Sentiment Classification of Amazon Product Reviews using BERT

Quinn Apostolico
qaapostolico@vt.edu

1 Motivation and Problem Statement

In an age increasingly dependent on electric commerce websites that gradually eclipse the necessity of brick and mortar retailers, companies such as Amazon are able to dominate commerce through their business model's convenience and accessibility, enabling consumers to purchase goods using their home computers and mobile phones in nearly every corner of the globe. Accompanying this accessibility to purchase is the accessibility for customers to leave feedback and reviews for these goods from their home devices. Previously, the only manner in which a customer could leave feedback for a good or service was either by word of mouth or by publishing it in some sort of editorial or television review. With the Internet, anybody with an Internet connection has the ability to leave product reviews, and as such the sheer volume of product reviews generated is unfathomably immense, exceeding the echelon of what even a large team of human beings is able to read, digest, and integrate into improvements to their products or business model. As such, in order for companies such as Amazon to capitalize on this immense but essential set of user feedback, a computational approach is crucial for processing these reviews to provide manufacturers and business leaders with key insights to improving their products and services, as well as inform product recommendation and content moderation algorithms. To accomplish this, Amazon and other electronic commerce websites seek approaches to automatically and algorithmically classify a given customer review according to the sentiment it expresses, crucially as to whether this user was satisfied or dissatisfied with the product. However, human-written text constitutes data that is incredibly noisy, and therefore challenging to automatically classify the sentiment of with a standard decision-tree algorithm. Human-written text contains informal language, slang, differing dialects, typing and grammatical errors, additional characters such as emoticons or emojis, context-dependent connotations and meanings, and other human elements that an explicit decision-tree would likely find difficult to classify. As such, an approach that is robust and adjustable to these nuances is crucial to extracting essential customer opinions.

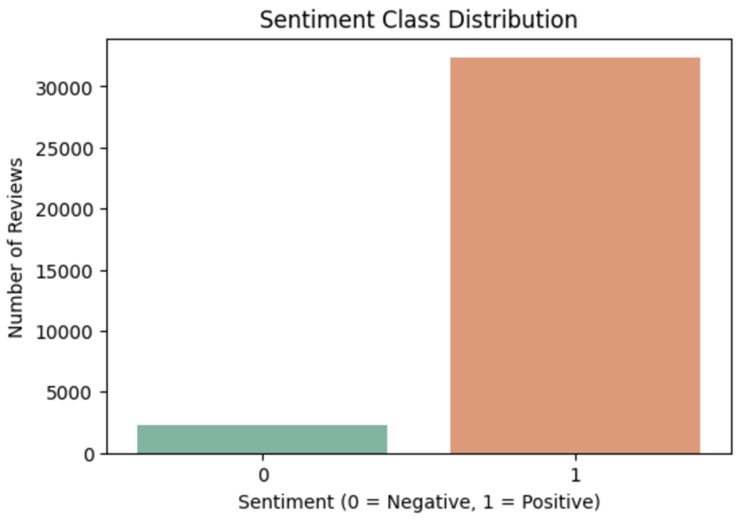
To meet these needs, this project aims to create a sentiment classification model that accurately predicts whether an Amazon product review is positive or negative based solely on its text. Using a combination of traditional machine learning techniques and deep learning models like Bidirectional encoder representations from transformers (BERT), along with techniques for preprocessing, class balancing, and feature enhancement, this project seeks to improve classification accuracy while ensuring model robustness and interpretability.

2 GitHub Repository

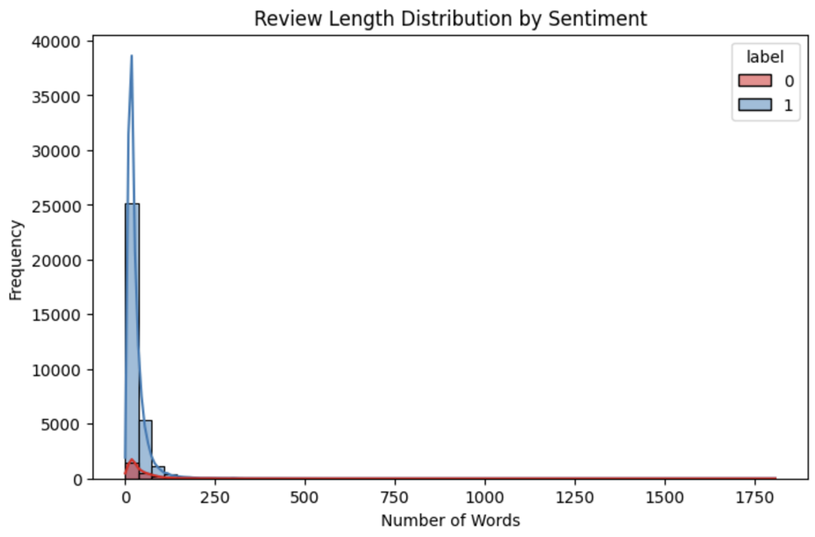
The GitHub repository of this project, containing its relevant code and data, can be found here: https://github.com/qaapostolico/STAT4744Final

3 Description of Data

The data set used in this project is sourced from Kaggle on the following link: https://www.kaggle.com/datasets/datafiniti/consumer-reviews-of-amazon-products. It features 34,660 consumer reviews for various Amazon products, including the Kindle, Fire TV Stick, and more. Each row corresponds with a particular review, containing the review's ID, product name, Amazon Standard Identification Number, brand, product categories, keys, manufacturer, review date, review date added, review date seen, whether the reviewer purchased the product, whether the reviewer recommended the product, rating, source URL, the text of the review itself, review title, reviewer's city, reviewer's province, the reviewer's username. The most pertinent variables for sentiment analysis are rating, text, title, and whether the user recommends the product. To frame sentiment as a binary classification problem, reviews rated with 4 or 5 stars are distinguished as positive reviews, whereas those rated with 1 or 2 stars are distinguished as negative. Those with 3 stars are labeled as neutral, and were excluded from training and evaluation in order to prevent ambiguity.
We can analyze this data by performing exploratory analysis. We first create a bar plot illustrating the frequency of reviews classified as positive and as negative, with 0 representing a negative review and 1 representing a positive review. We plot the two classifications of sentiment (0 as negative and 1 as positive) along the x-axis and the number of reviews along the y-axis:



From this plot, we can see that the vast majority of the reviews are classified as positive than those that are classified as negative. This suggests that most people who feel inclined to leave a product review are wanting to do so to express a positive experience with the product, with a smaller amount of people doing so to express a negative experience. Furthermore, it is also a point of interest to consider the length of a user's review, and how review length may relate to the positivity or negativity of the user's review. To do so, we create an additional bar plot of the frequencies of various review lengths, both with reviews labeled as negative (0) and positive (1). We plot the number of words along the x-axis and the frequency along the y-axis:



From this plot, we can see that since the plots of both the negative and positive reviews are heavily skewed to the right, this suggests that the vast majority of reviews are on the shorter end of this range. However, we note that there is a horizontal red line hovering just above the x-axis. This suggests that all of nearly all of reviews with a length longer than 250 words are classified as negative reviews. This suggests that users expressing negative reviews are more likely to write more verbose and long reviews than users expressing positive reviews.

4 Hyperparameter and Architecture Choices

The deep learning approach implemented in this project uses a BERT-based transformer model to perform sentiment analysis on Amazon product reviews. BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language model developed by Google that captures contextual relationships between words in a sentence by processing them bidirectionally. In this project, we fine-tune the bert-base-uncased model using the HuggingFace Transformers library. To do this, we feed the cleaned review text into BERT's tokenizer to generate input IDs and attention masks, which are then used to train a classification head on top of BERT's embeddings. Through fine-tuning, we enable the model to adapt the general language understanding learned from large-scale pretraining to the specific task of sentiment analysis and classification. This approach is particularly well-suited for handling the nuanced language patterns, domain-specific terminology, and context-sensitive sentiment expressions that are very frequent in users' product reviews, outperforming traditional models in many Natural Language Processing (NLP) benchmarks. Key hyperparameters used include setting the maximum review length in the tokenizer to 256, which limits how many tokens from the input review are used. Longer reviews are truncated, while shorter ones are padded. The learning rate was set to 2e-5, which is a commonly used value for fine-tuning transformer models in order to ensure stable convergence. The batch size was chosen to be 16 for both training and evaluation in order to balance computational efficiency with gradient estimation stability, particularly when working with the GPU constraints entailed by performing this project on Google Colab on a home computer. The model was trained for 3 epochs, which is generally sufficient for fine-tuning BERT on classification tasks without leading to overfitting. A weight decay of 0.01 was applied to help regularize the model and prevent overfitting. Our training procedure also included strategies to save and evaluate the model at the end of each epoch, with eval_loss being used as the criterion for selecting the best-performing model. These hyperparameters were passed through HuggingFace's TrainingArguments and executed through the Trainer API, which facilitates streamlined training, evaluation, and model checkpointing.

5 Results

The results of the sentiment classification model's performance on the validation set are included below. First, for each class (0 for negative reviews, 1 for positive reviews), we examine the scores for precision, recall, and F-1 that the classification model was able to achieve. The precision score refers to the proportion of predicted positives that are actually correct. In other words:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

The recall score refers to the proportion of actual positives that were correctly predicted. In other-words:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

The F-1 Score refers the harmonic mean of precision and recall, balancing both of these metrics. In other words:

$$\text{F-1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

These scores for the negative and positive classes are included in the table below:

Class	**Precision**	**Recall**	**F-1 Score**
0 (negative)	0.87	0.81	0.84
1 (positive)	0.82	0.88	0.85

Table 1: Precision, recall, and F-1 score for each class

From this table, through the precision scores we can see that roughly 87% of the reviews predicted to be negative were actually negative, and that roughly 82% of the reviews predicted to be positive were actually positive. Through the recall scores we can see that roughly 81% of the actually negative reviews were correctly classified as negative, while roughly 88% of the actually positive reviews were correctly classified as positive. To balance both of these metrics, through the F-1 score we find that the harmonic mean of precision and recall for actually negative reviews is roughly 0.84, while the harmonic mean for actually positive reviews is roughly 0.85.

The confusion matrix provides us with a breakdown of the model's classification performance on the validation set. It shows us the number of true positives, false positives, true negatives, and false negatives. This table is included below:

Table 2: Confusion Matrix

	Predicted	
Actual	0	1
0	374	89
1	55	407

From this confusion matrix, we can see that of the validation set, 374 reviews were predicted to be negative and were actually negative, 89 were predicted to be positive and were actually negative, 55 were predicted to be negative and were actually positive, and 407 were predicted to be positive and were actually positive. Therefore, the proportion of reviews predicted to be negative that were actually negative is $\frac{374}{374+55} \approx 0.87$, and the proportion of reviews predicted to be positive that were actually positive is $\frac{407}{407+89} \approx 0.82$, with both of these values matching the aforementioned precision scores of the negative and positive reviews respectively. The proportion of actually negative reviews that were correctly classified as such is $\frac{374}{374+89} \approx 0.81$, and the proportion of actually positive reviews that were correctly classified as such is $\frac{407}{55+407} \approx 0.88$, with both of these values matching the aforementioned recall scores of the negative and positive reviews respectively. Therefore, the overall accuracy can be calculated as such:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

Substituting our values:

$$\text{Accuracy} = \frac{374 + 407}{374 + 89 + 55 + 407} \approx 0.844$$

Therefore, roughly 84.4% of reviews, both positive and negative, were accurately classified.

Through calculating a macro average F-1 Score, we can treat both classes equally regardless of how many samples that each class has. We can calculate this as such:

$$\text{Macro Average} = \frac{F_{1,\text{Class 0}} + F_{1,\text{Class 1}}}{2} = \frac{0.84 + 0.85}{2} = \frac{1.69}{2} = 0.845$$

Because this macro average is so close to each individual class's F-1 score, this indicates that the model's predictive quality is balanced across the classes rather than being driven by one dominant class, as both classes contribute nearly equally to the macro average, so the average ends up almost exactly each class's F-1 score.

To account for class imbalance, we can also give each class's F-1 score a weight proportional to its frequency in the validation set. We can calculate this as such:

$$\text{Weighted Average} = \frac{n_0 F_{1,0} + n_1 F_{1,1}}{n_0 + n_1} = \frac{(374 + 89)(0.84) + (55 + 407)(0.85)}{374 + 89 + 55 + 407} \approx 0.845$$

This calculation gives larger classes more influence in the overall score. If one class dominates numerically, its performance will pull the weighted average more in its direction. Here we can see that the class imbalance is minor in the validation set, so the weighted F-1 average ends up nearly identical to the macro average.

The ROC AUC, or the Area Under the Receiver Operating Characteristic curve, represents the probability that, given a randomly chosen positive and negative example, the model will rank the positive higher than the negative, measuring how good the model distinguishes between classes. In this case, the computed ROC AUC Score was roughly 0.844.

6 Analysis of Results

From our table that displays the precision, recall, and F-1 scores for the negative and positive classes, we can see that Class 0 is slightly more precise through the precision scores. This indicates that there is a greater proportion of reviews predicted to be negative that were actually negative than the proportion of reviews predicted to be positive that were actually positive. Through the recall scores, we can see that Class 1 is slightly better recall than Class 0. This indicates that there is a greater proportion of actually positive reviews that were correctly classified than the proportion of actually negative reviews that were correctly classified. The F-1 Scores (0.84 for negative and 0.85 for positive) confirm that the model balances precision and recall well for both classes, since each F-1 Score lies very close to its precision and recall score. An overall accuracy of 84.4% tells us that roughly eight out of ten reviews are classified correctly, and the ROC AUC of 0.844 indicates this model's strong ability to distinguish positives from negatives: in about 84% of random positive-negative pairs, the model will give the higher score to the positive review. By computing the macro-average F-1 Score of 0.845, we treat both classes equally and see that performance is roughly the same for both negatives and positives. The weighted average F-1 Score is also 0.845, indicating that even if we account for class size differences, neither class dominates the overall score, and therefore class imbalance does not skew our evaluation. Overall, the model is accurate and well-balanced, minimizing both false positives and false negatives without sacrificing one for the other.

7 Insights and Discussions

From these results, although overall performance is strong, it still makes systematic mistakes, misclassifying about 12-19% of reviews. In a real-world customer-feedback environment, misclassifying neutral or positive reviews as negative may trigger unnecessary customer service interventions, while missing genuinely negative reviews could let dissatisfied customers slip through. As such, it is important to weigh the cost of each error type, possibly adjusting the decision threshold or use asymmetric loss functions in order to bias the model towards minimizing more expensive mistakes. Furthermore, since this model was trained on generic Amazon product reviews, domain-specific language such as technical jargon may still trip up the model, and performing error analysis by product category could reveal whether certain niches require specialized models or additional fine-tuning data. In addition, very short reviews consisting of single words or emojis can be ambiguous, and one should therefore consider filtering them out or augmenting training data with more of these edge cases. In future work, one could consider combining BERT with traditional ML classifiers (such as an ensemble using BERT for long and complex reviews and a lightweight logistic regressor for short reviews), potentially capturing the strengths of complementary methods. Furthermore, integrating attention-based explainers would help stakeholders identify which words drive positive versus negative classifications, improving trust and revealing common failure modes. Finally, as new reviews pour in, one could set up period re-training to maintain accuracy over time and adapt to evolving slang and product trends. In conclusion, this well-tuned classifier and those like it not only aids in customer service, but can also drive higher-level analytics: tracking brand sentiment over time, correlating review sentiments with spikes or dips in sales, and informing marketing strategies. By understanding both the model's strengths (balanced class performance) and its areas for improvement (edge-case handling, threshold calibration), it can become a robust component in an end-to-end feedback loop, continually refining business understandings, product recommendations, and more crucial components to the company and user experience.

8 References

- Amazon Product Reviews Dataset: `https://www.kaggle.com/datasets/arhamrumi/amazon-product-reviews`
- Scikit-learn Documentation: `https://scikit-learn.org`
- HuggingFace Transformers: `https://huggingface.co/transformers/`
- Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). *SMOTE: Synthetic Minority Over-sampling Technique*.