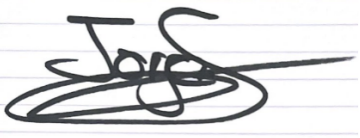



# Trend Detection for E-commerce Seller Feedback

Jorge Soldevilla Artajona

August 2025

Alumno	Director del proyecto
<p>Jorge Soldevilla Artajona</p> 	<p>Javier Rocamora García</p> 
<p>Fecha y firma: 27-08-2025</p>	<p>Fecha y firma: 27-08-2025</p>

# 1 Introduction to the Project

## 1.1 State of Art

In recent years, e-commerce has rapidly changed the way businesses operate and consumers shop. One of the most significant developments in this field has been the emergence of logistics services that eases online sales. Fulfillment by Solde (FBS) is one such model designed to help sellers manage inventory, packaging, and shipping. This approach allows sellers to focus on growing their business while relying on FBS to handle the operational complexity of order fulfillment.

The FBS business model allows individuals and businesses to sell their products through the platform, using FBS's infrastructure and without having to worry about any logistics. Sellers often use forums to share experiences, discussing various aspects of the service, such as inventory management or customer service. However, the massive volume of posts makes manual analysis difficult, highlighting the importance of automated systems that can process these comments and provide quick insights to drive action items.

## 1.2 Objectives

The main goal of this project is to design an AI system that analyzes anecdotes posted by sellers on the FBS forum. Through natural language processing and machine learning, the system will detect trending topics in seller feedback and generate actionable insights into their main concerns.

The project aims to achieve the following specific objectives:

1. **Automate Feedback Analysis:** The AI system will automate the analysis of seller anecdotes, reducing a lot the time and effort needed to review large amounts of feedback. In addition to improving efficiency, it will also enable real-time monitoring of seller sentiment.
2. **Extract Trending Topics:** Through classification and clustering topics, the AI will group seller anecdotes into specific topics and clusters. This will help to show recurring issues and new trends.
3. **Drive Insights for Improvement:** The findings from the analysis will be used by the stakeholders to help them in their decisions. By finding trends and patterns in seller feedback, Fulfillment by Solde can introduce targeted improvements to its services, strengthening both seller experience and satisfaction.
4. **Summarize Findings:** The AI system will produce short summaries of trending topics, allowing stakeholders to quickly grasp the main issues. Along with these summaries, it will generate short titles to further support rapid understanding.

### 1.3 Procedure

The pipeline for this project will consist of four key steps: classification of anecdotes into predefined topics, clustering of anecdotes within each topic, summarization of the clusters, and generation of titles for each cluster.

1. **Classification:** The first phase involves building a supervised model to classify seller anecdotes into predefined categories such as Listing, Ads & Deals, Seller Support, Fee, Inventory Management, Inbounding, and Insurance. Using a BERT-based architecture fine-tuned on a labeled dataset, the system will learn to capture context and meaning in text. Once trained, it will automatically assign new anecdotes that the model has not seen to the appropriate topic, enhancing the classification process.
2. **Clustering:** After classifying the anecdotes by topic, clustering will be applied within each topic to find out emerging themes. Using BERTopic, which combines UMAP for dimensionality reduction and HDBSCAN for clustering, similar anecdotes will be grouped together, making it possible to identify shared concerns and feedback among sellers.
3. **Summary:** Once the clustering is finished, the anecdotes of each cluster will be summarized using a Llama2 model. These summaries will capture the core ideas and sentiments, making it easier to identify the most relevant issues in each group.
4. **Title Generation:** Finally, a FLAN-T5 model will generate titles for each cluster based on the summaries generated in the previous step.

By applying this structured procedure, we will build an efficient pipeline that classifies seller anecdotes into predefined topics and drive actionable insights through clustering, summarization, and title generation.

## 2 Classification Task

### 2.1 Introduction

To effectively analyze seller feedback, it is necessary to develop a dedicated classification model capable of categorizing anecdotes into topics relevant to Fulfillment by Solde (FBS). Using a generic model would lead to suboptimal outputs, since these models do not have the context or understanding of the terminology used in FBS. By training on a labeled dataset that reflects the context and acronyms of FBS, the model can learn to identify and classify anecdotes with higher accuracy. Once fine-tuned, the model can be applied to new, unlabeled anecdotes, automating the classification process.

For the supervised training, three models were evaluated: LLaMA 2, GPT-2, and BERT. The LLaMA 2 model required extremely high computational resources, GPT-2, while easier to train, failed to deliver reliable results, as it

consistently assigned all anecdotes to a single topic. In contrast, the BERT model achieved strong performance, accurately classifying anecdotes into their respective topics. Therefore, BERT was selected for the classification task. The training dataset contained 667 anecdotes with a balanced distribution across all the topics. In the table below, it is shown example of anecdotes for each topic:

Anecdote	Topic
I set up a promotional offer last week, but I haven't seen any increase in traffic. I'm worried something went wrong with the setup.	ADS & Deals
I recently got a report showing a lower payout per item sold, but the product price hasn't changed. I need to understand where the extra deductions are coming from.	Fee
Had issues with the last inbound shipping where several items were marked as received but never Actually showed up in the inventory.	Inbounding
I've been trying to get information about the insurance coverage for damaged goods, but no one can seem to give me a clear answer.	Insurance
I noticed that the warehouse keeps mixing up my organic teas with another brand, leading to wrong shipments. Customers are getting frustrated!	Inventory Management
My product was listed under the wrong category, and now customers can't find it when they search. I've lost so many potential sales.	Listing
I've been waiting for a response for over a week about an issue with my account suspension. No update so far, and it's affecting my business.	Seller Support

Figure 1: Examples of anecdotes of each topic

## 2.2 Steps for Implementing BERT for Text Classification

To carry out the classification task, we implemented a supervised model based on BERT (Bidirectional Encoder Representations from Transformers), chosen for its ability to capture semantic context and perform well in text classification.

The first step involved initializing the BERT tokenizer to preprocess the anecdotes and mapping the textual labels into numerical indices:

```
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
label_dict = {label: idx for idx, label in enumerate(df['Topic'].unique())}
labels = torch.tensor(df['Topic'].map(label_dict).values)
```

The dataset was then split into training and validation sets and tokenized into input IDs and attention masks suitable for BERT. These were stored in TensorDataset objects and handled with DataLoader for batching.

The classification model was defined as a sequence classification BERT with the number of labels equal to the topics. Training was conducted with the AdamW optimizer and a linear scheduler:

```

model = BertForSequenceClassification.from_pretrained(
'bert-base-uncased', num_labels=len(label_dict))
optimizer = AdamW(model.parameters(), lr=1e-4, eps=1e-8)

```

The model was fine-tuned for two epochs with a batch size of 16. During training, we applied gradient clipping to stabilize learning and used a scheduler to dynamically adjust the learning rate.

Given that the dataset size is 677 and we are using a batch size of 16, we calculate the total number of batches as

$$\frac{677}{16} \approx 43$$

This means that in each epoch, the model will see approximately 43 batches of data. Since we are training for 2 epochs, the total number of parameter updates will amount to:

$$43 \times 2 = 86$$

```

epochs = 2
for epoch in range(epochs):
    model.train()
    total_train_loss = 0
    for batch in train_loader:
        batch = tuple(b.to(device) for b in batch)
        b_input_ids, b_input_mask, b_labels = batch
        model.zero_grad()
        outputs = model(b_input_ids, attention_mask=b_input_mask,
            labels=b_labels)
        loss = outputs.loss
        total_train_loss += loss.item()
        loss.backward()
        torch.nn.utils.clip_grad_norm_(model.parameters(), 1.0)
        optimizer.step()
        scheduler.step()
    avg_train_loss = total_train_loss / len(train_loader)
    print(f"Epoch {epoch + 1}/{epochs} - Training loss: {avg_train_loss:.4f}")

```

```

Epoch 1/2 - Training loss: 1.4593
Epoch 2/2 - Training loss: 0.5210

```

Figure 2: Training loss

As shown in the image above, the training loss decreases significantly in the second epoch, indicating that the model's parameters have adjusted to values that effectively minimize the loss function.

Once the training of the model is finished, we continue with the evaluation step, where we can evaluate the performance of the model to unseen data.

For evaluation, the model was tested on the validation dataset with dropout disabled (`model.eval()`), and predictions were compared to the true labels. Metrics such as accuracy, precision, recall, and F1-score were computed. The model achieved an accuracy of 92.38 %, with precision, recall, and F1-score values

above 92%. Following the completion of the evaluation, we move on to generating the accuracy report. This report is crucial as it not only provides an overall assessment of the model’s performance but also breaks down the results for each individual topic.

The classification report demonstrates that the model performs well across all topics. Even the lowest precision score of 0.78 for Seller Support is sufficiently high, indicating that the model maintains accuracy across all topics.

Finally, the fine-tuned model was applied to an unseen dataset of 300 new anecdotes, automatically classifying them into the predefined topics and demonstrating its scalability in real-world conditions.

The classification results for each topic are summarized in the image below:

**Breakdown of Predicted Topics:**

Predicted_Topic	
inventory management	89
Fee	62
Listing	55
ADS & Deals	34
inbounding	27
Insurance	18
Seller Support	15

Name: count, dtype: int64

Figure 3: Classification results for each topic

### 3 Clustering Task

The clustering task takes the analysis a step further, moving beyond simple classification to a deep dive into the anecdotes of the sellers. Although classifying the anecdotes into topics gives us an initial view of feedback, clustering helps to uncover more particular trends and patterns. By grouping similar anecdotes within each topic, we can show recurring concerns that might otherwise go unnoticed. This allows for a more precise understanding of seller challenges and provides insights that can guide targeted enhancements to the Fulfillment by Solde platform.

To do this, the anecdotes were first split by topic, so each one could be examined on its own and its specific issues addressed. Given the nature of the data, we used the BERTopic model, which combines UMAP for dimensionality reduction and HDBSCAN for clustering. UMAP reduces the complexity of the text data, while HDBSCAN forms clusters based on density, without needing to predefine their number. Standard parameters were applied, such as  $n_{\text{neighbors}} = 5$  and  $\text{min\_dist} = 0.0$  for UMAP, and  $\text{min\_cluster\_size} = 3$  and  $\text{min\_samples} = 3$  for HDBSCAN, ensuring flexibility in the formation of clusters. These clusters

gave both qualitative insights from the text and quantitative signals from the model, helping to build a clearer picture of seller sentiment and guide data-driven improvements to the platform.

```
for topic, topic_df in topic_dfs.items():
    # Define the UMAP model for dimensionality reduction
    umap_model = umap.UMAP(n_neighbors=5, n_components=2, min_dist=0.0, metric='cosine',
                           low_memory=False, random_state=891)

    # Define the HDBSCAN model for clustering
    hdbscan_model = hdbscan.HDBSCAN(min_cluster_size=3, min_samples=3, metric='euclidean',
                                     cluster_selection_method='eom', prediction_data=True, core_dist_n_jobs=1)

    # Define the CountVectorizer model for text vectorization
    vectorizer_model = CountVectorizer(ngram_range=(1, 2), stop_words='english')

    # Define the BERTopic model
    topic_model = BERTopic(language="english", verbose=True, umap_model=umap_model,
                           hdbscan_model=hdbscan_model, vectorizer_model=vectorizer_model)

    # Apply BERTopic to the current topic's anecdotes
    topics, probs = topic_model.fit_transform(topic_df['Anecdote_preprocessed'])
```

The table below summarizes the results of the clustering analysis for each topic.

	<b>topic</b>	<b>clusters</b>	<b>total anecdotes</b>
<b>0</b>	<b>Fee</b>	<b>6</b>	<b>62</b>
<b>1</b>	<b>inventory management</b>	<b>7</b>	<b>89</b>
<b>2</b>	<b>Listing</b>	<b>5</b>	<b>55</b>
<b>3</b>	<b>Seller Support</b>	<b>2</b>	<b>15</b>
<b>4</b>	<b>ADS &amp; Deals</b>	<b>4</b>	<b>34</b>
<b>5</b>	<b>Insurance</b>	<b>2</b>	<b>18</b>
<b>6</b>	<b>inbounding</b>	<b>3</b>	<b>27</b>

Figure 4: Cluster for each topic

## 4 Summarization

After clustering the anecdotes, we now have each topic linked to its corresponding cluster. The next step is to better understand the content of these clusters. Reading every anecdote would be too time-consuming, so instead we generate a summary for each cluster.

For this, we use the Llama2 model, a large language model well suited for text generation and summarization. By processing the anecdotes within each cluster, Llama2 produces concise summaries that capture the key themes and insights, making it easier to inform decisions and improvements on the Fulfillment by Solde platform. .

```
# Load the Llama2 model from Hugging Face
llama_tokenizer = AutoTokenizer.from_pretrained("meta-llama/Llama-2-7b-chat-hf")
llama_model = AutoModelForCausalLM.from_pretrained("meta-llama/Llama-2-7b-chat-hf")

# Create a pipeline for the Llama2 model
llama_pipeline = pipeline("text-generation", model=llama_model,
tokenizer=llama_tokenizer, device=0 if torch.cuda.is_available() else -1)
```

The next step was to define a function that takes the anecdotes of a cluster and creates a prompt for the Llama2 model. The model is tasked with producing an extremely short and cohesive summary based on the provided anecdotes. The importance of prompt engineering cannot be overestimated in this context; the way are framed the input prompts significantly influences the quality and relevance of the generated summaries. .

```
def generate_summary(input_text):
    prompt = (
        "You will receive comments from e-commerce sellers which belong to a specific cluster. "
        "Your task is to generate an extremely short and cohesive summary in a single sentence. \n" f"Summary: {input_text}\n\nProvide one extremely short and cohesive summary in a single sentence for the above comments.")
    try:
        response = llama_pipeline(prompt, max_length=1024, top_p=0.9, temperature=0.6)
        return response[0]['generated_text']
```

Once the function was defined, the anecdotes of each cluster were concatenated into a single string and passed to the summarization model. For each cluster, the generated summary was stored along with its topic and cluster information:

```
summaries = []
for topic, topic_dict in enumerate(dict_list_by_topic):
    for topic_name, anecdotes in topic_dict.items():
        clusters = set([anecdote['Cluster'] for anecdote in anecdotes])
        for cluster in clusters:
            cluster_anecdotes = [anecdote['Anecdote'] for anecdote
                                in anecdotes if anecdote['Cluster'] == cluster]
            input_text = " ".join(cluster_anecdotes)
            summary = generate_summary(input_text)
            summaries.append({"topic": topic_name, "cluster": cluster,
                             "summary": summary})
```

Below are the summaries of the clusters for the Fee topic:



Summary	Cluster
The referral fee is higher than expected, causing frustration among customers and impacting the seller's ability to forecast demand, leading to a pause in ads until the issue is resolved.	5
The payout delay issue is affecting sellers' ability to forecast demand, impacting sales, and causing unexpected operational overhead, with multiple occurrences this month, leading some to consider pausing ads until it's fixed.	1
Summary: The fluctuating packaging charges across identical products are causing disruptions in sales and forecasting, leading to unexpected operational overhead and frustrated customers, prompting the seller to consider pausing ads until the issue is resolved.	2
The seller faces daily budget drains with minimal conversions, leading to unexpected operational overhead and impacting their ability to forecast demand, causing frustration among customers, and happening multiple times a month, resulting in negative sales impact.	0
Unexpected storage costs are causing operational overhead and frustrating customers without a clear reason, happening multiple times this month.	4
Split shipments increase handling charges noticeably, impacting sales and forecasting demand, occurring multiple times this month.	3

Figure 5: Fee clusters summary

In short, using the Llama2 model to generate concise summaries of the clusters gave us clear insights into seller issues and sentiments. By condensing feedback into short, actionable statements, we were able to highlight key issues and trends within each topic. This not only improves our understanding of seller experiences but also prepares the ground for the next step: creating titles that make the findings easier to communicate and act upon.

## 5 Title generation

Following the summarization of clustered anecdotes, the next step involves generating titles that encapsulate the essence of each summary. This step is crucial as effective titles serve as quick reference points, allowing stakeholders to grasp the main ideas without delving into the details of each summary.

After summarizing the clustered anecdotes, the next step is to generate titles that capture the essence of each summary. Good titles act as quick reference points, helping stakeholders understand the main ideas at a glance. For this, we use the FLAN-T5 Large model, which is well suited for text-to-text tasks like title generation. Thanks to its training on diverse datasets, it can grasp context and semantics effectively, producing concise and accurate titles from the summaries. .

```
from transformers import T5Tokenizer, T5ForConditionalGeneration
# Load the FLAN-T5 tokenizer and model
model_name = "google/flan-t5-large"
tokenizer = T5Tokenizer.from_pretrained(model_name)
model = T5ForConditionalGeneration.from_pretrained(model_name)
```

We then define a function which takes a summary as input and generates an appropriate title. The function provides a prompt specifically designed to ask for a concise title to the FLAN-T5 model. .

```

def generate_title(summary_text):
    prompt = f"Generate a short and descriptive title for the following summary:
    {summary_text}"
    inputs = tokenizer(prompt, return_tensors="pt", max_length=512, truncation=True)
    output = model.generate(**inputs, max_new_tokens=500, num_beams=4, top_p=0.9, temperature=0.6)
    title = tokenizer.decode(output[0], skip_special_tokens=True)
    return title

```

Subsequently, we iterate through each summary in the dataset, generating a title for each one. Below it is shown the titles and summaries for the Fee topic:

Title	Summary	Cluster
Sellers: Referral fee is higher than expected, causing frustration	The referral fee is higher than expected, causing frustration among customers and impacting the seller's ability to forecast demand, leading to a pause in ads until the issue is resolved.	5
PayPal Payout Delay Is Affecting Sellers	The payout delay issue is affecting sellers' ability to forecast demand, impacting sales, and causing unexpected operational overhead, with multiple occurrences this month, leading some to consider pausing ads until it's fixed.	1
Sellers 'Pausing Ads' Until Issue	Summary: The fluctuating packaging charges across identical products are causing disruptions in sales and forecasting, leading to unexpected operational overhead and frustrated customers, prompting the seller to consider pausing ads until the issue is resolved.	2
Sellers: The seller faces daily budget drains with minimal conversions, leading to unexpected operational overhead and impacting their ability to forecast demand	The seller faces daily budget drains with minimal conversions, leading to unexpected operational overhead and impacting their ability to forecast demand, causing frustration among customers, and happening multiple times a month, resulting in negative sales impact.	0
Unexpected storage costs frustrate customers	Unexpected storage costs are causing operational overhead and frustrating customers without a clear reason, happening multiple times this month.	4
Sellers: Split shipments increase handling charges	Split shipments increase handling charges noticeably, impacting sales and forecasting demand, occurring multiple times this month.	3

Figure 6: Fee clusters titles and summaries

## 6 Conclusions

In this work, an AI-based pipeline was developed to analyze seller feedback on the Fulfillment by Solde (FBS) platform. Through the combination of classification, clustering, summarization, and title generation, a system was created to transform large volumes of unstructured anecdotes into clear and actionable insights.

Seller anecdotes were first classified into relevant topics using BERT, achieving high accuracy in the categorization task. Clustering with BERTopic was then applied, allowing the identification of sub-themes and emerging issues within each topic. Subsequently, summaries were generated with Llama2 to provide concise representations of each cluster, and titles were produced with FLAN-T5 to facilitate quick communication of the findings.

Overall, the proposed pipeline demonstrated the potential of advanced NLP techniques for gaining a deeper understanding of seller sentiment. In addition, it provided a scalable and efficient method to continuously monitor feedback and inform improvements to the platform. By structuring and highlighting seller

challenges, the approach contributes to enhancing the overall seller experience on FBS.