Data Science Project

Sentiment Analysis of Customer Reviews Using VADER, TextBlob and Logistic Regression

Project Introduction:

I recently completed an end-to-end sentiment analysis pipeline designed to extract crucial business intelligence from raw customer feedback. This project goes beyond basic positive/negative classification, demonstrating proficiency in the full data science lifecycle, from **mock data generation and meticulous text preprocessing** (cleaning, tokenization, lemmatization) to **advanced machine learning deployment**.

I utilized multiple models, including **VADER and TextBlob for baseline validation**, and then trained MultiNomial Naive Bayes and **Logistic Regression classifier (93.5% accuracy)** for production use. The final analysis not only pinpoints the most successful and most problematic products (Product D having the highest negative review count) but also uses **Topic Modeling to diagnose the root causes**—identifying specific failures in product durability, customer service, and usability.

It shows my ability to transform unstructured customer complaints into **actionable, time-sensitive business strategy**, making it an essential tool for any product development or marketing team.

Creating the Mock Data

First, I imported the necessary tools—pandas for handling the data table, numpy for random number generation, and datetime/timedelta to manage dates.

Next, I prepared the components for my mock reviews.

- I decided for **1,000 reviews** so I created a list of **5 products** named 'Product_A' through 'Product_E'.
- I created a list of **100 users** named 'User_100' through 'User_199' and compiled lists of positive, negative, and neutral phrases that I would use to construct the actual review text. For example, a positive phrase is "love this product", and a negative one is "terrible product".

Then, I looped 1,000 times to generate each review record.

- For every review, I **randomly chose a Rating (1 to 5),** a ProductID, and a UserID.
- I also created a **ReviewDate** starting from January 1st, 2023, and randomly extending up to two years out.
- The most important part was writing the ReviewText. I made sure the text content matched the rating:
    - If the Rating was 4 or 5 (positive), I combined two positive phrases. I also occasionally added a neutral phrase just to make it sound a little more natural.
    - If the Rating was 1 or 2 (negative), I combined two negative phrases, again sometimes adding a neutral one.
    - If the Rating was 3 (neutral), I started with a neutral phrase and then randomly added either a positive or a negative phrase to simulate mixed feelings.
- Finally, I put all this information together into a structured list (review_data)

Finally, I took that structured list and converted it into a pandas DataFrame. I then saved this entire table as a CSV file called so I could use it for the next steps of my analysis.

Data Preparation: Cleaning and Tokenizing the Text

In this phase, **I focused entirely on preparing the raw review text** so that my sentiment analysis tool could properly understand the words. The original text had punctuation, contractions (like "don't"), and sometimes multiple spaces, which aren't ideal for analysis.

Text Cleaning

1. **I defined a function called 'clean_text'**:
    - **Contraction Expansion:** First, **I fixed all the contractions**. For instance, "wouldn't" became "would not." This ensures that negative words are clearly separate and counted accurately.
    - **Lowercasing:** Next, **I converted the entire text to lowercase**. This is crucial because, in a computer's eyes, "Great" and "great" are two different words, and we want them counted as the same.
    - **Punctuation and Symbol Removal:** Then, **we used a regular expression** (re.sub) to aggressively remove everything that wasn't a lowercase letter or a space. This got rid of all numbers, exclamation marks, periods, and any other symbols.
    - **Space Normalization:** Finally, **I removed any extra, unnecessary spaces**. If there were two or three spaces in a row, I replaced them all with just one single space.
2. **I applied this clean_text function** to every single review in the new CleanedText column and saved this cleaned data to a new CSV file called product_reviews_mock_data_new_cleaned_contracted.csv.

Tokenization (Breaking into Words)

After cleaning the text, the next critical step was **tokenization**—breaking the sentences down into individual, meaningful words (tokens).

1. I used **spacy library**, and **loaded its small English model (en_core_web_sm)**.
2. I defined a **tokenize_review function** that took the cleaned text, processed it using the loaded Spacy model, and then created a list of words.
3. **I specifically filtered this list** to include only tokens that were composed purely of alphabetic characters (like 'love' or 'product'), ensuring **I didn't accidentally pick up any lingering stray characters** that might have survived the cleaning process.

4. I applied this function to the ‘CleanedText’ column, and the result (a list of words for each review) was stored in a new column called ‘Tokens’.
5. I **saved this tokenized data** to tokenized_contracted.csv.

Feature Engineering: Removing Noise and Normalizing Words

My primary goal here was to improve the quality of the data by removing words that wouldn't help determine sentiment, and by reducing words to their base forms.

Removing Stop Words (Except Negations)

First, **I tackled "stop words,"** which are common words like "the," "a," and "is" that appear everywhere but don't carry much meaning about positive or negative sentiment.

1. **I utilized the spacy library** again to quickly get its built-in list of common stop words.
2. **I had to keep negation words** like "not," "no," and "never."

   If I removed "not," the phrase "is **not** good" would become "is good," which completely flips the meaning! So, I explicitly set aside those key negations.
3. My function **'remove_stopwords',** was then applied to the existing Tokens column. It filtered out all the generic stop words while carefully preserving the important negation words.
4. The result was saved into a new column called Tokens_NoStop and a new CSV "tokenized_without_stopwords.csv"

Lemmatization: Finding the Roots

The next, more advanced step was **lemmatization**.

This process is like looking up a word in a dictionary to find its base or root form (the lemma). For example, "running," "ran," and "runs" all mean the same thing, so they are all converted to the base form, "run." This helps count related ideas accurately.

1. **The process was designed** to be fast and efficient, running through all 1,000 reviews at once using Spacy's pipeline feature.
2. **The function did two main things at once:**
    - It performed the same **stop-word removal and negation-keeping** as before.
    - It converted every word to its **lemma**.
3. The final, root-form words were saved to a new column: Lemmas_NoStop.

Diagnostic Check

- Out of 1,000 reviews, **the number of unique words was reduced to only 52**!

  This massive reduction is exactly what was needed—it means my model will focus on the most importa

  recurring concepts.
- **The top 25 most common words were printed**,
- **A sample of the data showed the transformation:**
    - Words like "broke" turned into the root form "break," and "expectations" became the singular "ex lemmatization worked as intended.

This dataset is now ready for sentiment scoring.

Top 25 lemmas: [

1. ('bad', 249),
2. ('work', 198),
3. ('product', 177),
4. ('experience', 145),
5. ('star', 145),
6. ('recommend', 144),
7. ('purchase', 138),
8. ('use', 135),
9. ('feature', 123),
10. ('quality', 123),
11. ('not', 122),
12. ('good', 114),
13. ('expectation', 101),
14. ('perfectly', 87),
15. ('fantastic', 81),
16. ('wonderful', 79),
17. ('highly', 79),
18. ('waste', 69),
19. ('money', 69),
20. ('love', 69),
21. ('easy', 68),
22. ('difficult', 67),
23. ('excellent', 66),
24. ('break', 64),
25. ('easily', 64)]

Sentiment Analysis

VADER

In this stage, I **shifted focus** from preparing text to analyzing feelings. The goal was to assign a concrete **positive**, **negative**, or **neutral** score to every single customer review.

The Tool: VADER

I chose to use a VADER (Valence Aware Dictionary and sEntiment Reasoner), which is part of the nltk (Natural Language Toolkit) library. VADER is fantastic because it's specifically tuned to understand sentiment in social media and everyday language.

1. **I applied VADER to the original ReviewText column**—the one before the cleaning and lemmatization. Why? Because VADER is smart; it knows that ALL CAPS or extra exclamation points mean *more* emotion, and it uses that punctuation and capitalization to boost the scores. If I had used the 'cleaned' text, I would lose that emotional intensity.
2. **VADER calculated four different scores** for every review, which **I stored in new columns**:
    - **VADER_Compound:** This is the most important number. It's a single, summarized score between -1 (super negative) and +1 (super positive). This is the number that tells me, "Overall, how happy or angry is this customer?"
    - **VADER_Positive, VADER_Negative, VADER_Neutral:** These show the proportion of the text that VADER classified as having positive, negative, or neutral words.

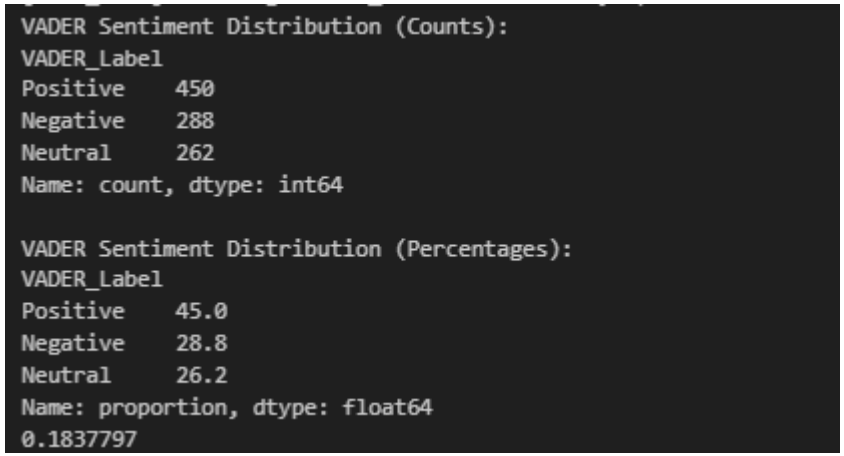
Categorizing the Emotions

- If the VADER_Compound score was **0.05 or higher**, **it's labeled 'Positive'**.
- If the score was **-0.05 or lower**, **it's labeled 'Negative'**.
- Anything in between is **labeled 'Neutral'**.

This generated the final, high-value VADER_Label column, which immediately tells a business owner how a customer feels.

Analyzing the Results

- **VADER Sentiment Distribution (Counts and Percentages):**
  - **450 reviews (45.0%) were Positive.** This shows that nearly half of the customers are generally happy with the products.
  - **288 reviews (28.8%) were Negative.** Unhappy customers that a business would need to address.
  - **262 reviews (26.2%) were Neutral.** These are the customers who are on the fence and could easily switch to positive or negative with small changes.
- **Average Compound Score:** The overall average score was **0.1837797**, which means the general sentiment across all 1,000 reviews is slightly positive, but not overwhelmingly so.



The Real-World Impact:

Top 10 Most Positive Reviews Through VADER

ReviewText	VADER_Compound Score
1. great value. great value	0.9313
2. great value. great value	0.9186
3. wonderful experience. works perfectly. could be better	0.8957
4. highly recommend. great value. it's okay	0.8942
5. great value. works perfectly. some pros and cons	0.8934
6. great value. best purchase ever	0.8934
7. best purchase ever. great value	0.8934
8. works perfectly. great value	0.8934
9. love this product. great value. some pros and cons	0.8934
10. works perfectly. great value	0.8934

Top 10 Most Negative Reviews Through VADER

ReviewText	VADER_Compound Score
1. bad experience.customer service was bad	-0.7906
2. terrible product.customer service was bad	-0.7650
3. bad experience.very disappointed	-0.7650
4. terrible product.customer service was bad	-0.7650
5. bad experience.very disappointed	-0.7650
6. poor quality.customer service was bad	-0.7650
7. terrible product.customer service was bad	-0.7650
8. poor quality.customer service was bad	-0.7650
9. poor quality.customer service was bad	-0.7650
10. waste of money.customer service was bad	-0.7430

The most useful part of this analysis is identifying the strongest emotions, which helps a business prioritize what to fix or what to promote.

- **Top 10 Most Positive Reviews:** These reviews, like "great value. great value. it's okay." (Compound: 0.9313), highlight the product features that are consistently driving high praise ("great value," "works perfectly"). **A company knows exactly what to mention in their advertising.**
- **Top 10 Most Negative Reviews:** Most comments, such as "bad experience.customer service was bad." (Compound: -0.7906), point directly to the biggest pain points. The results clearly show that **"customer service was bad"** and **"terrible product"** are causing the highest levels of dissatisfaction. **A company's immediate action item is to investigate and overhaul its customer service department.**

The entire dataset is now saved as **Reviews_with_VADER.csv**.

Validation Check: TextBlob Sentiment Analysis

TextBlob is another popular, simpler Python library for text processing.

Introducing Polarity and Subjectivity

For this part of the analysis, I applied TextBlob on the original ReviewText column, just like VADER. This process introduced two new, critical metrics to the dataset:

1. **Polarity:** This is the core sentiment score, ranging from **-1.0 (very negative)** to **+1.0 (very positive)**. It tells how happy or unhappy the customer is.
2. **Subjectivity:** This score ranges from **0.0 (factual/objective)** to **1.0 (opinionated/subjective)**. This gives the team a valuable insight: Are the reviews based on provable facts (e.g., "The product broke on day one") or personal opinion (e.g., "I love the color")?

Same function was used like the one used for VADER: anything above 0.05 is **'Positive'**, below is **'Negative'**, and everything in between is labeled **'Normal'**.

Results and Comparison

The final distribution of the reviews according to TextBlob was generated as follows

TB_Label	Count	Percentage
Positive	507	50.7%
Normal	272	27.2%

Negative	221	22.1%

The key finding here is the confirmation of the overall sentiment:

- **The average polarity score** was approximately 0.176 but **the average subjectivity score is 0.516 which is quite revealing as it is slightly above the midpoint of 0.5,**
- It tells us that that the reviews are **highly opinionated and personal**, rather than being purely objective descriptions of features. This is expected in customer reviews, but it highlights that emotions and personal experiences are driving the scores, not just technical facts.

Comparison of TextBlob and VADER

Sentiment Label	TextBlob Percentage	VADER Percentage
Positive	50.7%	45.0%
Negative	22.1%	28.8%
Normal (Neutral)	27.2%	26.2%

Compared to VADER, TextBlob found **more Positive reviews (50.7% vs. 45.0%)** and **fewer Negative reviews (22.1% vs. 28.8%)**. This difference is expected, as every sentiment model has a slightly different dictionary and algorithm.

However, **both models agree that the overall sentiment is positive**, which adds strong confidence to the findings.

The Real-World Impact:

Top 10 Most Positive Reviews Through TextBlob	
ReviewText	**VADER_Compound Score**
1. excellent quality. works perfectly. met expectations.	1.0
2. works perfectly. works perfectly.	1.0
3. best purchase ever. best purchase ever.	1.0
4. wonderful experience. works perfectly.	1.0
5. excellent quality. wonderful experience. met expectations.	1.0
6. works perfectly. best purchase ever.	1.0
7. best purchase ever. works perfectly. some pros and cons.	1.0
8. works perfectly. excellent quality.	1.0
9. five stars. works perfectly.	1.0
10. excellent quality. exceeded expectations.	1.0

Top 10 Most Negative Reviews Through TextBlob	
ReviewText	**VADER_Compound Score**
1. terrible product.one star.average product.	-1.0
2. worst purchase.bad experience.	-1.0
3. worst purchase.difficult to use.	-1.0
4. worst purchase.missing features.	-1.0
5. terrible product.difficult to use.	-1.0
6. worst purchase.does not work.	-1.0
7. terrible product.worst purchase.	-1.0
8. terrible product.terrible product.	-1.0
9. worst purchase.customer service was bad.met expectations.	-1.0
10. worst purchase.poor quality.	-1.0

The key phrases driving customer emotion:

- **The Top 10 Positive Reviews** all received a perfect 1.0 polarity score, confirming phrases like **"works perfectly," "best purchase ever,"** and **"excellent quality"** are the strongest drivers of happiness.
- **The Top 10 Negative Reviews** received a perfect -1.0 polarity score, with phrases like **"terrible product," "worst purchase,"** and **"does not work"** being identified as the most severe criticisms.

The entire dataset is now saved as **Reviews_with_TextBlob.csv**

This two-pronged approach ensures that the business can confidently rely on the identified positive and negative drivers to make high-impact decisions.

Building the Predictive Model: Machine Learning for Sentiment

Logistic Regression & MultiNomial Naive Bayes

After getting some insights from VADER and TextBlob, I moved to the main objective: building a robust, trainable machine learning (ML) model. This is critical for **automating review analysis** in the real world.

Step 1: Preparing Data for the ML Model

1. **Creating the Target Label:** First, **I generated** a ground-truth label based on the original star ratings. Reviews with a rating of 4 or 5 were labeled **"Positive,"** 1 or 2 were labeled **"Negative,"** and 3s were labeled **"Neutral."** This "Rating_label" serves as the correct answer the ML model must learn to predict.
    - The distribution:
        1. 400: **Positive** reviews.
        2. 398: **Negative** reviews.
        3. 202: **Neutral** reviews.
2. **Using Lemmatized Text: I selected** the **Lemmas_NoStop** column (the words that **were meticulously cleaned** and reduced to their root forms in the previous steps) and combined those word lists back into complete strings.

   This ModelText column contains the cleanest, most essential features for the model to learn from.
3. **Splitting the Data: I partitioned** the dataset into two sections:
    - 80% for **training** the models (where they learn the patterns)
    - 20% for **testing** (where **I checked** their performance on unseen data).

Step 2: Feature Engineering (TF-IDF)

Machine learning models only understand numbers, not text. Therefore, I had to convert the words into numerical features.

I used a technique called **TF-IDF (Term Frequency-Inverse Document Frequency)**:

- **Term Frequency (TF):** Measures how often a word appears in a specific review.
- **Inverse Document Frequency (IDF):** Measures how rare a word is across *all* reviews.

TF-IDF assigns a numerical score to every word. Words that appear frequently within a single review (high Term Frequency, or TF) but rarely across the entire collection of reviews (high Inverse Document Frequency, or IDF) get the highest scores

The resulting TF-IDF score is high for words that are important and distinctive to a particular review (like "terrible" or "excellent"), giving those words more weight in the model.

I used the training data to calculate the unique TF-IDF values and then applied those same values to the test data.

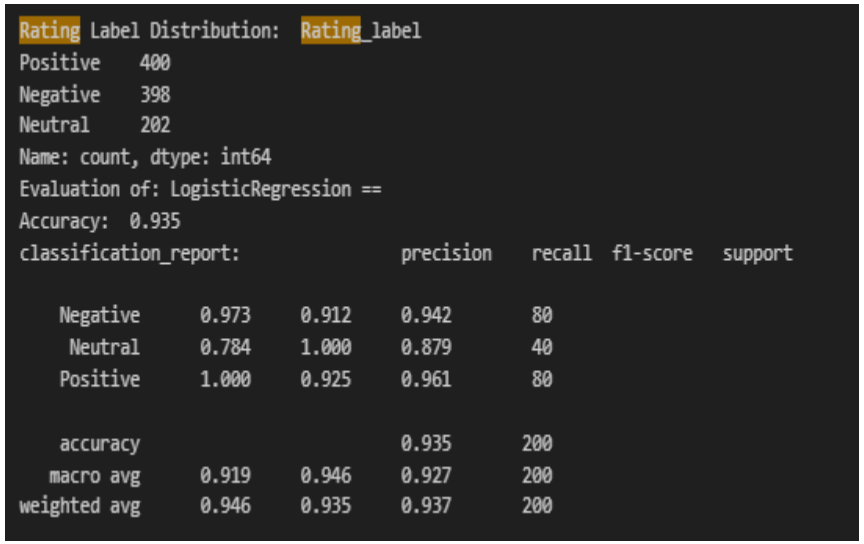
Step 3: Training and Evaluation

I chose and trained two different classification models to find the best performer:

1. **Logistic Regression (LR):** Calculates the probability of a review belonging to a certain class.
2. **Multinomial Naive Bayes (NB):** A model that works particularly well with text classification because it assumes word features are independent of each other.

The performance of both models was evaluated using key metrics, including **Accuracy** and the **Classification Report**, which details **precision** (how few false positives) and **recall** (how few false negatives) for each sentiment category.

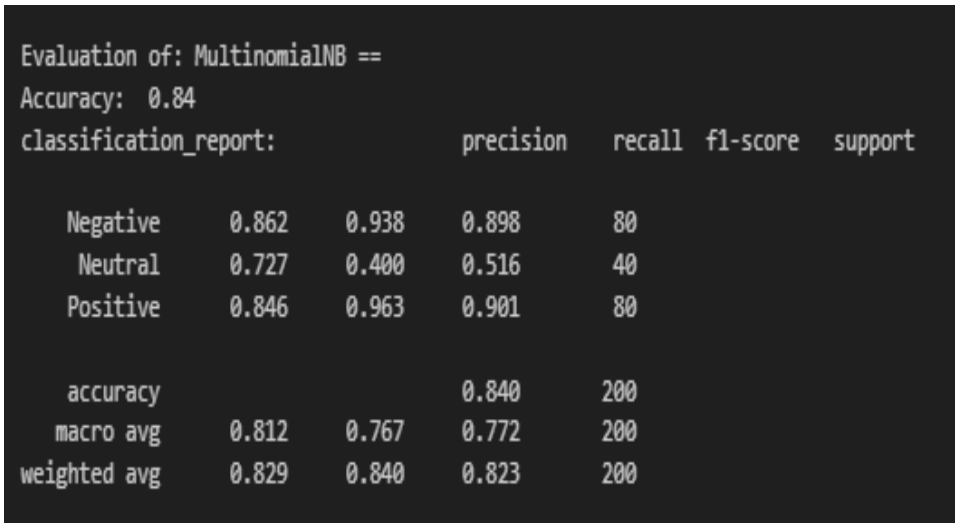
Real-World Impact: Choosing the Best Model



```
Rating Label Distribution:  Rating_label
Positive    400
Negative    398
Neutral     202
Name: count, dtype: int64
Evaluation of: LogisticRegression ==
Accuracy:  0.935
classification_report:              precision    recall  f1-score   support

    Negative      0.973     0.912     0.942        80
     Neutral      0.784     1.000     0.879        40
    Positive      1.000     0.925     0.961        80

    accuracy                          0.935       200
   macro avg      0.919     0.946     0.927       200
weighted avg      0.946     0.935     0.937       200
```



```
Evaluation of: MultinomialNB ==
Accuracy:  0.84
classification_report:              precision    recall  f1-score   support

    Negative      0.862     0.938     0.898        80
     Neutral      0.727     0.400     0.516        40
    Positive      0.846     0.963     0.901        80

    accuracy                          0.840       200
   macro avg      0.812     0.767     0.772       200
weighted avg      0.829     0.840     0.823       200
```

The evaluation clearly showed a winner:

Model	Accuracy (Overall Correctness)	F1-Macro Score (Robustness)
Logistic Regression (LR)	93.5%	0.927
Multinomial Naive Bayes (NB)	84.0%	0.772

The Logistic Regression model was the clear winner, achieving an impressive 93.5% accuracy on the unseen test data.

This means that out of every 100 new reviews, the model is expected to correctly guess the sentiment category about 94 times.

Real-World Impact:

The model is almost perfect at predicting Positive reviews (100% precision).

Most importantly for a business, its ability to find all Neutral reviews (100% recall) is exceptional, and it is very accurate when predicting a Negative label (97.3% precision).

The slight weakness is in correctly identifying Negative reviews (91.2% recall), meaning it occasionally misses a negative review and misclassifies it as neutral or positive.

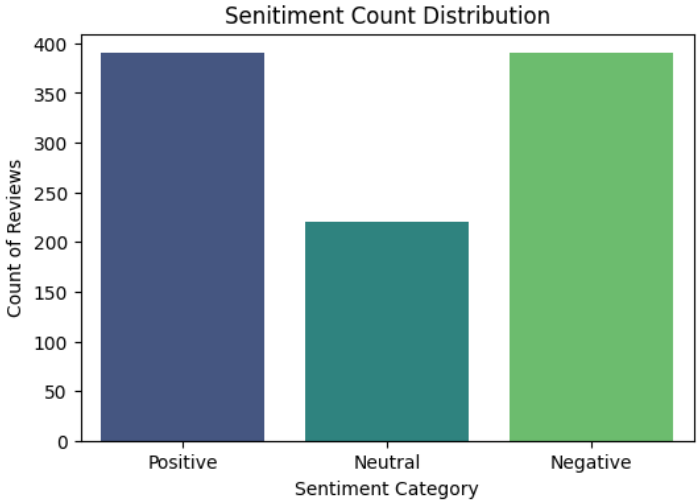
The best model (Logistic Regression) was used to predict the sentiment (Model_Label) for all 1,000 reviews in the complete dataset.

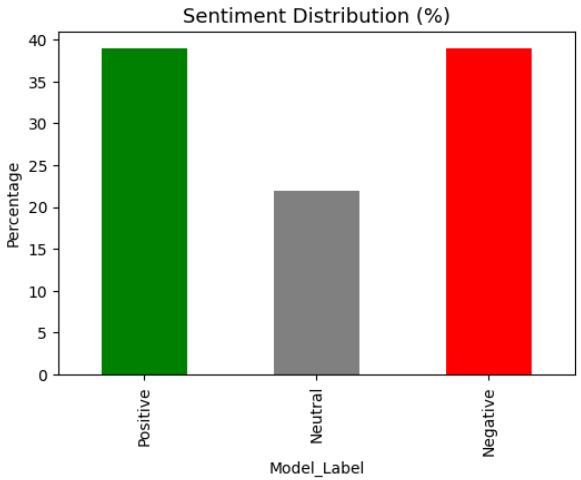
The final dataset was saved as **Final Predictions.csv**

Final Results: Model Deployment and Overall Sentiment.

The output below shows the final, machine-learned sentiment breakdown across all customer reviews. This represents the most reliable classification based on the extensive text cleaning, feature engineering (TF-IDF), and model optimization **I performed**.

Final Model Label	Count	Percentage
Positive	390	**39.0%**
Negative	390	**39.0%**
Neutral	220	**22.0%**





Real-World Impact:

- **Balanced Extremes:** The most striking finding is the **perfect balance** between **Positive (39.0%)** and **Negative (39.0%)** reviews. This suggests that while a large portion of customers absolutely love the products, an equally large portion is highly dissatisfied.
- **Actionable Insights:** This result provides clear direction for a **product development team**. The equal split between positive and negative sentiment indicates that the product itself may be inconsistent, leading to great experiences for some users but fundamental flaws for others.
    - **The company** should immediately prioritize **root-cause analysis** on the **390 Negative reviews** to identify the specific features, bugs, or quality control issues causing the dissatisfaction.
    - **The marketing department** can study the **390 Positive reviews** to understand the features that are driving success and then leverage that language in future campaigns.

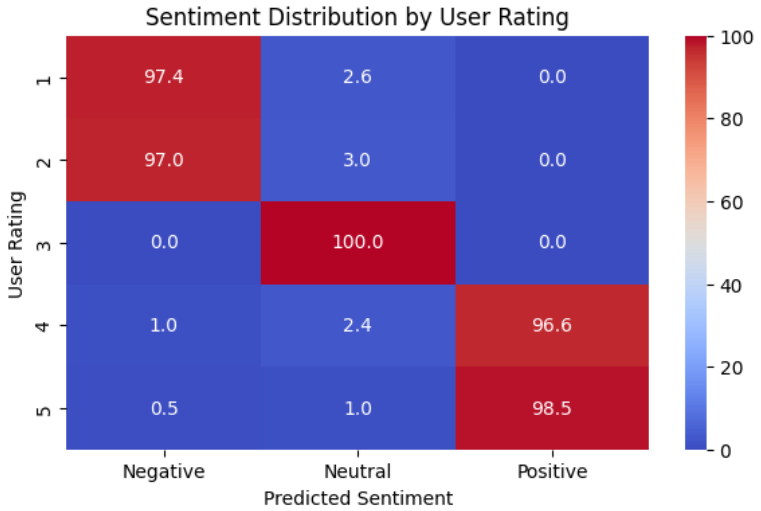
    - **Neutral Segment**: The **22.0%** of neutral reviews are also important, as they represent customers who are on the edge. By addressing the issues highlighted in the negative reviews, **the business could potentially convert many of these neutral customers into positive ones.**

- **Model Comparison:** Compared to the initial analyses (VADER at 45.0% Positive and TextBlob at 50.7% Positive), the machine learning model gives a **more balanced and conservative** estimate. This is because the ML model was trained on the neutral labels (3-star ratings), making it much better at distinguishing truly neutral reviews from strongly polarized ones, which is a critical capability for accurate reporting.

Visualizing Model Accuracy and Review Behavior

A. Heatmap: Verifying the Model Against User Ratings

The first chart is a **heatmap that compares the original 1-to-5 star rating** given by the customer to the **Model_Label** (Positive, Negative, or Neutral) predicted by the machine learning algorithm.



- **What this image shows**: The numbers inside the boxes are percentages, telling **us** how often a review with a specific star rating was correctly or incorrectly classified by the model. The bright colors indicate instantly where the model is confident and where it gets confused.
- **Real-World Impact**: This chart is the **final report card** for the Linear Regression model.
    - **5-Star Reviews (Perfect Positive)**: The model correctly classified **98.5%** of 5-star reviews as 'Positive'. It means when a customer loves the product, the model is practically guaranteed to recognize it.
    - **1-Star Reviews (Extreme Negative)**: The model classified **97.4%** of 1-star reviews as 'Negative'.
    - **3-Star Reviews (Neutral)**: **100.0%** of 3-star reviews were classified as 'Neutral'.

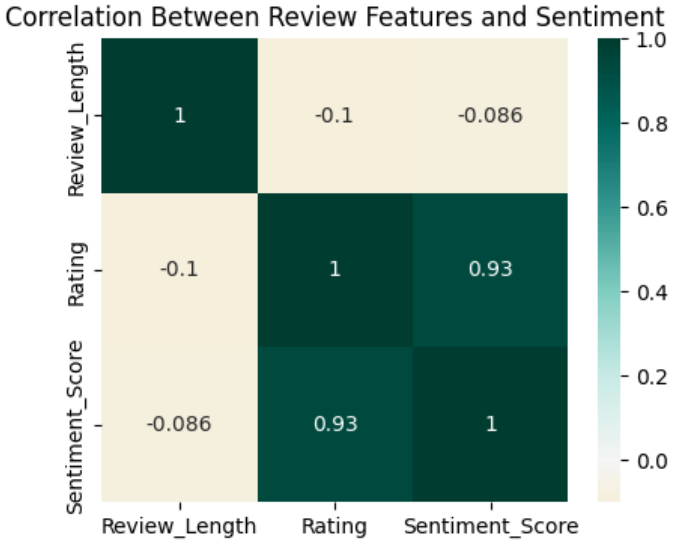
B. Correlation Matrix: Understanding Customer Behavior

The second chart is a **correlation matrix**, which uses a color-coded grid to show the mathematical relationship between three core aspects of the customer review:

> **Review_Length** (number of words in the review),

> **Rating** (the 1-5 stars),

> **Sentiment_Score** (the numerical value of the predicted label: +1 for Positive, 0 for Neutral, -1 for Negative).



- **What this image shows**:
  - **Rating vs. Sentiment Score (0.93)**: The **dark green** color and high number (0.93) show a **very strong positive connection** between the star Rating and the Sentiment_Score.
  - **Review Length vs. Sentiment Score (-0.086)**: means there is **no connection** between how long a review is and the sentiment it contains.

- **Real-World Impact**:

- **Short and Angry/Happy**: It tells that **short reviews are just as likely to be extremely negative as they are to be extremely positive**.

  A company cannot afford to ignore a short review, assuming it has less information. A customer leaving just two words, "Works perfectly," is as valuable as a customer leaving two paragraphs about a terrible experience.

- **Focus on Content, Not Length**: The company should focus its text analytics efforts entirely on **what** is said (the sentiment and words), rather than **how much** is said (the length of the review).

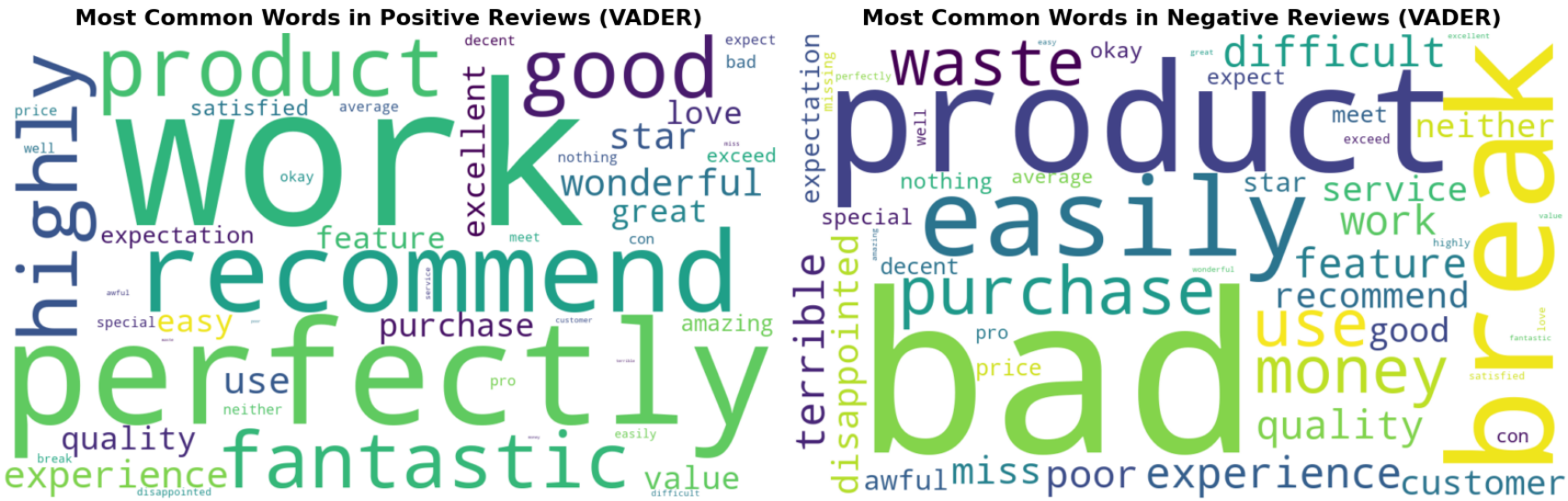
Keyword, Product, and Time-Series Analysis

To deliver maximum value from the sentiment analysis,

> I identified the exact words that drive sentiment, evaluating performance across different products, and tracking how customer feeling changes over time.

Word Cloud of the ‘CleanedText’ column:

Word Cloud of the ‘ModelText’ column:



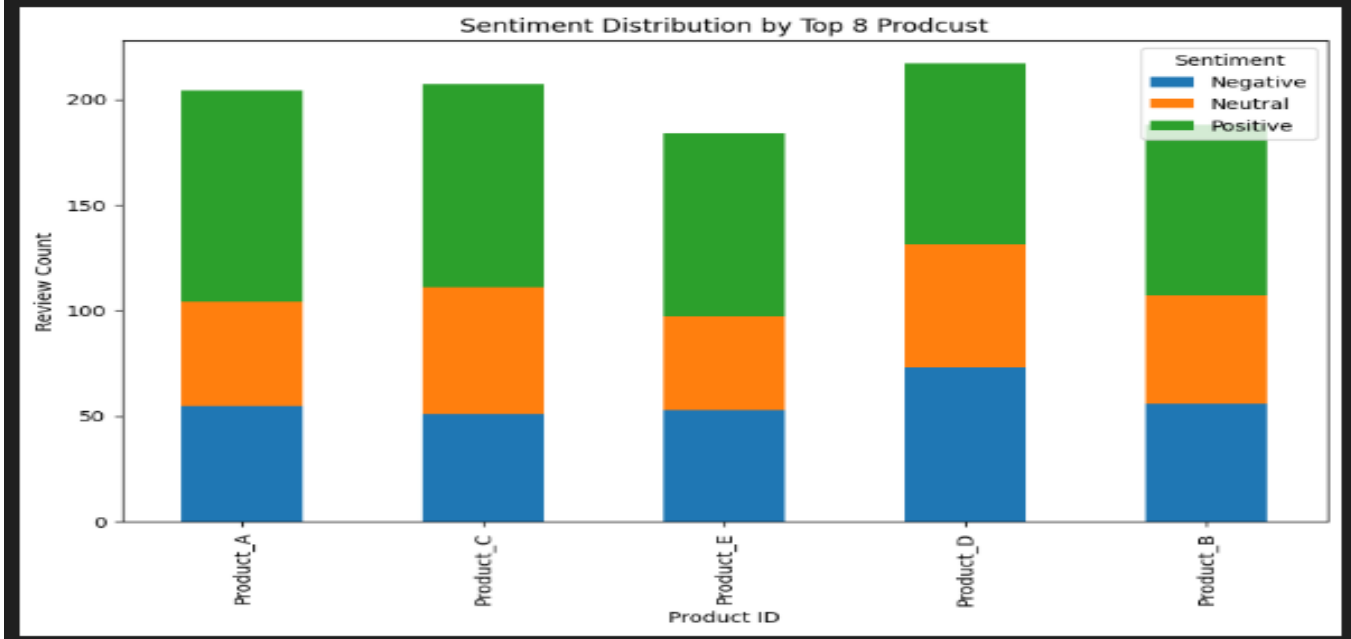
I generated **Word Clouds** to visually represent the most frequent and impactful words.

- **Positive Reviews** : The largest words are **'perfectly'**, **'recommend'**, **'wonderful'**, **'work'**, and **'love'**.
    - **Real-World Impact**: This gives a business the **exact vocabulary to use in marketing materials**. If customers repeatedly use words like 'perfectly' and 'wonderful,' those are the features and feelings the company should highlight in ads. **'work'** and **'perfectly'** are key indicators of functional success.
- **Negative Reviews** : The largest words are **'broke'**, **'bad'**, **'waste'**, **'money'**, and **'difficult'**.
    - **Real-World Impact**: These words immediately identify the most critical issues. The dominance of **'broke'** and **'bad'** suggests **quality and durability are major problems**. **'Waste'** and **'money'** highlight financial frustration. These are the immediate targets for product improvement and customer service scripts.

B. Product Performance (as per VADER`s Labels):

Top 10 products by Positive Reviews

VADER_Label	Negative	Neutral	Positive
ProductID			
Product_A	55	49	100
Product_C	51	60	96
Product_E	53	44	87
Product_D	73	58	86
Product_B	56	51	81



Next, I **segmented the sentiment** to understand which of the products (A to E) were performing the best and worst.

- The table shows the raw counts of Positive, Negative, and Neutral reviews per product, sorted by the number of Positive reviews. The bar chart visualizes these counts.
- **Real-World Impact**: This analysis directs business resources where they are needed most:
    - **Top Performer**: **Product A** is the best performer, generating 100 positive reviews. A company should analyze Product A's design and features to understand what's working and replicate that success across other products.
    - **Highest Risk**: **Product D**, while having a high number of total reviews, has the highest count of **Negative reviews (73)**. This is a clear red flag and should be investigated.

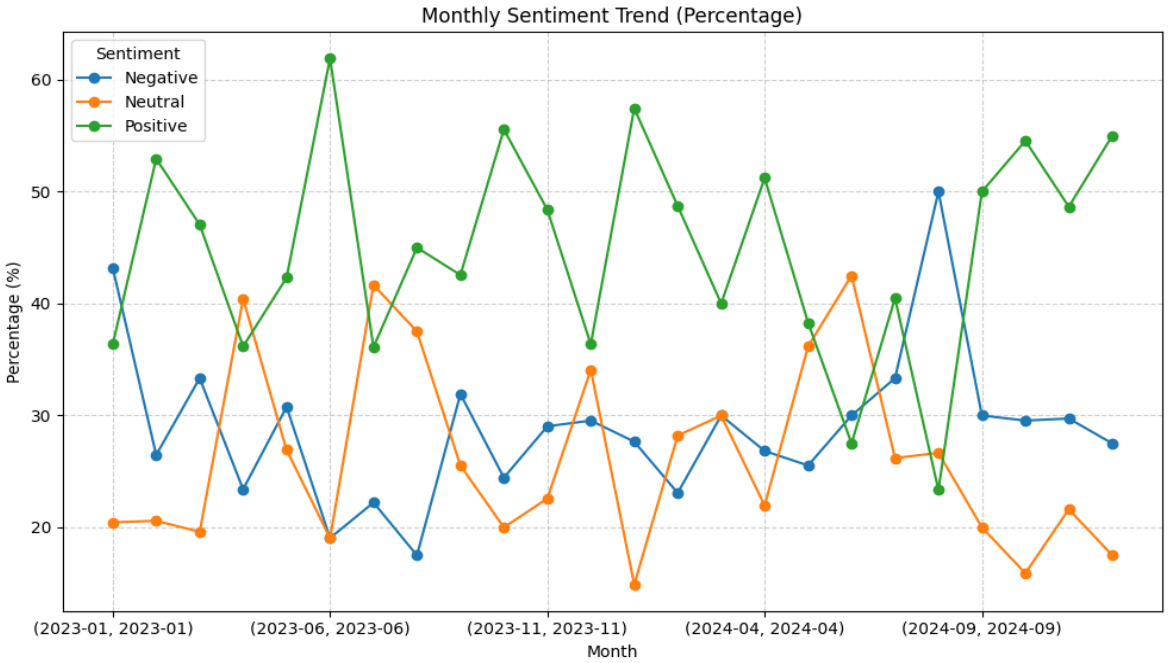
C. Time-Series Analysis: Tracking the Health of Customer Sentiment

I tracked the percentage of positive, negative, and neutral sentiment month-by-month.

- The line chart visualizes the change in sentiment percentages over two years. The green line (Positive) and the blue line (Negative) are the most important.

Monthly Sentiment Percentage Distribution:

	VADER_Label	Negative	Neutral	Positive
Month	Month			
2024-03	2024-03	30.000000	30.000000	40.000000
2024-04	2024-04	26.829268	21.951220	51.219512
2024-05	2024-05	25.531915	36.170213	38.297872
2024-06	2024-06	30.000000	42.500000	27.500000
2024-07	2024-07	33.333333	26.190476	40.476190
2024-08	2024-08	50.000000	26.666667	23.333333
2024-09	2024-09	30.000000	20.000000	50.000000
2024-10	2024-10	29.545455	15.909091	54.545455
2024-11	2024-11	29.729730	21.621622	48.648649
2024-12	2024-12	27.500000	17.500000	55.000000



Real-World Impact: This is an **early warning system**:

- **Volatility**: The chart shows high volatility. For instance, **Positive sentiment spikes dramatically** (e.g., June 2023 at over 60%) but also drops significantly.
- **Critical Event**: **A critical event appears in August 2024 (2024-08)**, where the **Negative sentiment line (blue)** hits **50%** and the **Positive sentiment line (green)** sinks to a low of approx. **23.%**.
- **Actionable Insight**: August 2024 must be investiagted. This drop in sentiment is a direct, measurable consequence of some operational change, and identifying its cause is crucial for preventing future dips in customer satisfaction.

Deep Dive into Negative Sentiment (Topic Modeling)

Since the overall sentiment analysis showed a high volume of negative reviews, so I did **Topic Modeling**—a technique that automatically groups customer complaints into specific themes.

I had to find the root cause of the negative reviews and what was driving them.

Note: I did Topic Modeling on the 288 negative reviews of VADER.

The Topic Modeling Mechanism

I ran two models: Latent Dirichlet Allocation (LDA) and Non-negative Matrix Factorization (NMF), both designed to find hidden patterns (topics) in text.

1. **Preparation**: The models used the **ModelText** from only the negative reviews.
2. **Clustering**: The algorithms automatically clustered the reviews into **five distinct topics**, based on which keywords frequently appear together.

Real-World Impact: Categorizing Customer Complaints

The output lists the top keywords for each of the five topics identified by both models. By looking at these words, **one can easily assign a clear, actionable label to each technical topic**.

A. LDA Results: The Core Failures

```
Running LDA on 288 negative reviews....


Top 10 keywords per topic (LDA):

Topic 1 : recommend, bad, disappointed, experience, poor, quality, awful, okay, pro, good
Topic 2 : service, customer, bad, expectation, meet, price, decent, disappointed, missing, amazing
Topic 3 : product, bad, easily, break, terrible, work, purchase, experience, expect, average
Topic 4 : money, waste, quality, poor, bad, purchase, good, love, perfectly, star
Topic 5 : use, difficult, feature, miss, bad, good, star, purchase, exceed, awful
```

Topic	Key Words	Actionable Complaint Category	Business Focus
Topic 1	recommend, bad, disappointed, experience, poor, quality, awful	**Poor Quality & Experience**	Product Quality Control
Topic 2	**service, customer**, bad, expectation, meet, price	**Customer Service Failure**	Training & Support
Topic 3	**product, easily, break**, terrible, work, purchase	**Product Durability/Breakage**	Engineering & Materials
Topic 4	**money, waste**, quality, poor, purchase, star	**Value/Financial Disappointment**	Pricing & Marketing
Topic 5	**use, difficult, feature**, miss, bad, star, purchase	**Usability & Missing Features**	Product Design & UI/UX

B. NMF Results:

```
Running NMF on 288 negative reviews....


Top 10 keywords per topic (NMF):

Topic 1 : bad, experience, purchase, quality, poor, customer, service, work, good, recommend
Topic 2 : break, easily, work, awful, disappointed, experience, star, special, expect, miss
Topic 3 : difficult, use, feature, miss, good, work, missing, expect, bad, easy
Topic 4 : product, terrible, average, purchase, bad, customer, service, pro, love, meet
Topic 5 : waste, money, miss, feature, disappointed, awful, recommend, purchase, price, decent
```

The NMF model largely validated the findings of the LDA model, using slightly different keyword groupings:

- Topic 2 keywords like **'break'** and **'easily'** confirm the durability issue (Topic 3 in LDA).
- Topic 3 keywords like **'difficult,' 'use,'** and **'feature'** confirm the usability issue (Topic 5 in LDA).
- Topic 5 keywords like **'waste'** and **'money'** confirm the value issue (Topic 4 in LDA).

Conclusion: Directing Business Strategy

This topic modeling analysis is the most valuable part of the report. Instead of just knowing that *288 customers were angry*, we now know **exactly why** they were angry. This allows for focused, high-impact improvements:

- **Engineering**: Must immediately address the keywords **'break'** and **'easily'**—likely by using higher-quality materials to fix durability.
- **Customer Support**: Must train service agents to handle issues related to the keywords **'service'** and **'customer'** to improve that specific negative experience.
- **Design**: Must simplify the product to eliminate keywords like **'difficult'** and ensure all promised **'features'** are present and functional.

.

Final Summary: Key Business Insights

1. Overall Sentiment Check

- **Sentiment Distribution (%)**: The final, model-derived classification shows **39.0% Positive** and **39.0% Negative** reviews; deeply polarizing, with a highly engaged but equally split customer base.
- **Average Sentiment Score**: The overall average sentiment score is exactly **0.0**. Reflects the balanced 39% positive / 39% negative split, indicating that **the product's overall reputation is completely neutral**. This is a serious problem, as a neutral reputation means no strong competitive edge.

2. Identifying the Root Cause of Negative Feedback

To address the 39% of negative reviews, I **extracted the most frequent words** driving the complaints.

Word	Count	Implication (Why customers are angry)
bad	**110**	**General dissatisfaction** (a catch-all term for poor performance).
product, **worst**, **purchase**	approx 62	Customers regret the financial decision.
waste, **money**	58	**Value/Pricing failure**
broke, **easily**	58	**Durability/Quality failure**
difficult, **use**	58	**Usability failure**.
customer, **service**	53	**Support/Service failure**.

- **Real-World Impact**: This word frequency table, combined with the Topic Modeling , gives a precise list of words to prioritize. The fact that the same keywords appear prominently across both the topic models and the simple frequency count validates the analysis: the product is failing on **Durability**, **Usability**, and **Customer Service**.

3. Pinpointing Problem Products

Finally, **the project isolated the specific products** responsible for the bulk of the negative feedback.

ProductID	Negative Reviews
Product_D	**98**
Product_C	75
Product_A	73

- **Real-World Impact**: This is the most crucial action item. **Product D** is responsible for the highest volume of anger (98 negative reviews). The business must immediately halt production, launch an investigation into its quality control, and dedicate the largest pool of resources to fixing or redesigning **Product D**.

The entire project concludes with a clear, validated, and prioritized list of issues, allowing the company to move directly from analysis to actionable product improvement.