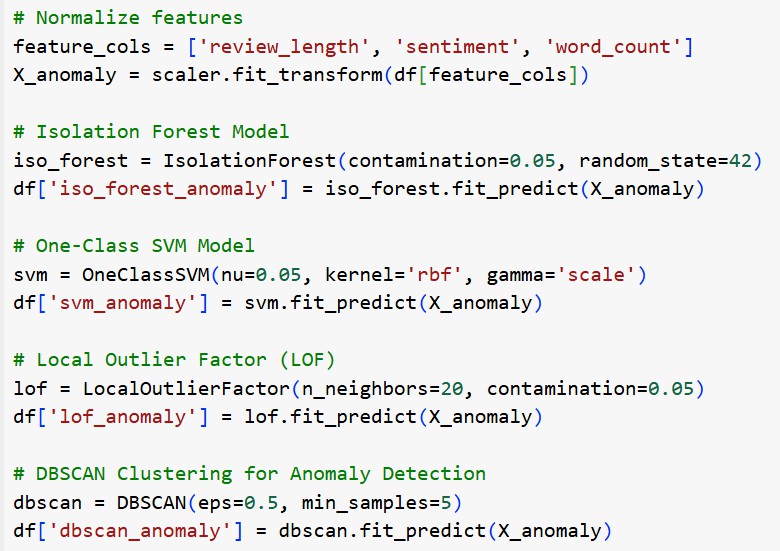
NMF produces sharper, more distinct topics, making it better for structured analysis.

The word clouds show the most relevant words for each topic, helping to categorize themes in customer reviews. The topic distribution charts indicate how different topics are spread across reviews.

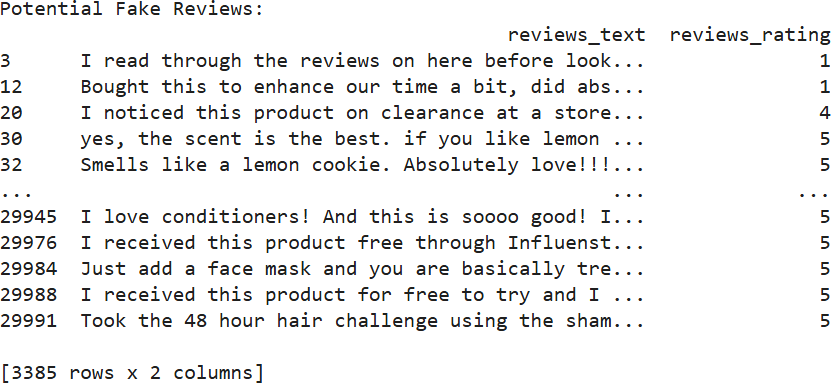
# Anomaly Analysis and Detection

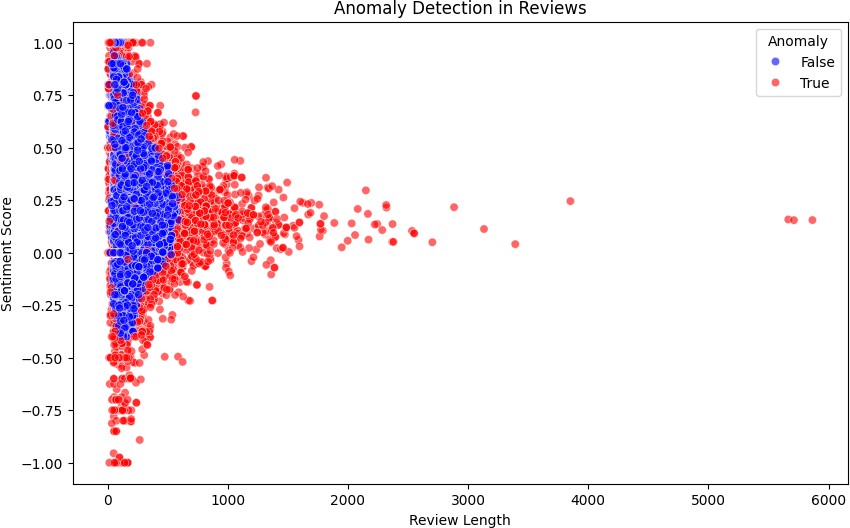
Anomaly detection helps identify rare, unusual, or fraudulent instances in a dataset. We used Isolation Forest, an unsupervised algorithm that isolates anomalies by building random decision trees, one class SVM as well as DBSCAN to identify potential or fake reviews.

To begin with, we extracted 3 new columns from review\_text, i.e review\_length, sentiment and word\_count, and we normalized the features. Then we’ve applied isolation forest, one-class SVM and local outlier factor (LOF) for producing an anomaly score. DBSCAN too helped in detecting anomaly points.



Ultimately, we plotted a graph to visualize the relationship between ‘review\_length’ and ‘sentiment\_score’. The potential fake user reviews are





As seen in the scatter plot, the shorter or longer reviews are marked as anomalies. Extreme sentiment scores are also most likely to be anomalous. Heatmap for the same shows

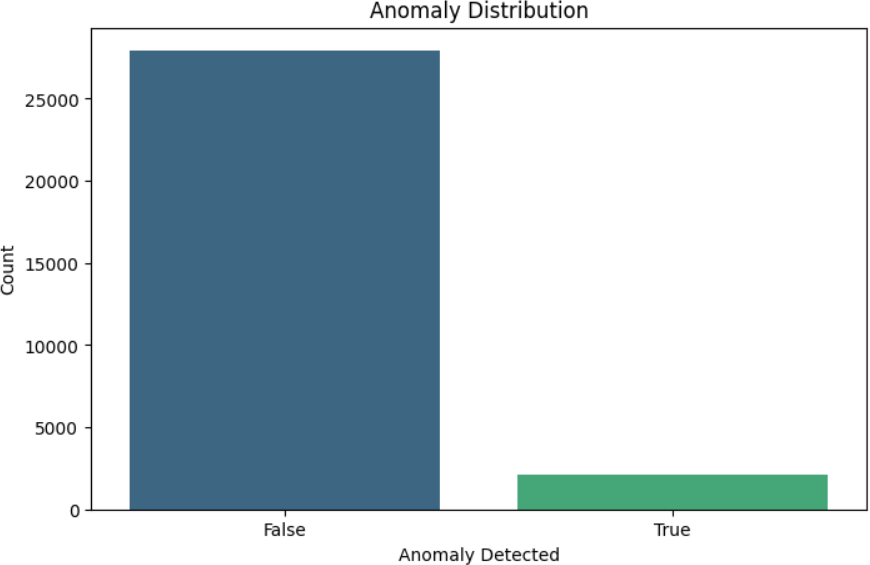


The heatmap reveals that while review\_length and word\_count are perfectly correlated, the sentiment feature has weak negative correlations with both. This suggests that while longer reviews tend to contain more words, longer reviews may result in slightly lower sentiment ratings.

Another method, with which we’ve implemented anomaly detection is by aggregating user behaviour and normalizing it before implementing isolation forest, LOF and DBSCAN.



The resulting anomalous users were found to be 1244 by isolation forest, which is because isolation forest works by isolating points based on random splits. LOF has a moderate count of 377 anomalous users and DBSCAN has fewer anomaly users of 14.



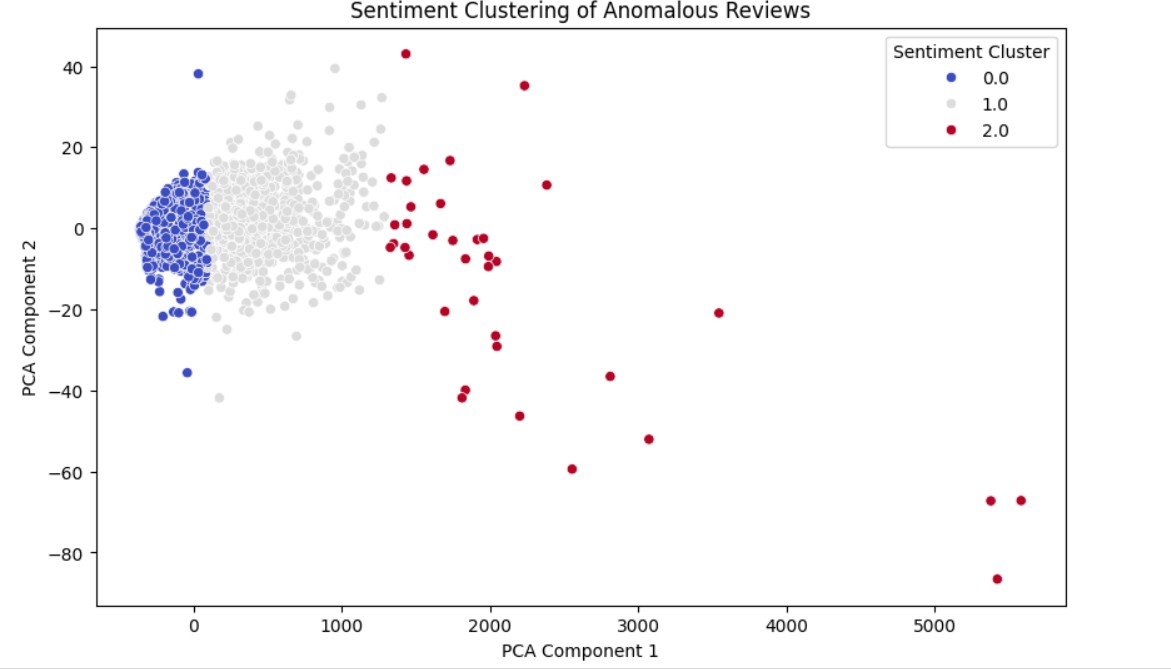
### Anaomly detection using Kmeans with PCA

The scatter plot depicts the clustering of sentiments associated with anomalous reviews. It utilizes Principal Component Analysis (PCA) to reduce the dimensions, showcasing the distribution of reviews in a two-dimensional space.

Color Legend: The data points are color-coded to represent different sentiment clusters. Blue points indicate a sentiment cluster of 0.0, gray points represent 1.0, and red points signify 2.0. This differentiation helps in visualizing the sentiment trends among the reviews.

PCA Components: On the x-axis, PCA Component 1 captures a significant variance of the data, while the y-axis shows PCA Component 2. The clustering suggests that there are distinct groups of reviews with varying sentiments, primarily set apart by the red and blue clusters.

Insights: The concentration of blue points on the left indicates a predominant neutral sentiment, while the sparse distribution of red points on the right highlights a less frequent but highly negative sentiment. The overall distribution of gray points suggests a moderate sentiment in between.



# Recommendation Systems

The recommendation system for our model was developed with 2 methods.

### Using cosine similarity and tf-idf features + Nearest neighbours

First we encoded usernames and product names into unique numerical IDs with Label Encoder.

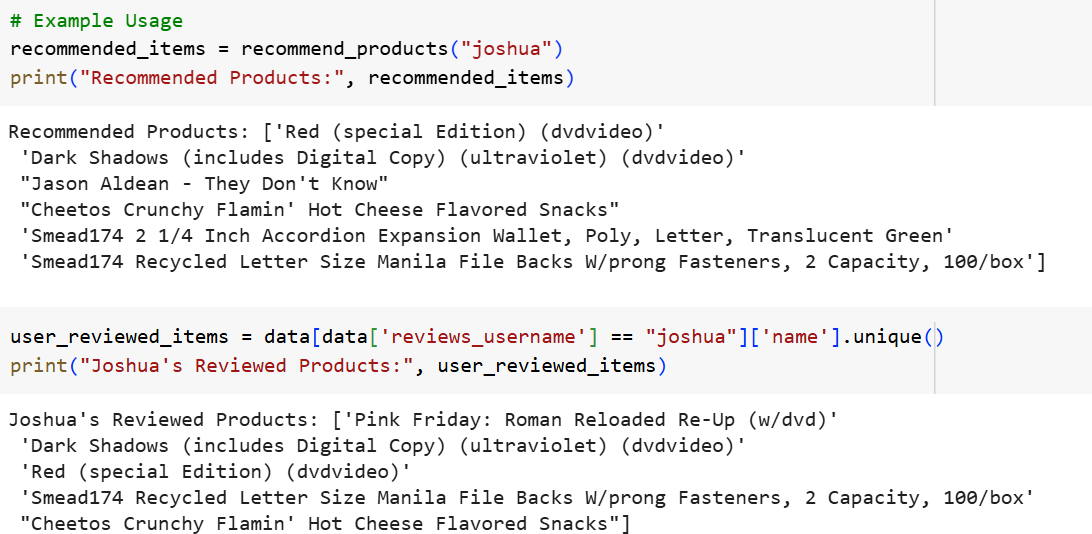
Then we converted it into a TF-IDF Vector and trained it to Nearest Neighbour Model, where we will use cosine similarity metric for building recommendation function. Cosine similarity is used to measure the similarity between two vectors.



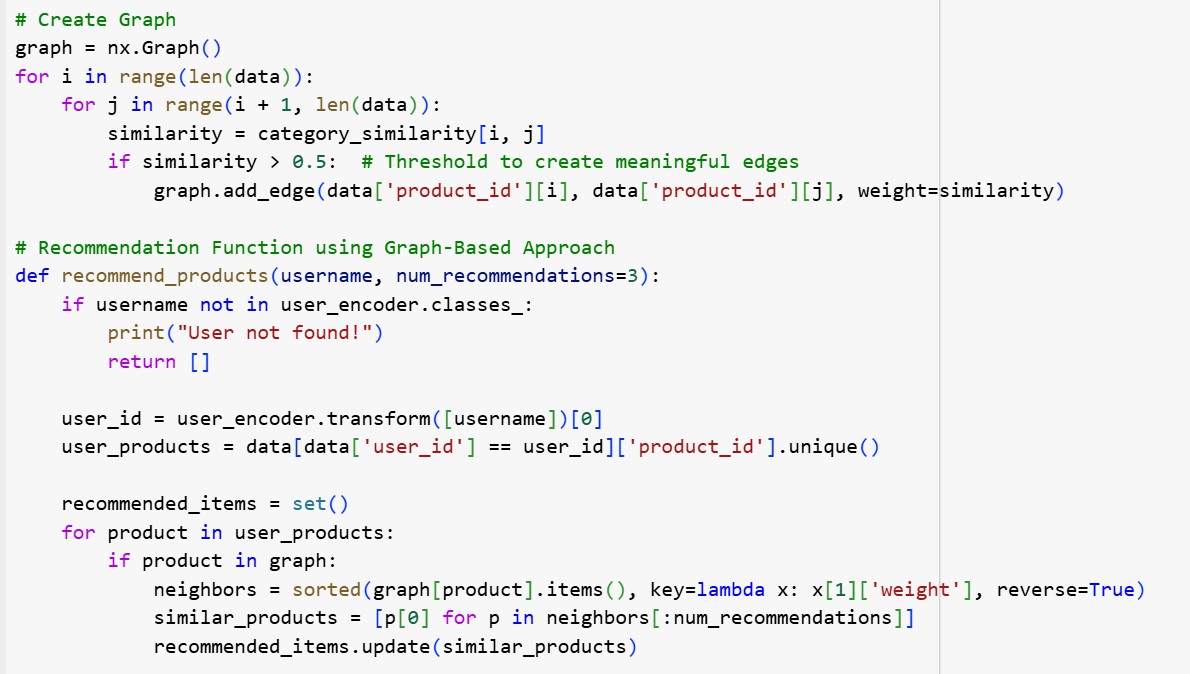
The steps within the function are as follows:

* Takes a username and finds the corresponding user ID.
* Finds products the user has interacted with.
* Uses the Nearest Neighbors model to find similar products based on their TF-IDF category vectors.
* Returns a list of recommended product names.

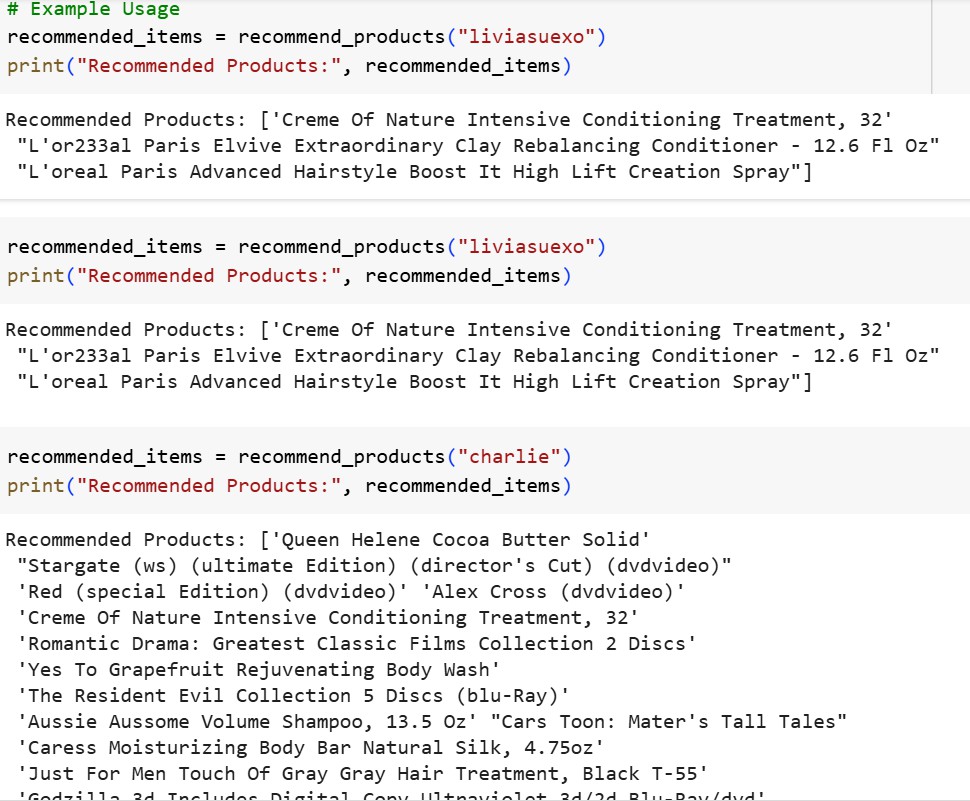
We can check the functionality by calling a username, to see the recommended products.



### By using Graph nodes

In this method we have used a Word2Vec model, which is an NLP technique for obtaining vector representations of words. We’ve used it to get category embeddings for which we have taken out cosine similarity.

Here, we are implementing a graph based approach, for finding recommendations, so that when a username is given, it will find products interacted with, find similar products in the graph and recommend the top matched ones.



The number of nodes and edges performed in the graph were, Number of Nodes: 271

Number of Edges: 12525

**Conclusion**

By integrating machine learning and NLP techniques, we successfully enhanced the accuracy and reliability of our product recommendation system. The combination of clustering, anomaly detection, and collaborative filtering ensures that users receive trustworthy and personalized recommendations, ultimately improving their overall experience. This project highlights the power of data analytics in tackling real-world challenges and optimizing digital platforms for e-commerce marketplaces, improving the experience of customers.

The links for this project are mentioned below:

[https://drive.google.com/drive/folders/13z3cX3sM5G5vZ49gjybkPJQeqQBYZXDu](https://drive.google.com/drive/folders/13z3cX3sM5G5vZ49gjybkPJQeqQBYZXDu?usp=sharing)

[?usp=sharing](https://drive.google.com/drive/folders/13z3cX3sM5G5vZ49gjybkPJQeqQBYZXDu?usp=sharing)