

# Shopee Negative Reviews Analysis – Text Mining Project

## 1. Project Overview

This project analyzes **30K+ Shopee app reviews** to identify recurring issues, track topic trends across app versions, and detect potential topic drift. The goal is to provide actionable insights for app improvement, focusing on negative user experiences.

## 2. Objectives

- Discover **key complaint clusters** among negative reviews.
- Predict sentiment reliably using text-based classification.
- Track topic trends and spikes across app versions.
- Detect **topic drift** over time to understand evolving user concerns.
- Provide visualizations and representative reviews to guide product prioritization.

## 3. Methodology

### 3.1 Data Collection

- Scraped **reviews** from Google Play.
- Collected: `review_text`, `rating`, `likes`, `review_app_version`, `review_date`, etc.
- Target dataset: **30K+ reviews** across multiple app versions.

### 3.2 Preprocessing

- Light cleaning: lowercasing, punctuation removal, minimal tokenization.
- Cleaned text stored in `texts_clean`.

### 3.3 Sentiment Prediction (Supervised Learning)

- Model: **IndoBERT** (via SentenceTransformer).
- Purpose: classify sentiment regardless of review rating (to handle inconsistencies like 1-star but positive text).
- Applied **full dataset prediction** to label negative, neutral, or positive sentiment.

### 3.4 Unsupervised Topic Modeling (Global Topics)

**Target:** All negative reviews.

**Model:** KMeans clustering on **IndoBERT embeddings** (optimal k=3, determined via **elbow method & silhouette score**).

**Reproducibility:** Fixed random seeds.

- Assigned each review to a **global topic**.
- Extracted **top keywords** per topic using TF-IDF.
- Computed **topic trends per version**.
- Visualized topics with **WordClouds** and line charts for trend analysis.

### 3.5 Trend Analysis

- Tracked **topic frequency per version**.
- Identified:
  - Persistent issues
  - Version-specific spikes
- Visualization: **line charts per topic**.

### 3.6 Topic Drift Analysis

- Compared **topic stability across versions** using:
  - **Cosine similarity** of top keywords
  - **Jensen-Shannon divergence**
- Goal: detect semantic drift or merging/splitting of topics over time.
- Result: topics **stable**; no significant drift detected.

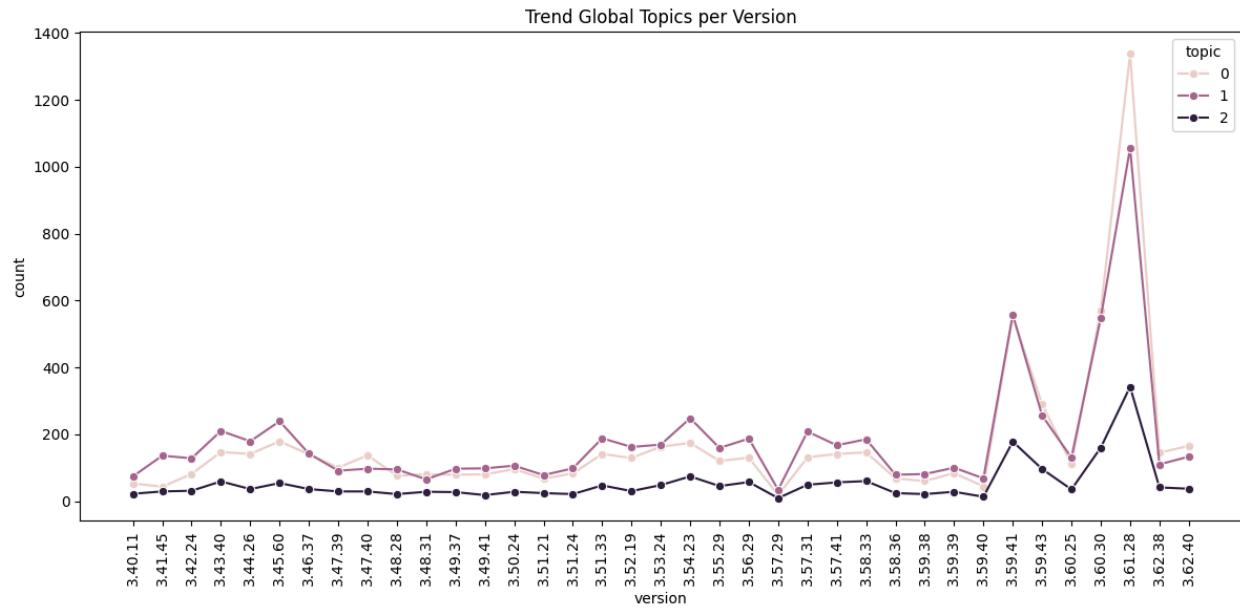
Topic	Dominant Keywords	Trend	Interpretation	Notes
0	pengiriman, lama, barang, paket, kurir	Stable, fluctuates	Shipping issues	Operational; no strong correlation with app versions
1	aplikasi, buka, video, iklan, makin	Upward trend; dominant	App/UX issues	Main pain point; persistent across versions
2	lama, lemot, iklan, lambat	Persistent, minor	Technical performance	Low-level but consistent; relevant for long-term optimization

### Representative Reviews (Sample):

- **Topic 0:** "Barang sampai terlambat, paket lama sekali dikirim SPX."
- **Topic 1:** "Aplikasi sering tidak bisa dibuka, iklan muncul terus."
- **Topic 2:** "Aplikasi lemot dan lambat, sering hang saat video diputar."

## 4.2 Trend Analysis

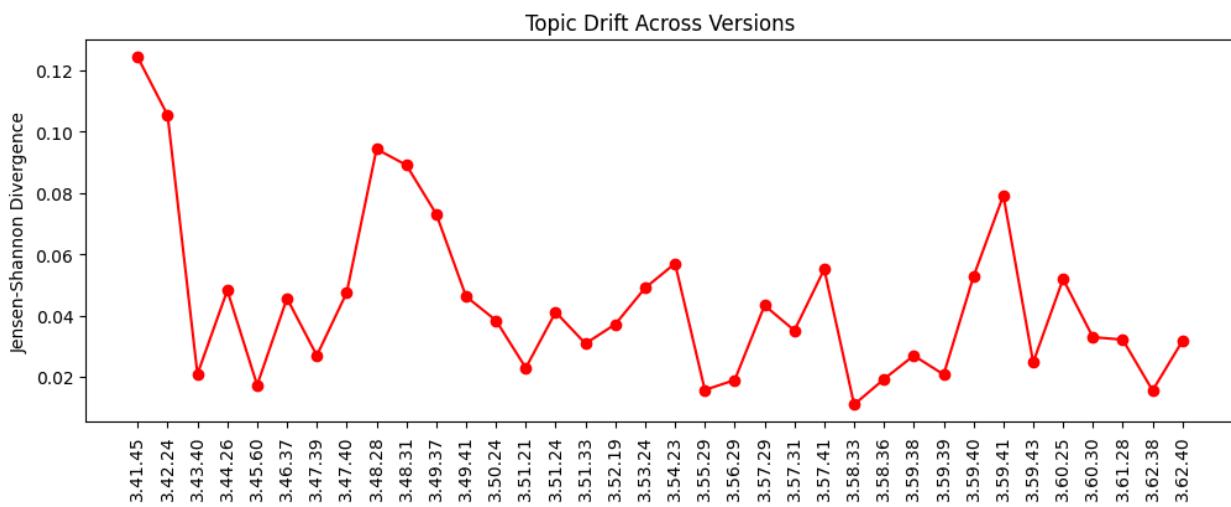
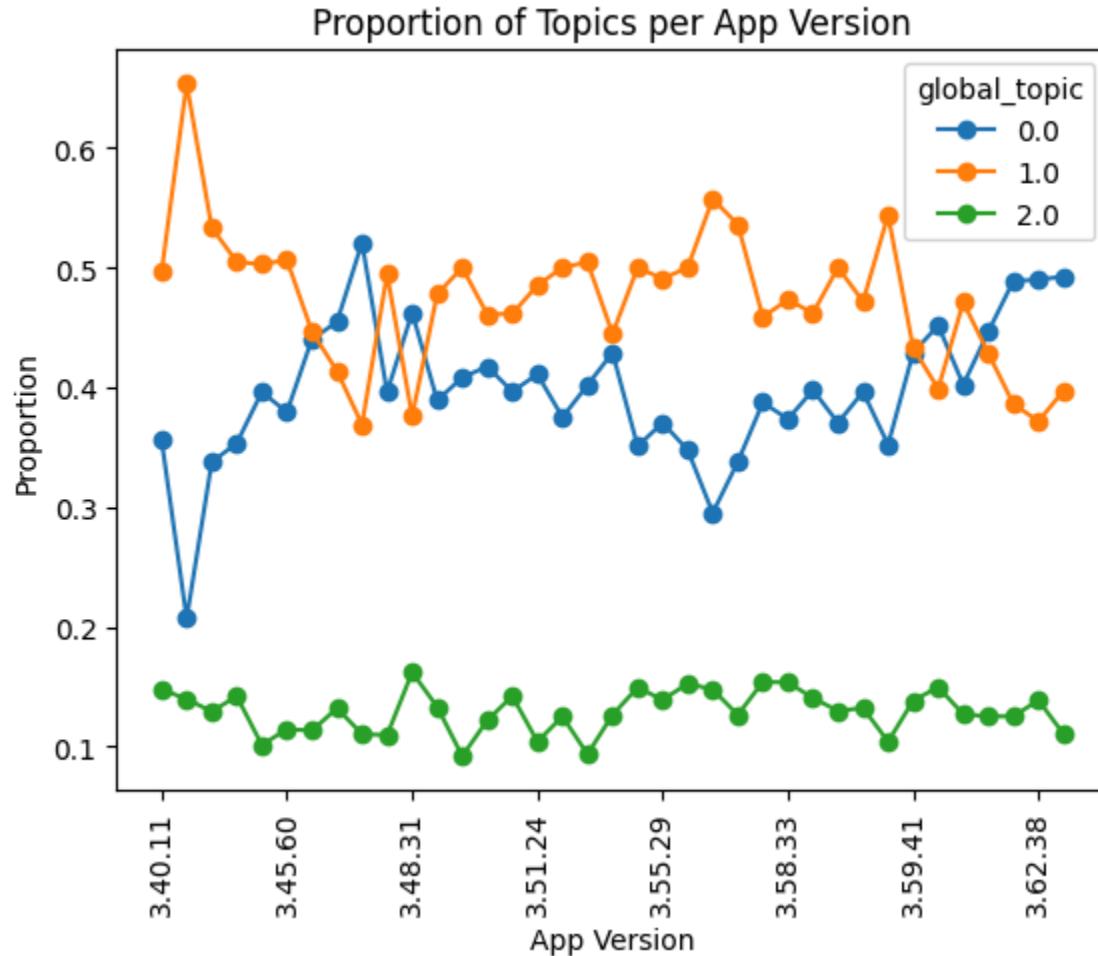
- **Topic 1 (App/UX issues)** dominates recent versions (3.54–3.62), indicating **persistent UX concerns**.
- **Topic 0 (Shipping)** fluctuates but remains significant; operational problem rather than technical update.
- **Topic 2 (Performance)** minor but consistent across versions.



## 4.3 Topic Drift

- Cosine similarity between consecutive versions > 0.8 for all topics. **Stable semantic clusters**.
- JS divergence peaks aligned with system-wide spikes in all topics. Not indicative of drift.

- Conclusion: No significant topic drift detected.



## 5. Key Insights

- **Three major complaint clusters** identified: Shipping, UX/Features, Technical Performance.
- **UX issues** are the fastest-growing, persistent pain point.
- **Shipping issues** remain operational but fluctuating.
- **Technical performance** minor yet persistent; long-term optimization required.
- Version-specific spikes highlight the impact of large app updates or sudden user surges.
- Sentiment prediction ensures reliable classification, avoiding misinterpretation from ratings alone.

## 6. Visualizations

1. **WordClouds** for top keywords per topic.
2. **Line chart** of topic counts per version.
3. **Cosine similarity heatmap** for consecutive versions.
4. **JS divergence plot** highlighting potential drift.

## 7. Conclusion

- Topic modeling successfully separated user complaints into **distinct, meaningful clusters**.
- Analysis provides **actionable insights** for prioritizing app improvements:
  - Focus on **UX and app functionality issues** first.
  - Address **shipping operations** as recurring operational concern.
  - Monitor **technical performance** over time.
- Version-based trend tracking and topic drift analysis **ensure robustness of findings**.