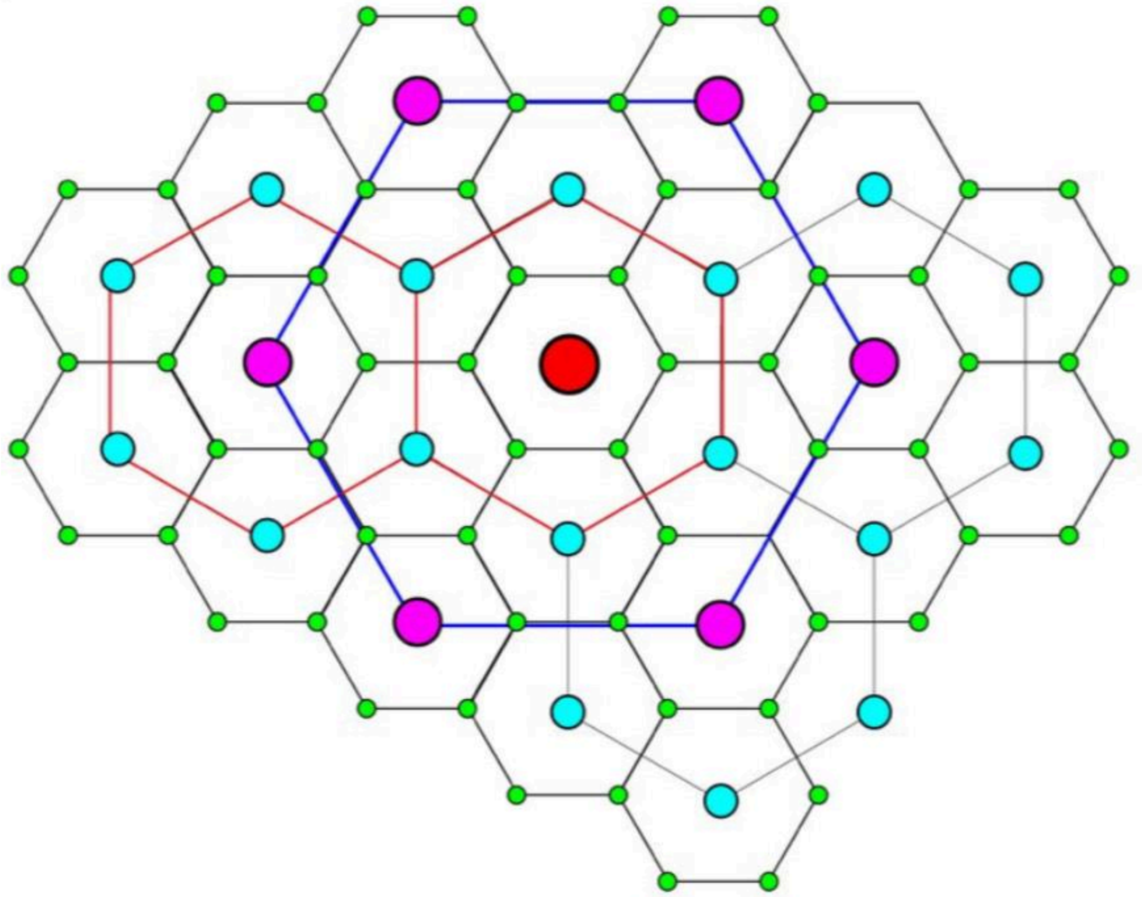
الأكاديمية السعودية الرقمية
SAUDI DIGITAL ACADEMY

NLP Project
Automated Customer Reviews



Team Members:
Alhanouf Aljamaan - Dana AlOmari

1. Introduction

Project Overview

This project aims to develop an end-to-end NLP pipeline that automatically processes and analyzes customer product reviews. The system performs three main tasks:

1. Sentiment Classification of customer reviews.
2. Product Category Clustering.
3. Review Summarization & Recommendation Generation.

The final objective is to assist customers in making better product decisions by generating human-like blog-style recommendation summaries.

Datasets Used

- Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv
- Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products_May19.csv
- 1429_1.csv

All datasets were merged and cleaned. Ratings were mapped to three sentiment classes:

- 1-2 stars → Negative (0)
- 3 stars → Neutral (1)
- 4-5 stars → Positive (2)
-

To ensure balanced training, we manually limited the positive class to 2000 samples, while keeping all neutral and negative samples.

Additionally, a new column called Cluster_Category was created to store **manually named clusters**

2.Sentiment Classification

sentiment classification was employed to analyze customer review content and determine whether the feedback is positive or negative. This section provides an overview of customer satisfaction levels based on the textual review content.

Technology Used

- Model used: pipeline("sentiment-analysis") from the Transformers library
- The model outputs two primary labels: "POSITIVE" or "NEGATIVE" or “NEUTRAL”
- Sentiment analysis is applied to the first 5 reviews per product to balance speed and accuracy

How It Works

1. Five sample reviews are selected for each product.
2. These reviews are passed to the sentiment classification model.
3. The number of positive and negative sentiments is calculated.
4. The final percentage of each sentiment type is presented.

Preprocessing Steps:

- Combined all datasets.
- Removed null/empty values.
- Applied mapping of star ratings to sentiment labels.
- Used stratified split and manual balancing.
-

Model:

- Pretrained Model: bert-base-uncased
- Tokenizer: Bert Tokenizer Fast
- Framework: HuggingFace Transformers

Training Setup:

- Optimizer: Adam W
- Batch Size: 8
- Epochs: 3
- Learning Rate: 2e-5
- Early Stopping: patience=2
- Manual Class Balancing (positive samples limited to 2000)

Evaluation:

	Negative	Neutral	Positive
precision	0.8785	0.7613	0.8198
Recall	0.8207	0.8124	0.8075
F1 - Score	0.8486	0.7860	0.8136

Confusion Matrix was generated for visualizing class-wise performance.

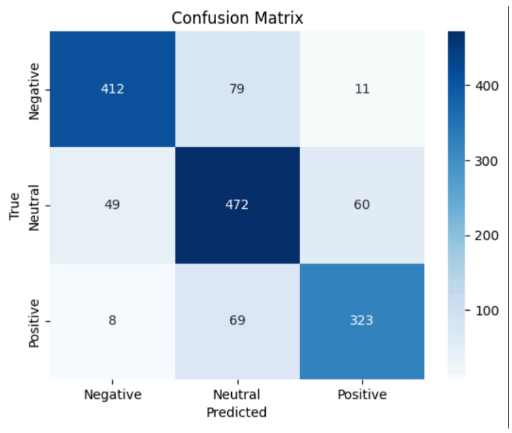


Figure 1. Sentiment Classification Confusion Matrix

The sentiment classification system proved highly effective in automatically extracting general impressions from customer reviews, aiding in assessing product quality based on real-world user experience.

3.Product Category Clustering

clustering techniques were used to group products based on similar review patterns, aiming to simplify the analysis process and identify general trends within product categories.

Technology Used

Clustering was based on keyword matching within the Cluster_Category column.

Products were grouped into categories such as

"Supplies", "Electronics", "Health", and others.

Some categories were manually adjusted (e.g., beauty → Health) to improve classification accuracy.

How It Works

1. The Cluster_Category column is read from the review dataset.
2. Cleaning operations are performed (e.g., removing missing values).
3. Partial word matching is used to standardize category names.
4. These categories are used to filter and analyze products within each cluster.

Model:

- Embedding Model: Sentence Transformer (MiniLM)
- Clustering Algorithm: KMeans
- Tools: scikit-learn, matplotlib, seaborn

Method:

- Encoded product categories using MiniLM.
- Applied Elbow Method to determine optimal k.
- Used Silhouette Score for cluster quality evaluation.
- Cluster labels were manually renamed and saved into a new column Cluster_Category for reuse.

Clustering Outcome:

Grouped products into 6 meaningful clusters:

- Supplies
- Electronics
- H Electronics
- Tablets
- Batteries
- Computer Accessories

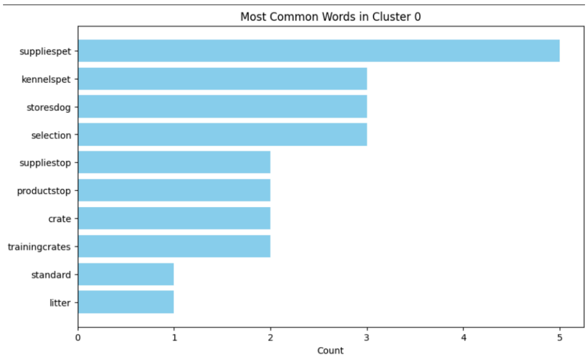


Figure 2 . Cluster 0

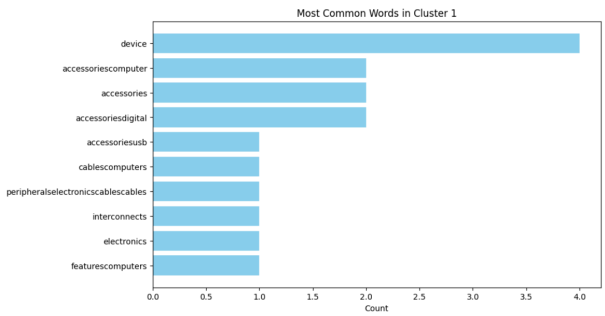


Figure 3 . Cluster 1

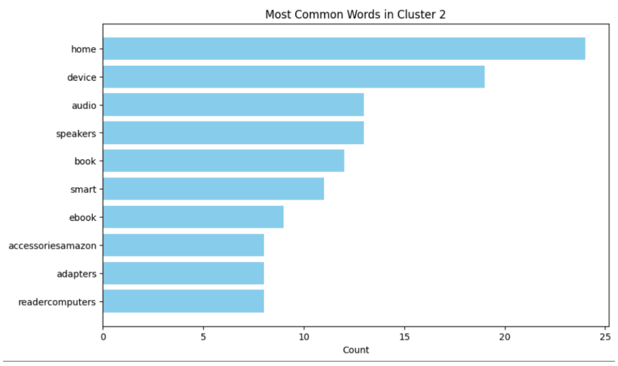


Figure 4 . Cluster 2

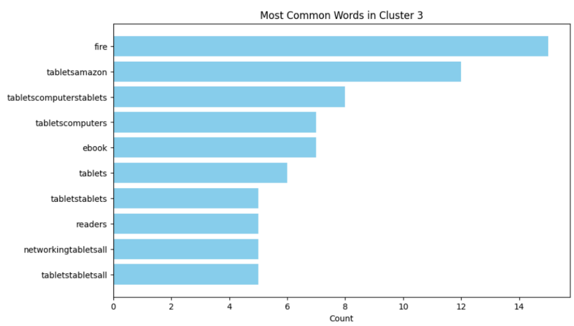


Figure 5 . Cluster 3

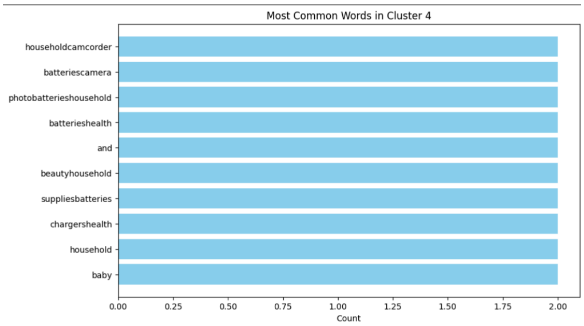


Figure 6 . Cluster 4

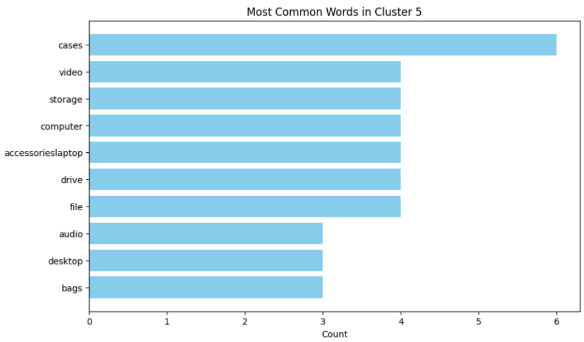


Figure 7 . Cluster 5

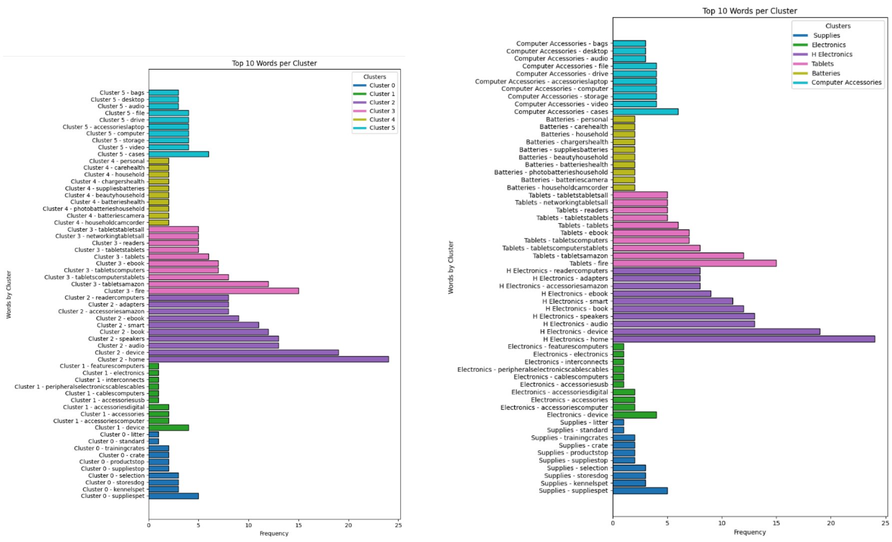


Figure 8 . matplotlib Cluster 6 clustering

These were later used in the summarization UI and analysis.

Clustering contributed to more effective data organization and analysis by grouping products with shared characteristics, making it easier to explore ratings and trends within each category

4. Generative Summarization (Product Recommendations)

summarization techniques were applied to extract the core content of long textual reviews, providing a brief and comprehensive overview of customer opinions for each product.

Technology Used

- Model used: facebook/bart-large-cnn
- Library: Transformers by Hugging Face
- Utilizes pipeline("summarization") for direct summarization
- The first 10 reviews for each product are merged and summarized as one block

How It Works

1. The first 10 reviews per product are selected.
2. They are combined into a single block (up to 1024 characters).
3. The text is passed to the summarization model.
4. Output length is controlled between 30 and 150 words.
5. The result is shown as a concise summary representing the reviews

Model:

- Model Used: facebook/bart-large-cnn
- Tool: HuggingFace pipeline("summarization")

Summary Structure:

- Top 3 Products + Key Differences
- Top Complaints for Each Product
- Worst Product & Reason to Avoid

Automatic summarization transformed lengthy reviews into quickly digestible insights, enhancing users' ability to analyze products and make informed decisions more efficiently.

5.Deployment

aims to provide an interactive interface that enables users to upload review files and analyze them seamlessly using pre-trained models. **Gradio** was used to create the interface and deliver a streamlined user experience

Tools & Technologies

- Programming Language: Python
- Deployment Library: **Gradio**
- AI Models: Hugging Face Transformers (BART, Sentiment Classifier)
- Supports uploading and processing of CSV files
- Can run locally or online using share=True

How Deployment Works

1. The app is executed locally using gr.Interface.
2. Pre-trained models are loaded at the start of the script.
3. When a user uploads a CSV file, it is passed to the processing function.
4. Results (summarization, sentiment, etc.) are displayed directly in the UI.
5. The app can be shared publicly via a temporary link if share=True is enabled

Web Application (Gradio)

- A unified web interface was built using Gradio.
- Features included:
  - Uploading a CSV file
  - Choosing a cluster category (from Cluster_Category)
  - Automatically displaying summarized blog-style recommendations
- Models (BERT for classification, MiniLM for embedding, BART for summarization) were pre-loaded.
- The Gradio app could be launched locally or shared via a hosted link.

Final Output Options:

- Web UI using Gradio
- Exported text/CSV recommendations
- Presentable blog-style summaries

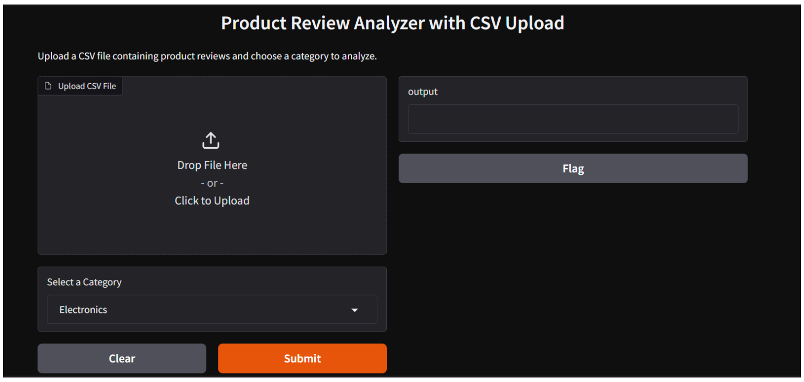


Figure 9 . Deployment Gradio interface

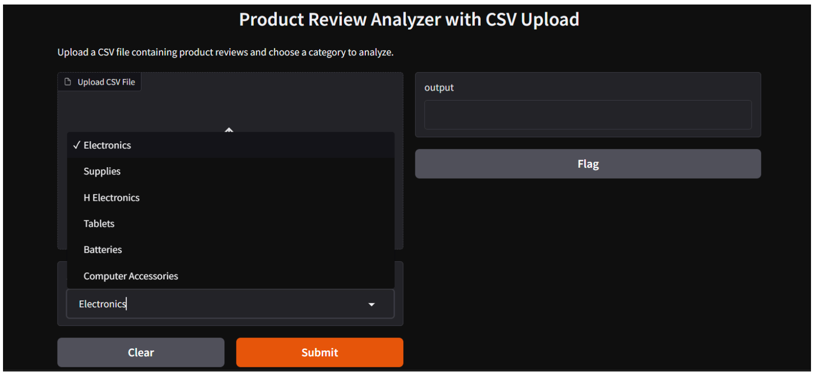


Figure 10.. Deployment Clustering

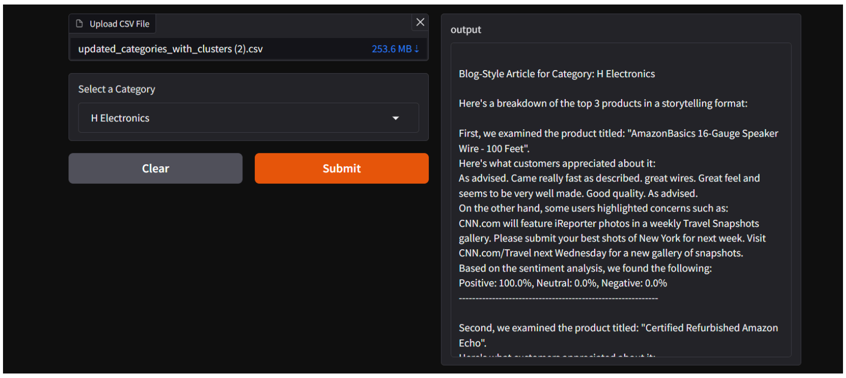


Figure 11.. Deployment Summarization

Using Gradio simplified the deployment process by transforming the model from a prototype script into an interactive analytical tool that anyone can use without technical expertise. This greatly enhanced accessibility and user experience.

6.Conclusion

This project successfully implemented a complete Natural Language Processing (NLP) pipeline that integrates three key tasks: sentiment classification, review summarization, and clustering-based topic discovery. By leveraging state-of-the-art transformer models such as BERT and BART, we were able to extract meaningful insights from customer reviews at scale.

The classification model provided accurate predictions of user sentiment based on textual inputs, while the summarization model generated concise and human-readable overviews of product feedback. Additionally, clustering techniques allowed us to group similar reviews and uncover hidden patterns within the data.

Overall, this system enhances the ability of businesses and consumers to make informed decisions by transforming unstructured review data into structured, actionable insights.

7. Future Work

While the current implementation achieved its objectives, several enhancements can be explored to improve system performance and applicability:

1. Expand Dataset Variety: Incorporate more diverse and multilingual review datasets to increase the generalizability of the models.

2. Fine-Tuning Pretrained Models: Instead of relying solely on default Hugging Face pipelines, fine-tune models such as BERT and BART on domain-specific review data to boost accuracy.

3. Sentiment Granularity: Introduce multi-class sentiment classification (e.g., very negative, negative, neutral, positive, very positive) for more detailed analysis.

4. Topic Modeling Integration: Combine unsupervised clustering with topic modeling (e.g., LDA or BERTopic) to provide more interpretable cluster labels.

5. Summarization Optimization: Implement extractive + abstractive hybrid summarization techniques to increase fluency and factual consistency.

6. Scalability & Performance: Optimize memory and runtime performance for large datasets by applying batch processing and using efficient vector stores.

7. Deployment on the Web: Extend the current Gradio interface into a full-stack deployment (e.g., using Flask or FastAPI) for broader accessibility.

8. User Feedback Loop: Introduce a user feedback mechanism to continuously improve model predictions based on real-world inputs.