EASTERN UNIVERSITY

Machine Learning Project Overview

DTSC691: Applied Data Science

Brea Koenes

Objective and Scope

Objective and purpose

Starbucks is exploring the use of social media data to gain insights into product outages. Traditional inventory stockout records can be inconsistent, and customer feedback shared on platforms like Facebook, Instagram, and Uber Eats may offer a complementary perspective on product availability issues. This project aims to build a high-performing text classification model to identify posts related to product outages, with the goal of maximizing the F1 score. If successful, the model could help Starbucks better understand which outages customers care about most, highlight gaps in existing inventory reporting, and support more informed inventory prioritization strategies.

Scope

The scope of the project includes preprocessing text data, training classification models, and tuning them for optimal F1 score performance. The final model is deployed in a web application that allows classification of user-inputted text. The project is limited to the “Text” column, which contains social media text; other columns are not included in the scope of this project. Additionally, no analysis of patterns or business conclusions are drawn from the classified results.

Dataset Description

Dataset overview

This project uses 2 datasets containing customer sentiment from social media platforms. Starbucks’ internal data science team provided these 2 datasets to be used for this specific use-case. The first dataset, “Facebook”, includes 76,037 text posts about Starbucks from Facebook and Instagram. The second dataset, “Uber Eats”, contains 84,055 reviews from customers who ordered Starbucks through the Uber Eats app. Both datasets include columns such as dates, customer text, and platform identifiers.

Preprocessing

Minimal preprocessing was applied to the text data prior to receiving the data. In the customer text column, Starbucks replaced numerical values with hashtags and removed emojis. The text still requires cleaning before model training.

Exploratory Data Analysis

Process and rationale

Exploration of the “Text” column in both datasets began with the display of random samples. Additionally, a manual review of several hundred rows was done to develop an understanding of the content. The manual review confirmed a significant class imbalance: posts related to product outages represent only about 1.5% of the total data. This imbalance underscores the need for class balancing techniques to ensure the classifier performs effectively on the minority class.

Next, summary statistics were generated to evaluate each dataset’s size and data quality. These included total row counts, number of duplicates, null values, and maximum character length per post. The datasets are relatively balanced in size, as also seen in the previous section. Duplicate entries were identified in both—1,790 duplicates in “Facebook” and 22,091 in “Uber Eats.” The “Facebook” dataset contains no null values, but “Uber Eats” includes 114 rows with null text. In general, Facebook posts are longer, with a maximum post length of 10,510 characters, compared to 5,776 characters in the Uber Eats dataset. These findings reveal data quality issues that could negatively affect model performance if not addressed.

To further understand the dataset composition, a bar chart was created to visualize the number of posts by source, confirming a roughly even distribution across platforms. A boxplot of post lengths by dataset highlighted substantial variation in text length and revealed numerous outliers, particularly in the “Facebook” dataset.

Data Preparation and Cleaning

Data cleaning

To prepare the data, basic cleaning was applied. First, all null entries were removed, as they provide no meaningful input. Duplicate entries were also dropped to ensure the dataset reflects unique customer feedback and to prevent overfitting. Given the low frequency of product outage posts in both datasets, ensuring diversity within the training and test sets is essential. For example, if the training set includes 100 outage posts and 10 of them are duplicates, the model would be exposed to a skewed and limited representation of outage language. Similarly, if the test set contains duplicate outage posts, performance metrics could be artificially inflated, failing to reflect how the model would perform on unseen data. The final cleaning step involved trimming excessively long posts to a maximum of 1,000 characters. Extremely long posts would introduce noise and slow down the model.

Data sampling for labeling

To construct a labeled dataset for training, a targeted sampling approach was used to emphasize representativeness and class balance. From each dataset—Facebook and Uber Eats—2,000 posts were sampled and split into training (80%) and test (20%) sets. The test set was sampled randomly, stratified across each dataset, to reflect the natural class imbalance of each set and provide realistic evaluation metrics.

For the training set, a stratified sampling approach was applied to oversample posts containing product outage-related keywords from each dataset: 40% of the training set consisted of flagged posts, while the remaining 60% was drawn from non-flagged posts. The set of outage-related keywords (e.g., “out of stock,” “sold out,” “restock”) was defined based on manual data exploration and input from Starbucks domain experts. Manual review showed that approximately 20% of keyword-flagged posts were false positives, which helps maintain an estimated 20% outage rate in the training set. This strategy helps address the severe class imbalance without resorting to synthetic data generation, which would risk misrepresenting the diversity and nuance of real outage-related posts.

After sampling, empty label fields were added to the datasets for manual annotation. Each of the 4,000 sampled posts was labeled as either a product outage (1) or not a product outage (0). Labeling was performed by me, with support from Starbucks domain experts for posts that were ambiguous. Once labeling was completed, the finalized training and test sets were imported back in.

Feature Engineering

Approaches

3 feature engineering approaches were implemented to represent the text data. First, a traditional TF-IDF pipeline was developed. Text was preprocessed by converting to lowercase, removing URLs and punctuation, tokenizing, lemmatizing, and removing stop words. Gensim’s Phraser was applied to detect multi-word expressions, based on Starbucks’ guidance that it outperformed traditional TF-IDF n-grams in capturing relevant phrases in this data. TF-IDF vectorization was then applied using optimal parameters: max_features=2700, min_df=2, and max_df=0.85, which yielded the highest F1 scores during model testing compared to lower and higher numbers.

Second, a BERT-based approach was used, requiring no additional text cleaning after the labeled datasets were imported. Preprocessing was avoided for BERT to preserve the natural language structure critical for contextual understanding. Among the tested models, bert-base-uncased outperformed bert-large, achieving better F1 scores. The optimal configuration included batch_size=16, max_length=128, and the use of CLS token embeddings. Max pooling was tested but underperformed relative to CLS embeddings. The optimal configurations were chosen based on the F1 scores after model training. After embedding generation, embeddings were normalized before modeling.

Third, TF-IDF and BERT embeddings were concatenated to create a combined feature set. This approach was used to test if integrating both frequency-based and context-based representations would improve model performance.

Rationale

TF-IDF features capture the relative importance of words across the corpus, emphasizing frequency-based signals, while BERT embeddings provide contextual representations based on surrounding words, capturing deeper semantic meaning. Given the nature of product outage language (both keyword-driven and context-dependent), this hybrid representation aimed to leverage the strengths of both methods. Combining these representations enabled the model to leverage both syntactic and semantic information. All 3 approaches were carried forward into model training to identify the one that maximizes F1 score.

Model Training and Tuning

Intent

The goal of model training was to build a classification model that can accurately identify social media posts referencing product outages. This is a binary classification task, where the input is a text post and the target label indicates whether the post refers to a product outage (1) or not (0).

Model Training Rationale

4 classification algorithms—Logistic Regression, Support Vector Machine (SVM), XGBoost, and LightGBM—are chosen. These model types were chosen based on their strengths in text classification, ability to handle class imbalance, and scalability.

- Logistic regression: chosen for simplicity and known good performance on spare, high-dimensional text data such as TF-IDF features.
- SVM: well-suited for high-dimensional text classification problems and has a strong ability to maximize margin between classes.
- XGBoost: included for its reputation as a high-performance gradient boosting framework. It is especially good at learning from structured data with non-linear relationships.
- LightGBM: scalable, quick, and able to handle large feature sets with imbalanced data. It supports leaf-wise tree growth and histogram-based learning, which can make it result in better performance.

NOTE: In my project proposal, I said, “if these models do not achieve a F1 score above 0.9 after tuning, I will also test BERT models.” This is referencing improving the BERT embeddings the models are trained on, which I tested and found that bert-base-uncased performed best (see feature engineering section).

Baseline model

To establish a performance baseline, a naive logistic regression classifier was created that always predicts the majority class. Because the dataset is highly imbalanced, the naive model achieves deceptively high accuracy by always predicting the majority (non-outage) class. However, its F1 score for the minority class is 0, illustrating its inability to identify true outages. This F1 of 0 is the baseline against which all other models are evaluated.

Model training

12 models are trained from the 4 model types, each type using the 3 feature representations (TF-IDF, BERT, combined). All models were trained using 5-fold stratified cross-validation to preserve class distribution. Class weighting was applied to address imbalance, and F1 score was used as the primary evaluation metric due to its ability to balance precision and recall in the presence of rare classes.

Out of the 12 models trained, the top 4 based on cross-validated F1 score were:

- Logistic Regression with TF-IDF (F1 = 0.7583)
    - Max iterations set to 2000 to ensure convergence during training
- SVM with TF-IDF (F1 = 0.7443)
    - Iterations determined automatically by solver until convergence
- XGBoost with TF-IDF (F1 = 0.7319)
    - Iterative boosting model; number of rounds determined by cross-validation
- LightGBM with Combined Features (F1 = 0.7596)
    - Iterative tree boosting; CV used to determine optimal iterations

While all performed competitively, LightGBM with combined TF-IDF and BERT embeddings emerged as the best model, achieving the highest cross-validated F1 score. This outcome aligns with theoretical expectations: LightGBM’s gradient-boosted decision trees are capable of capturing non-linear interactions and perform particularly well with high-dimensional feature spaces. This positions it uniquely to extract value from both syntactic frequency patterns and semantic context—an essential capability when classifying subtle outage-related language in social media.

Model tuning

Following this initial model selection, the 4 top models were tuned to reflect real-world deployment conditions. While training was performed on an oversampled training set to ensure sufficient and representative examples of the minority class, tuning was done using a validation set that maintains the true class distribution of the original data (approximately 1.5% outages). This strategy simulates production use cases, where the model will encounter very few outage posts among a large stream of irrelevant content. Hyperparameters and decision thresholds were optimized using this validation set to ensure model generalizability under realistic class imbalance. Basic hyperparameters were chosen to be tuned for each model type, selected to ensure each model received relatively equal tuning. The threshold range tested during tuning was 0.1 to 0.9, surrounding the default threshold of 0.5.

F1 scores dropped when models were evaluated on the validation set with the more severe class imbalance. Below are the F1s and optimal parameters found, including the max_iter set for logistic regression during tuning:

- Logistic Regression with TF-IDF (F1 = 0.4444)
  - 'max_iter': 1000 for faster tuning
  - 'C': 10, 'penalty': 'l2'
  - Threshold: 0.3
- SVM with TF-IDF (F1 = 0.5000)
  - 'C': 0.1, 'kernel': 'linear'
  - Threshold: 0.8
- XGBoost with TF-IDF (F1 = 0.5000)
  - 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 300
  - Threshold: 0.600
- LightGBM with combined (F1 = 0.5833)
  - 'learning_rate': 0.1, 'max_depth': 6, 'n_estimators': 200
  - Threshold: 0.5

LightGBM achieves the best F1 score again, which shows its balance between precision and recall. Precision and recall were also outputted to help with model selection. In this scenario, LightGBM achieved a precision of 0.4667 and recall of 0.7778. This reflects a desirable tradeoff: the model captures nearly 78% of actual outage posts, while keeping false positives within a reasonable range. In scenarios like outage detection, recall is important—missing an outage post is typically costlier than flagging a few extra posts for manual review.

Additionally, log loss and confusion matrices were outputted for each tuned model. LightGBM had the lowest log loss after tuning (0.0650), which indicates that the model not only makes correct classifications, but also assigns high confidence to its predictions. This is important in real-world applications, where downstream systems may rely on model confidence scores to prioritize which posts to review or escalate.

LightGBM’s confusion matrix revealed:

- 7 true positives (correctly identified outage posts)
- 2 false negatives (missed outages)
- 8 false positives (incorrectly flagged as outages)
- 623 true negatives (correctly identified non-outage posts)

These results represent a good performance, especially considering the scarcity of outage posts. Other models had worse confusion matrix performance: less true positives were identified or there was an increased amount of incorrectly identified outages.

Model selection

In summary, LightGBM with combined TF-IDF and BERT features is selected as the final model due to its superior performance across multiple evaluation metrics and its ability to integrate sparse and dense textual representations. Its success under severe class imbalance, both in terms of F1 score and log loss, shows that it generalizes well to the actual data it will be given. Furthermore, its decision tree ensemble design allows it to capture complex feature interactions that linear models cannot, especially when those features include both token frequency and contextual semantics. This makes it not only the empirically best model, but also the most robust and interpretable choice for deployment for real-world social media outage detection. In the future, Starbucks will be using the best model for highly imbalanced datasets like this one.

Model Evaluation

Test set evaluation

The best-performing model—LightGBM trained on combined TF-IDF and BERT embeddings—was retrained on the full training dataset using the optimal hyperparameters and threshold values found. This retraining step ensures the model has access to the maximum amount of labeled data for learning, which can improve generalization. It was then evaluated on a held-out test set reflecting real-world class imbalance (~1.5% outage posts), which was not used during feature selection or tuning.

Performance metrics

The primary metric used is the F1 score for the product outage class, which provides a balanced measure of precision and recall. The final F1 score on the test set is 0.6316, indicating solid performance given the extreme class imbalance and limited outage examples in the training data. This is substantially larger than our baseline F1 of 0, which was found by the naive model. Supplementally, the model achieved a precision of 0.8571, meaning that when it predicts a post as referring to a product outage, it is usually correct. However, recall is 0.5, showing that while the model is conservative and avoids false positives, it does miss some true outage posts—an expected trade-off in highly imbalanced datasets.

In addition to classification metrics, log loss is calculated to assess the model's probabilistic calibration. A low log loss of 0.0348 confirms that the model not only makes correct predictions but does so with high confidence and well-calibrated probability estimates. The confusion matrix further contextualizes these metrics: out of the total test examples, the model correctly classified 6 posts as product outages (true positives) and 787 posts as non-outages (true negatives), while 6 true outage posts were misclassified as non-outages (false negatives) and only 1 non-outage post was incorrectly classified as an outage (false positive). This shows the model maintains a low false positive rate. This is an important trait for practical deployment in systems where false alerts could create unnecessary interventions.

To deepen the evaluation, a manual review was conducted of both misclassified posts. Among the false negatives, several examples contained clear product outage indicators, including phrases like “mine is out too,” “store out of product,” and “Starbucks said no bagels.” These posts suggest that while the model is effective at identifying frequent outage expressions, it can still struggle with nuanced or less common language patterns. This limitation is likely due to insufficient representation of these variations in the training data. As a result, future work should include retraining the model with a larger and more diverse labeled dataset to improve recall and robustness across different linguistic formulations of outage-related content.

Results and discussion

LightGBM with combined features effectively captured nuanced linguistic patterns that single-feature models could not. It was the only model that was able to perform exceptionally well on combined TF-IDF features and BERT embeddings; the others performed better with only TF-IDF features. The inclusion of BERT embeddings likely enabled the model to interpret contextually rich or indirectly phrased outage posts, while TF-IDF grounded it in surface-level signals like keywords and phrase frequency.

Despite the substantial class imbalance in the test set, the model achieved a respectable F1 score of 0.6316, precision of 0.8571, and recall of 0.5000. Its F1 score, achieved using only 4,000 labeled examples and a rare target class, suggests an efficient use of data and strong model generalization. These qualities make the approach practical and scalable.

The high precision indicates that when the model flags a post as an outage, it is usually correct—an important consideration in operational settings where false alarms could lead to wasted interventions. The recall, while moderate, reflects the difficulty of capturing all relevant outage posts given the limited and imbalanced training data. These results suggest the model is learning meaningful, generalizable patterns from a relatively small amount of labeled data.

In an applied setting, the model can serve as an early warning system for inventory teams by flagging social media signals related to product outages. It could supplement internal inventory data with customer-reported issues, enabling faster, data-driven responses to stock disruptions. Real-time integration into monitoring systems, anomaly detection pipelines, or feedback loops could help Starbucks better understand and respond to operational issues surfaced by customers online. In addition, it could help Starbucks identify which products that go out of stock have the most customer sentiment.

Active learning

Improvement strategy

After final model evaluation, active learning was applied to evaluate whether it could improve model performance for potential large-scale deployment by Starbucks. A new set of

3,000 unlabeled posts, separate from the original training and test sets, was selected for this experiment. These posts were preprocessed using the same combined TF-IDF and BERT feature pipeline and then passed through the final LightGBM model to generate predicted probabilities.

To identify the most informative examples for manual labeling, posts predicted as outages or ones with uncertain predictions (probabilities close to 0.5) were selected. This was to target data points likely to refine the model. A total of 42 posts met these criteria and were manually reviewed and labeled, again using domain expertise to resolve ambiguous cases.

These newly labeled posts were appended to the original training set and the combined feature representations were regenerated. The LightGBM model was retrained with this expanded dataset and evaluated on the same held-out test set used previously. Evaluation metrics remained unchanged in terms of F1 score, precision, and recall. Log loss was 0.0340, which is slightly lower but not significant. The confusion matrix also showed identical classification results, with no additional true positives or reductions in false negatives.

Results

This result suggests that, at least at this small scale, active learning did not lead to performance improvements. While the approach was methodologically sound, the limited number of new, diverse outage examples may not have been sufficient to shift the model's decision boundaries meaningfully. For active learning to be impactful, a larger batch of uncertain or product outage examples would likely be needed. Nonetheless, this experiment establishes a repeatable workflow that could be scaled for Starbucks' future use.

User-Interface Deployment

Deployment process

The final model was saved and deployed using a Streamlit web application, providing an interactive user interface for demonstrating model functionality. Streamlit was chosen over Amazon Web Services (as stated in the project proposal) due to ease of use. All components of the preprocessing pipeline—including the Gensim Phraser, TF-IDF vectorizer, and BERT embedding scaler—were serialized and integrated into the app. When a user inputs text, it is processed using the same combined feature engineering pipeline used during model training, and predictions are generated using the best-performing LightGBM model.

Overview

The web application includes multiple sections: a homepage, résumé, project portfolio, and page deploying the best classification model. The page deploying this model provides a comprehensive overview of the problem statement, methodology, and results. Most importantly, it includes an interactive module that allows users to input text and receive real-time predictions on whether it is product outage-related. This deployment not only showcases the model's capabilities but also enables direct engagement for instructors.