Review-Based Defect Prediction on Amazon

Team7

Team Members: Amisha Kelkar, Shrinidhi Bhide, Siddharth Kant, Guyu Zhu, Pei-Chun Yang

Project Introduction - Problem

E-commerce platforms like Amazon feature millions of products. Sometimes defective or low-quality products make it through, hurting customer satisfaction and trust. Negative experiences from defective products can result in:

- Increased returns and customer service costs
- Negative reviews that harm platform and seller credibility
- Reduced customer loyalty and conversion rates

"The average ecommerce return rate 20-30%"

* Reference: https://www.readycloud.com/info/50-statistics-on-ecommerce-returns-for-2024

Amazon experiences

14.1 million

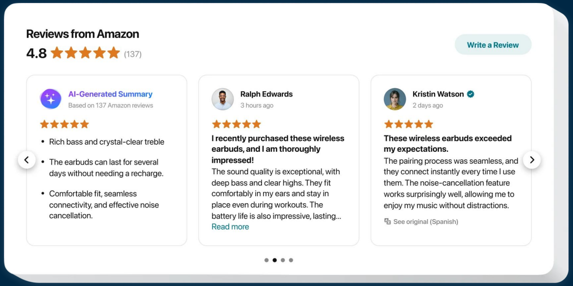
defective orders

annually

Project Introduction – Motivation and Goal

Our Motivation:
With millions of products and reviews on Amazon, it’s increasingly difficult for customers to identify which products are genuinely reliable. By analyzing review data at scale, we aim to help consumers avoid defective purchases and support platforms in identifying low-quality sellers.

Our Goal:



- Improve quality control
- Reduce defect-related costs
- Build greater customer trust

Data and Preprocessing

Dataset Source: Amazon Reviews 2023 Dataset

Focused Category:

Musical Instruments	Appliances
👤 1.8M users 📦 213.6K items ⭐ 3.0M ratings	👤 2.0M users 📦 825.9K items ⭐ 2.5M ratings

Column Selection:

Columns for User Reviews

rating','title','text','images','asin'','user_id','timestamp','helpful_vote','verified_purchase'

asin

'main_category', 'title', 'average_rating', 'rating_number', 'features', 'description', 'price', 'images', 'videos', 'store', 'categories', 'details','bought_together', 'subtitle', 'author'

Columns for Item Metadata

Data and Preprocessing

Tools used: **Google Cloud PySpark** for large-scale data handling

Preprocessing steps:

1. Convert all the review texts to lowercase.

2. Removed noise, such as extra punctuation and symbols

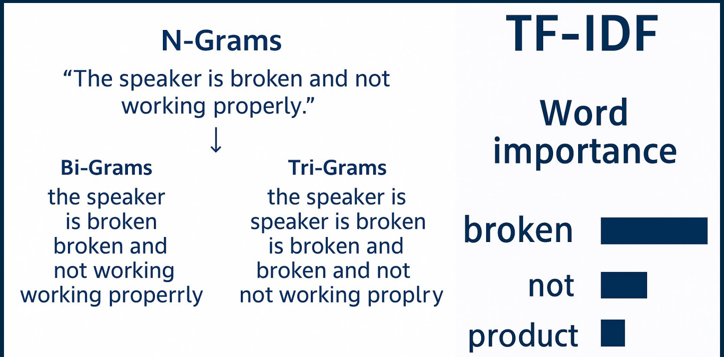
Tokenization:

First step to dealing with words to get insights .

Extracting Defect Signals: From Text to Vectors

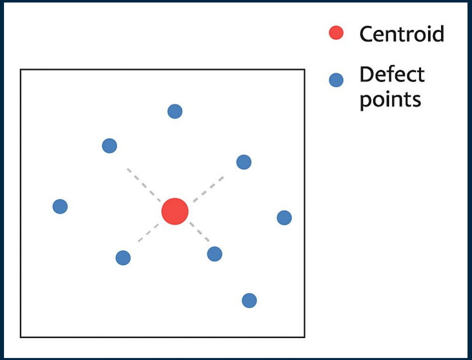
N-Grams:

- Used bi-grams and tri-grams to capture defect-related phrases (e.g., "not working", "missing parts")



TF-IDF:

- Transformed reviews into numerical vectors based on token frequency and importance



Defect Dictionary:

- Created a list of defect-related keywords
- Built a representative vector ("centroid") for defect language to compare reviews against

From Review Similarity to Store-Wide Defect Detection

Calculate cosine similarity between reviews and defect keyword vectors

↓

Apply a tuned threshold to classify reviews as defect signals

↓

Surface products associated with multiple defect-flagged reviews

↓

Flag stores if multiple products under them show high defect rates in flagged reviews

store	total_products	defective_products	defective_percent
Pyle	12	7	58.33
Ivation	26	14	53.85
GE PROFILE	17	9	52.94
Igloo	88	46	52.27
Aqua Plumb	10	5	50.0
SIMPLECUPS	15	7	46.67
NutriChef	15	7	46.67
Gevi Household	11	5	45.45

Cost Analysis and Business Impact

Category	OpenAI API (GPT-4o Mini)	Spark Classifier on Dataproc
Input Tokens per Minute	40,000,000	40,000,000
Output Tokens per Minute	1,000,000	1,000,000
API Input Cost	$6.00	N/A
API Output Cost	$0.60	N/A
Total API Cost	$6.60	N/A
Dataproc Cost per Minute	$0.004	$0.004
Compute Engine Cost	$0.019	$0.019
Total Cost per Minute	$6.623	$0.023

What about Speed?

Metric	OpenAI GPT-4o Mini API	Spark Classifier on Dataproc
Tokens per Second	~63 tokens/sec	~16,667 rows/sec (based on 1M rows/min)
Latency	Variable; can be up to 30 secs for large inputs	Low; depends on cluster configuration
Scalability	Limited by API rate limits and token processing speed	High; scales with cluster resources
Parallelism	Limited; sequential API calls	High; distributed processing across nodes

Limitations of both models

Aspect	OpenAI GPT-4o Mini API	Dataproc Spark Classifier
Language Support	Supports multiple languages and dialects.	Trained on US English; will not perform with other languages and dialects (e.g. British English).
Model Maintenance	Continuously updated by OpenAI; minimal maintenance required.	Requires periodic validation (e.g. using Open AI models) and updates.
Customization	Limited; customization is restricted to prompt engineering and fine-tuning.	High; models can be tailored to specific datasets and requirements.
Data Privacy	Data is processed by OpenAI; users must ensure compliance with data protection regulations.	Data remains within your controlled environment, enhancing privacy and compliance.
Dependency on External APIs	Dependent on OpenAI's API availability and terms of service.	None; operates within your infrastructure.

Challenges

Compute Related Issues

Configuring Dataproc clusters to meet large memory and compute demands.

Word2Vec training was too slow and resource-heavy; switched to TF-IDF with bi-grams and tri-grams for efficiency.

Thresholds for Categorization

Required setting thresholds at multiple stages (reviews, products, stores).

Optimal thresholds determined through trial and error and manual inspection.

Challenges with pre-trained Models

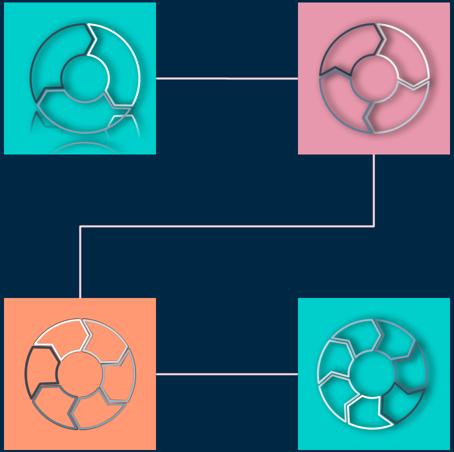
Tried using Spark NLP pre-trained models (e.g., BERT, USE) but faced Spark/Python compatibility issues and model download failures.

Despite troubleshooting (adding JARs, etc.), issues remained unresolved within the project timeline.

Future Steps

Build a neural network model which detects language, place it before defective classifier

Validate predictions from time to time using Open AI models and update accordingly



Build custom models for frequently used languages and deploy them as necessary

Build a lightweight interface for store managers or QA teams to validate/override model outputs, with feedback fed back into retraining pipelines.

THANK YOU!
Q&A