Large-Scale Sentiment and Topic Analysis of Amazon Product Reviews

Chenzheng Li
CMPT 732
Simon Fraser University

Wenxiang He
CMPT 732
Simon Fraser University

December 5, 2025

Project Repository Link:
`https://github.sfu.ca/cla429/CMPT732_AUV`
Data Visualization Link:
`https://large-scale-sentiment-and-topic-analysis.onrender.com/`

Abstract

This project presents a scalable big data pipeline for analyzing Amazon product reviews, deployed on the SFU Hadoop cluster using Apache Spark and YARN. We aimed to uncover micro-level user behavioral patterns and macro-level economic correlations by processing unstructured review text. We engineered a distributed inference mechanism using Pandas UDFs to parallelize a pre-trained BERT model for sentiment analysis.

We successfully deployed and validated the full pipeline on a pilot dataset. Our architectural verification highlighted significant infrastructure challenges: while our distributed design proved functional, the shared cluster's hardware constraints severely limited deep learning inference throughput compared to local baselines. Despite these limitations, our pilot analysis suggests a negative correlation between inflation rates and consumer sentiment. The project's primary contribution is a robust implementation of big data technologies, successfully overcoming critical challenges related to cluster environment isolation, dependency management (`venv-pack`), and distributed file system (HDFS) integration. Additionally, we developed a web-based user interface with interactive data visualizations.

1 Introduction

In the rapidly growing e-commerce landscape, customer reviews have become a primary driver of purchasing decisions. Platforms like Amazon generate millions of unstructured text reviews daily, creating a "Big Data" challenge where manual analysis is impossible. While traditional analytics rely on structured star ratings, these metrics can be misleading—failing to capture sarcasm, nuanced complaints, or external factors. Furthermore, consumer sentiment is increasingly hypothesized to be influenced by broader economic conditions, but connecting unstructured review data with structured macroeconomic indicators requires scalable computing power.

This project addresses these challenges by engineering a comprehensive data analytics system capable of processing large-scale review datasets. We leverage cluster computing (Hadoop/Spark) to perform advanced NLP tasks that would be computationally prohibitive on a single machine. Our objective is twofold: to validate a scalable technical architecture and to derive meaningful insights regarding the interplay between user sentiment, rating behaviors, and the global economic environment.

2 Problem Definition

The primary issue we investigate is the information gap between star ratings and sentiments hidden within unstructured text. Our research revealed that users often exhibit contradictory behavior—such as awarding a product five stars while simultaneously using harsh language to criticize its price and shipping speed. Our challenge lies in detecting these micro-level discrepancies to distinguish genuine product quality from externally induced dissatisfaction. Moreover, consumer sentiment can be heavily influenced by macroeconomic pressures. To explore this, we integrate diverse datasets—historical inflation rates and filtered, year-matched Amazon review data—to determine whether rising inflation impacts users' product evaluations.

"I've never been more disappointed with a purchase. The product arrived damaged, customer service was unhelpful, and it broke after just one week. Complete waste of money." **[Rating: 5/5 stars]**

We encountered significant engineering challenges regarding scalability and hardware computation. Processing hundreds of thousands of reviews using computationally intensive BERT models proved infeasible on standard hardware, necessitating a distributed inference approach. However, deploying such deep learning pipelines on shared, constrained Hadoop clusters presents unique challenges, including stringent security constraints and the need to coordinate efficient data flow between distributed file systems and local execution environments.

3 Methodology

To address the challenges of processing massive volumes of comment data, we built a comprehensive distributed data pipeline on SFU's Hadoop cluster. The system is built on Apache Spark (PySpark) and YARN, decomposing complex deep learning (NLP) tasks into scalable distributed jobs. Our method implements the following process pipeline (Figure 1).

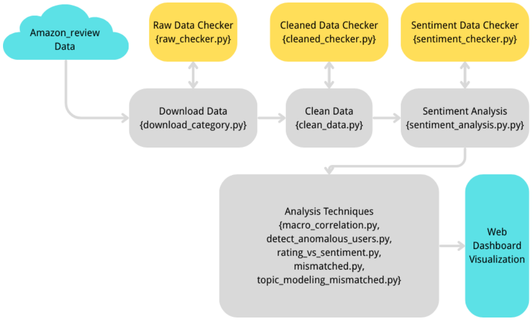
Amazon_review Data
Raw Data Checker {raw_checker.py}
Cleaned Data Checker {cleaned_checker.py}
Sentiment Data Checker {sentiment_checker.py}
Download Data {download_category.py}
Clean Data {clean_data.py}
Sentiment Analysis {sentiment_analysis.py.py}
Analysis Techniques {macro_correlation.py, detect_anomalous_users.py, rating_vs_sentiment.py, mismatched.py, topic_modeling_mismatched.py}
Web Dashboard Visualization

Figure 1: process

The pipeline consists of three stages:

3.1 Data Clean & ETL

Data Acquisition: Due to cluster security policies restricting official script execution, we developed custom scripts to directly scrape raw JSONL files from the source.

Spark Cleaning: We utilized PySpark to load raw data into distributed tables. Beyond standard operations like removing null values and deduplicating records, we focused on timestamp standardization and field extraction. The processed "clean data" was saved in Parquet format to enhance subsequent read speeds.

3.2 Distributed Emotional Reasoning System

We deployed a pre-trained BERT model (nlptown/bert-base-multilingual-uncased-sentiment) onto the cluster to score each review on a 1-5 star scale. To ensure efficient execution on a GPU-less cluster, we implemented three key optimizations:

Batch Acceleration (Pandas UDF): Standard Spark processing handles data row-by-row, which is too slow. We employed Pandas UDF technology to feed data to the model in batches, significantly boosting CPU utilization.

Environment Packing: Cluster nodes lack internet access and cannot install software. We packaged the complex environment containing PyTorch and Transformers into a single file locally, then distributed it to each compute node using Spark's broadcast mechanism.

Offline Model Strategy: To circumvent the lack of internet access on nodes, we pre-downloaded the model files and distributed them alongside the tasks. We also modified the code to force it to read the local model, ensuring task stability.

3.3 Analysis Techniques

After obtaining sentiment scores, we explore the data from two dimensions:

Micro Level: We filtered out reviews that appeared contradictory (e.g., giving 5 stars while using abusive language). For these anomalies, we employed BERTopic for clustering analysis, automatically extracting core topics (e.g., "too expensive," "discontinued") to explain inconsistent ratings.

Macro level: We averaged annual sentiment scores and cross-referenced them with World Bank inflation data. By calculating their correlation, we sought to answer: When prices skyrocket, do people's review tones deteriorate?

4 Problems Encountered & Solutions

During implementation and deployment, we encountered several significant engineering challenges.

4.1 Hugging Face Dataset Loading Failure

Problem: The `datasets` library enforces strict security policies, disabling remote loading scripts, causing our initial data ingestion to fail with a `trust_remote_code` error.

Solution: We reverse-engineered the dataset repository and implemented a custom ingestion script (`download_category.py`) that directly fetches and parses raw `.jsonl` files via their URLs.

4.2 Silent Failure in Cluster Mode

Problem: Initial Spark jobs finished with `SUCCEEDED` status but produced no output. Our code used `os.path.exists()` to check paths, but in Cluster Mode, the Driver cannot verify HDFS paths using local commands, causing silent exits.

Solution: We refactored our scripts to support a `-cluster` flag, which bypasses local checks and forces Spark to read directly from HDFS URIs.

4.3 Model Download Permission & Networking

Problem: Worker nodes lack internet access and write permissions, causing the Transformers library to crash when attempting to download models.

Solution: We implemented an "Offline Model Strategy." We downloaded the BERT model on the Gateway node, packaged it into a `.tar.gz` archive, and distributed it to workers using Spark's `-archives`.

4.4 Sentiment Analysis Misclassification

Problem: The sentiment model occasionally misclassifies negative reviews with low ratings as positive sentiment scores. This stems from the inherent accuracy limitations of the chosen pre-trained model, which may struggle with certain linguistic patterns, sarcasm, or domain-specific expressions.

Solution: Model selection plays a crucial role in reducing misclassification rates. By carefully evaluating and selecting models with higher accuracy and better domain alignment, we can significantly improve sentiment prediction reliability.

4.5 Hardware Constraints & Computational Bottleneck

Problem: Despite optimizing with Pandas UDFs, we hit a severe hardware bottleneck on the shared cluster. Deep learning inference (BERT) is highly CPU-intensive. Benchmark comparison revealed that Local Environment (AMD Ryzen 7 3800X) processed a sample batch in approximately 8 minutes, while SFU Cluster (Shared Xeon Nodes) took over 50 minutes for the same workload.

Impact: Extrapolating to the full "All_Beauty" dataset (700k+ reviews) estimated a runtime exceeding 48 hours, which was infeasible within the shared cluster's resource quotas. Consequently, we focused our analysis on a statistically significant pilot subset.

5 Results

5.1 Model Performance Validation

We validated the reliability of our sentiment inference model by comparing user star ratings (1-5) with predicted sentiment scores (1-5). Analysis reveals a strong positive correlation. As shown in Table 1, the model performs exceptionally well on positive reviews, with 5-star ratings corresponding to an average predicted sentiment of **4.62**. Lower ratings show a slight positivity bias (e.g., 1-star ratings have an average sentiment of 1.51), which likely reflects "polite negativity" in user text. The overall Mean Absolute Error (MAE) is approximately **0.407**, confirming that the BERT model is a robust proxy for user satisfaction.

Table 1: Average Predicted Sentiment per User Rating Category

User Rating	Avg. Pred. Sentiment	Avg. Abs. Diff	Count
1.0	1.51	0.51	63
2.0	2.13	0.57	63
3.0	2.82	0.59	111
4.0	3.87	0.42	189
5.0	4.62	0.38	574

5.2 Macro-Analysis: Economic Correlation

We aggregated sentiment scores annually and correlated them with US inflation data. As illustrated in Figure 2, our analysis suggests a negative correlation between inflation rates and average review sentiment. During periods of higher inflation, we observed a dip in consumer sentiment scores, supporting the hypothesis that external economic stress may negatively bias product feedback.

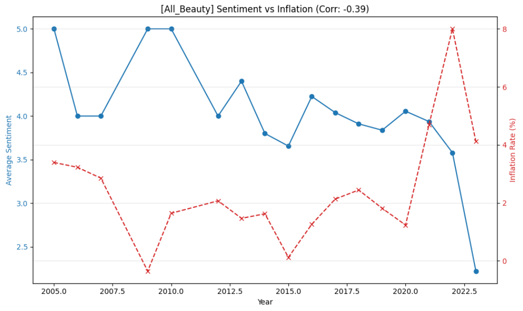

Figure 2: Temporal analysis of Average Sentiment vs. Inflation Rate. The chart reveals a negative correlation, highlighting a potential inverse relationship between economic pressure and consumer sentiment.

5.3 Micro-Analysis: The "Mismatch" Phenomenon

We identified reviews where user ratings deviated significantly from textual sentiment (e.g., $|Rating - Sentiment| \geq 3$). Our analysis reveals distinct patterns:

Price vs. Quality: Users often praise product quality but assign low ratings due to pricing. For example, one review states "only 2 problems with this product–expensive and hard to get". Despite the positive text, the user assigned a 5-star rating.

User Error / Inconsistency: We found instances of clear user error, such as a review stating "Can barley pick up any eye shadow... I threw them out" which was assigned a 5.0 rating despite highly negative text.

Domain Specificity: Some reviews like "Felt synthetic" (Rating: 1.0) received a high sentiment score (5), suggesting the general-purpose BERT model may struggle with domain-specific negative terms.

6 Project Summary

1. **Getting the data** - 1/20: Download raw JSONL from Hugging Face datasets using custom scripts, save as Parquet, and store on HDFS.

2. **ETL** - 2/20: Spark cleaning pipeline: missing value handling, type conversion, timestamp transformation, year/month extraction, and key field retention.

3. **Problem** - 2/20: Define two research questions: (1) rating vs. sentiment mismatch to identify fake reviews; (2) inflation impact on consumer sentiment through macro-economic correlation.

4. **Algorithmic work** - 3/20: Integrate BERT sentiment model via PySpark Pandas UDFs, compute MAE for consistency verification, set thresholds for mismatch analysis, and apply BERTopic for topic modeling.

5. **Bigness/parallelization** - 4/20: *Primary focus.* Spark on YARN distributed pipeline with Pandas UDF parallel inference. Resolve offline model/dependency distribution (venv-pack), cluster mode path issues, and HDFS integration. Conduct scalability evaluations.

6. **UI** - 4/20: Flask web dashboard with RESTful API architecture, interactive Chart.js visualizations, searchable/sortable tables, and real-time Spark data loading.

7. **Visualization** - 3/20: Multiple Chart.js charts (bar, line, doughnut) including dual Y-axis yearly trends, and interactive data tables with search functionality.

8. **Technologies** - 1/20: Engineering implementation of PySpark, HDFS, YARN stack with transformer-based sentiment analysis and Flask dashboard development.

AI-Assisted Development Statement

In accordance with the course policy, we acknowledge the use of AI-assisted development tools (Large Language Models) to support this project. AI was used to help brainstorm solutions for cluster-related engineering issues and provide implementation suggestions. **All final design decisions, core implementation, experiments, and result interpretation were completed and verified by the project team.**

References

[1] McAuley-Lab. “Amazon Reviews 2023.” Hugging Face Datasets. *Available:* `https://huggingface.co/datasets/McAuley-Lab/Amazon-Reviews-2023`. [Accessed: Dec. 03, 2025].

[2] World Bank. (2025). Inflation, consumer prices (annual %)” World Development Indicators. *Available:* `https://data.worldbank.org/indicator/FP.CPI.TOTL.ZG`. [Accessed: Dec. 01, 2025].