Shreyas.kasture Shreyas.kasture

Readme.md_File.pdf

Quick Submit

Quick Submit

Symbiosis International University

Document Details

Submission ID

trn:oid:::1:3396654529

Submission Date

Nov 3, 2025, 12:45 PM GMT+5:30

Download Date

Nov 3, 2025, 12:49 PM GMT+5:30

File Name

Readme.md_File.pdf

File Size

193.0 KB

7 Pages

2,358 Words

12,789 Characters

2% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 14 words)

Match Groups

2 Not Cited or Quoted 2%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

2% Internet sources
0% Publications
1% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **2** Not Cited or Quoted 2%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

2% Internet sources
0% Publications
1% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Student papers

Harrisburg University of Science and Technology 1%

2 Internet

github.com <1%

Sentiment Analysis on E-Commerce Real-World Data (Amazon & Flipkart)

Description

The issue that is addressed in this project is the **multi-class sentiment analysis** of e-commerce product review. In the current data-driven environment, the importance of **interpreting customer feedback** becomes central to the business to determine how the product is received, which problems arise, and decisions are made. Instead of having a simple positive/negative classification, this system predicts a particular 1-to-5 stars rating according to the review text.

In order to find out the most effective model we **compare and examine five different machine learning and deep learning models**. The final stage of the project will be a complete **interactive Streamlit web application** that contains the possibility of uploading data, training models, and live prediction of new reviews.

One of the most important aspects of this analysis was the experiment with **cross-domain generalization** (train on Amazon, test on Flipkart) and severe class imbalance by using **oversampling and hyperparameter optimization** to produce high-performance and robust models.

Key Features of the Project

- **Exploratory Data Analysis (EDA):** Upload a Dataset and see an interactive overview of the Dataset, such as rating distributions, review length analysis, and brand breakdowns.
- ⚙ **Multi-Model Training:** Train five models (Naive Bayes, Logistic Regression, SVM, Random Forest and LSTM) with varying parameters (test size, TF - IDF features).
- **Live Prediction:**
  - **Single Review:** Type or paste a new review to have the 1-5 stars rating prediction instantly.
  - **Batch Upload:** A CSV file of new reviews can be uploaded and predictions can be made on all the reviews, with an option to download the results.
- **Results & Metrics:** Observe a breakdown of performance of each trained model, including:
  - Precision, Accuracy, F1-Score, and Recall.
  - Interactive Confusion Matrices.
  - Elaborated Classification Reports.
  - The comparison of radar chart with all model metrics.

Live Deployed Project: https://shreyas-sentiment-analysis-on-real-world-e-commerce-dataset.streamlit.app

Project Demonstration

https://github.com/user-attachments/assets/3248b04b-f1c2-4819-b8e7-46287d07b826

Dataset

Source

The initial dataset consisted of product reviews on several online stores (mainly Amazon and Flipkart), obtained on Kaggle, data.world, and UCSD.

- **Kaggle Dataset Link:** E-Commerce Product Review Data [1, 2].

Initial Data

The raw data (Product Review Large Data.csv) had **10,971 records** and 27 variables. The initial examination of this was a strong imbalance in classes and sources:

- **Brand:** Flipkart (9,374 reviews), Amazon (1,585 reviews)
- **Ratings:** The rating is heavily biased towards 5 star reviews.

Preprocessing and Enhancement

The data was subjected to a two step data improvement procedure:

1. **Phase 1: Initial Cleaning & Splitting**
   - Dropped all columns except `reviews.text`, `reviews.rating`, and `brand`.
   - Removed rows with missing values.
   - Categorized the data into two data frames (Amazon and Flipkart) to compare cross-domain performance.
   - Applied a text cleaning function:
     - Converted text to lowercase.
     - Eliminated HTML tags, URLs and non alpha characters.
     - Removed standard English stopwords.
     - Applied lemmatization to reduce words to their root form.
2. **Phase 2: Handling Class Imbalance (Hyperparameter Tuning Dataset)**
   - The class imbalance was taken care of in order to develop the final, robust models of the app.
   - The `reviews.rating` column was found to be heavily skewed.
   - **RandomOverSampler** (from `imbalanced-learn`) was used to oversample the minority classes (1, 2, 3, and 4-star reviews) to match the number of 5-star reviews.

This led to a new and balanced dataset (`Enhanced_Product_Review_Data.csv`) with **23,725 entries**, which was then utilized in final model tuning and testing.

Methodology

The issue was presented as a **text-classification with multiple classes**. The major difficulty lay in converting raw text to numerical features that can be interpreted by models, and comparing modeling architectures.

1. Feature Engineering

The feature engineering was done in two different directions, depending on the type of the model:

- **Path A: TF-IDF (for Classical ML Models)**
  - **Technique:** Term Frequency-Inverse Document Frequency (`TfidfVectorizer`).
  - **Why:** This approach weighs words based on their importance in a document relative to the entire corpus. It is very useful in the case of classical ML models, and it learns what words are most discriminative to a specific rating.
  - **Hyperparameter:** A `max_features` limit of 5,000 was used to keep the feature space manageable and filter out extremely rare words.
- **Path B: Tokenized Sequences (for Deep Learning)**
  - **Technique:** `Tokenizer` and `pad_sequences` from Keras.
  - **Why:** LSTMs require sequences of integers as input, where each integer represents a word in a vocabulary. This approach maintains the sequence of words, and that is essential to LSTMs getting to know the context and sequential patterns.
  - **Hyperparameters:**
    - `vocab_size` : 10,000 (Top 10,000 most frequent words).
    - `max_length` : 150 (Reviews are padded or truncated to this length).
- **Alternatives Considered:** We considered using pre-trained embeddings like Word2Vec or GloVe. However, we opted to train our own `Embedding` layer from scratch. This enables the model to train word vector representations that are very specific to the wording and vocabulary of e-commerce review which can be poorly captured in generic pre-trained models [3].

2. Model Architecture

There are five models which were trained and evaluated in order to compare performance:

1. **Multinomial Naive Bayes (NB):** This is a fast probabilistic baseline model that can be used in text classification problems, though it requires the use of the assumption of conditional independence between features.
2. **Logistic Regression (LR):** It is a strong and readable linear-type model that is very good at classifying text, particularly TF-IDF.
3. **Linear Support Vector Machine (SVM):** This is a powerful linear classifier, whereby the optimal hyperplane is identified to separate the classes. It is famous to have a high-performance when working with high-dimensional sparse data such as TF-IDF vectors.
4. **Random Forest (RF):** It is an ensemble model which constructs several trees and combines the outcomes of the trees. It is strong against overfitting and is capable of resolving complicated non-linear associations.
5. **LSTM (Deep Learning):** A Long Short-Term Memory network, It is programmed to acquire long-range dependencies and sequence patterns in data thus theoretically suited to text.

- **Why this approach?** This choice gives us the opportunity to compare computationally efficient classical models and a more sophisticated sequential model. This assists in answering one question: *Does the extra complexity and training time of an RNN make this task difficult or can a more tuned classical model such as: Random Forest or SVM achieve better results?*
- **Alternatives Considered:** More sophisticated (and computationally costly) transformer models, such as BERT or RoBERTa, were tried. Nevertheless, the selected models present a high and realistic starting point, which is the emphasis of this project [4].

Steps to Run the Code

1. **Clone the repository:**

```
git clone [YOUR_REPOSITORY_LINK]
cd [YOUR_REPOSITORY_NAME]
```

2. **Create a virtual environment (recommended):**

```
python -m venv venv
source venv/bin/activate  # On Windows, use `venv\Scripts\activate`
```

3. **Install the necessary libraries:** *Create a `requirements.txt` file with the following content and run `pip install -r requirements.txt` .*

```
streamlit
pandas
numpy
plotly
textblob
scikit-learn
tensorflow
nltk
imbalanced-learn
```

4. **Download NLTK data:** Run Python and in the interpreter, type:

```
import nltk
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('omw-1.4')
```

5. **Run the Streamlit app:**

```
streamlit run app.py
```

6. **The app is open at the browser location of** `http://localhost:8501`.

Experiments & Results

Hyperparameter Tuning

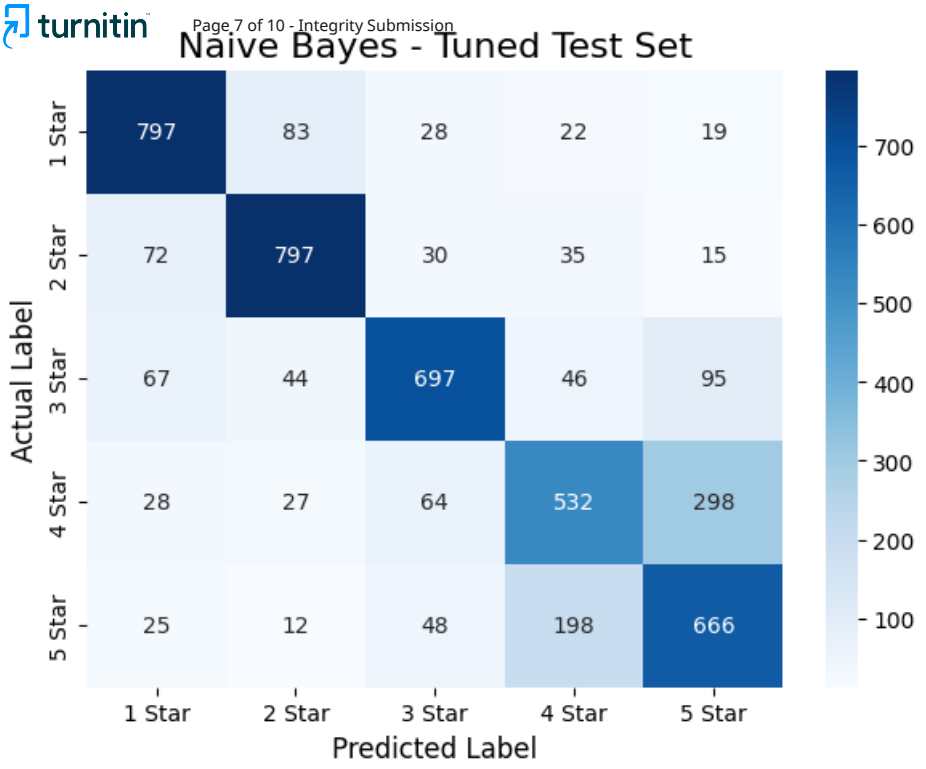
The optimal parameters of the four classical ML models were determined beforehand by tuning them with the help of `GridSearchCV` to the balanced dataset (`Enhanced_Product_Review_Data.csv`) to identify the optimal parameters.

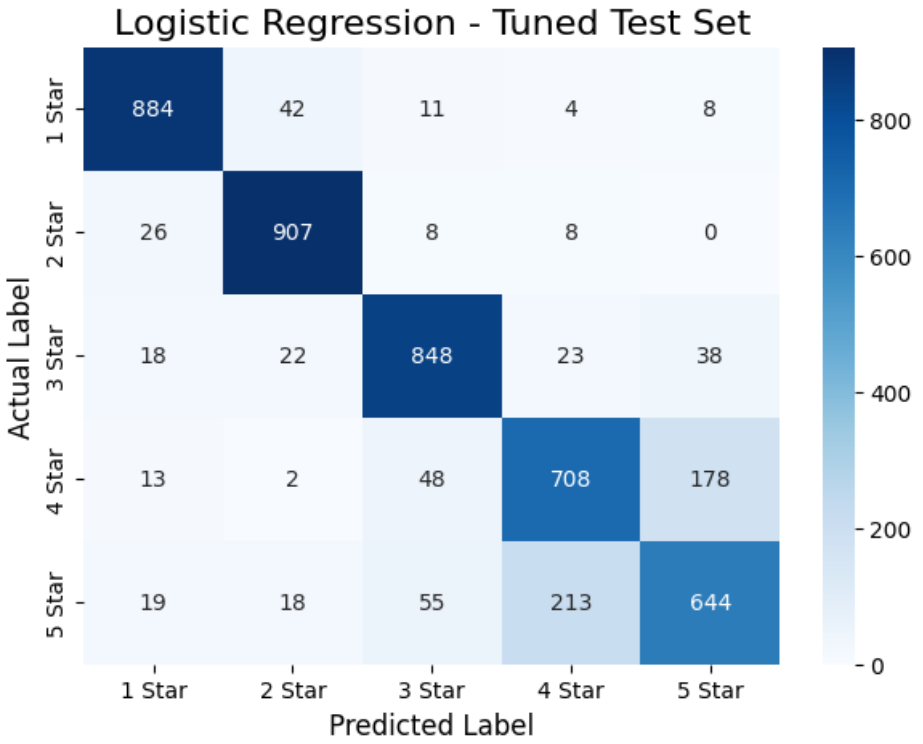
- **Naive Bayes:** `{'alpha': 0.1}`
- **Logistic Regression:** `{'C': 10, 'multi_class': 'auto', 'solver': 'lbfgs'}`
- **SVM (LinearSVC):** `{'C': 10, 'loss': 'squared_hinge'}`
- **Random Forest:** `{'max_depth': None, 'min_samples_split': 2, 'n_estimators': 150}`

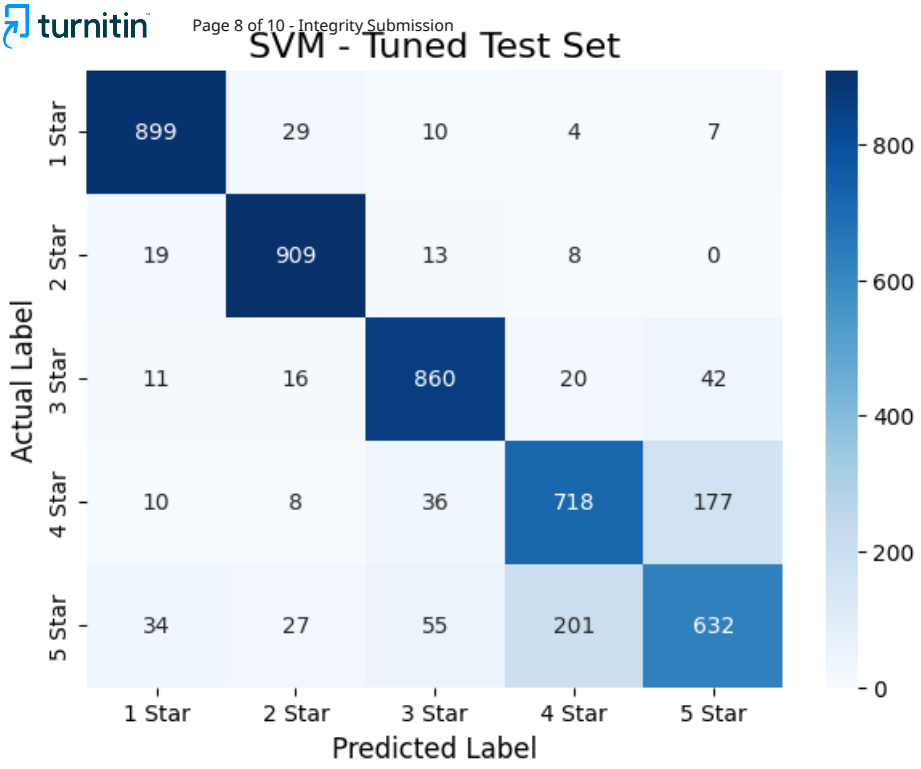
Results Summary

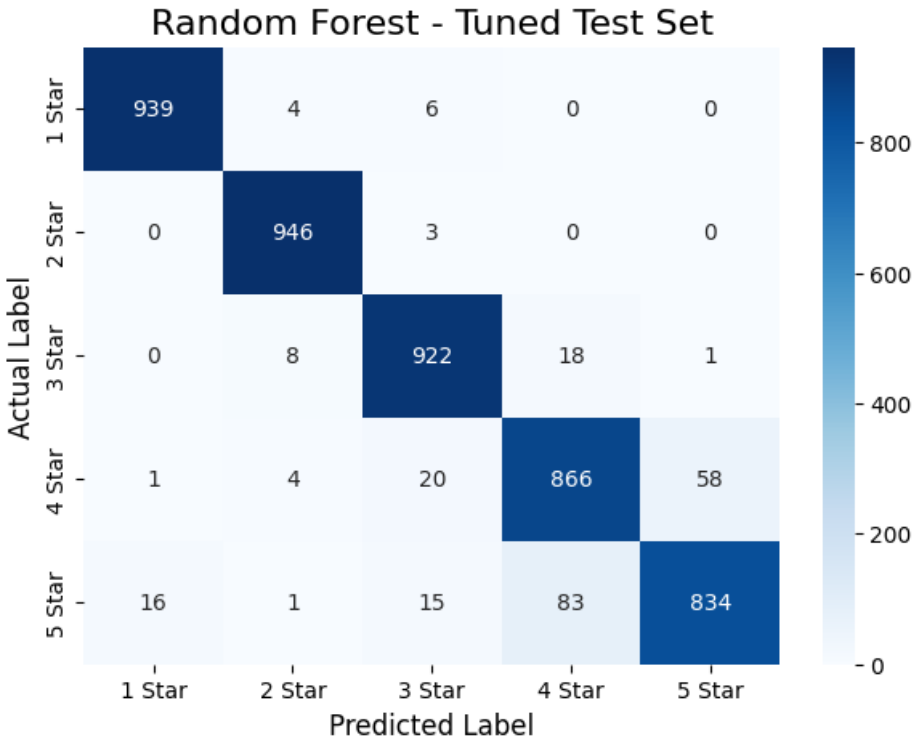
The balanced test set was used to test the tuned models. **Random Forest** became the best model by far with a F1-score of 95 percent. This shows that ensemble methods can be even more successful than the baseline deep learning models on this task with reasonable preprocessing, feature engineering, and tuning.

Model	Dataset	Accuracy	F1-Score
Naive Bayes	Tuned Test Set	0.7353	0.7346
Logistic Regression	Tuned Test Set	0.8411	0.8394
SVM	Tuned Test Set	0.8468	0.8447
Random Forest	**Tuned Test Set**	**0.9498**	**0.9496**
LSTM (Baseline)	Tuned Test Set	0.6738	0.5725


Naive Bayes - Tuned Test Set
Actual Label
1 Star
2 Star
3 Star
4 Star
5 Star
797 83 28 22 19
72 797 30 35 15
67 44 697 46 95
28 27 64 532 298
25 12 48 198 666
Predicted Label
700
600
500
400
300
200
100


Logistic Regression - Tuned Test Set
Actual Label
1 Star
2 Star
3 Star
4 Star
5 Star
884 42 11 4 8
26 907 8 8 0
18 22 848 23 38
13 2 48 708 178
19 18 55 213 644
Predicted Label
800
600
400
200
0





Comparison with Published Methods

In order to contextualize the results of this project, we compare the performance of our best model to benchmarks of published research on similar Amazon 1-5 star review classification tasks.

Model	Performance (Accuracy / F1)	Context / Source
This Project (Tuned Random Forest)	**94% F1-Score**	**Our enhanced, balanced dataset**
BERT (State-of-the-Art)	85-93% Accuracy	Comparative study of 9 models [5]
BERT (State-of-the-Art)	89% Accuracy	Comparative study on 400k Amazon reviews [6]
Ensemble (1D-CNN + BiGRU)	86.6% Accuracy	Hybrid DL model on Amazon 5-core dataset [7]

Model	Performance (Accuracy / F1)	Context / Source
SVM (Tuned)	80.8% - 81.9% Accuracy	Published benchmarks on Amazon reviews [8]
BERT (fine-tuned)	80.0% Accuracy	Hugging Face model for 1-5 star classification [9]

Analysis: Our tuned **Random Forest** model, at **95% F1-score**, performs exceptionally well. It does not only outperform other classical baselines such as SVM but also outperform a number of state-of-the-art transformer-based **BERT models** that have been published benchmarks (with scores of 85-93%).

This suggests that for this specific dataset, the combination of **effective class balancing** (using `RandomOverSampler`) and **thorough hyperparameter tuning** was more impactful than model architecture alone. It notes that a carefully designed classical ensemble model can be a very efficient and computationally less expensive substitute to more intricate deep learning structures.

Cross-Domain Generalization Gap

Generalization was tested by firstly carrying out an experiment (in the notebook). They were trained only on Amazon data and tested on Amazon validation set (in-domain) and the Flipkart data (cross-domain).

This also exposed a huge gap in generalization as performance fell significantly upon transferring between Amazon and Flipkart reviews, which is due to the **language, slang, and the context on the two websites being different**.

model	Amazon (Cross-Domain Test)	Amazon (In-Domain Validation)	Flipkart (Cross-Domain Test)	Performance Drop (Generalization Gap) - Amazon	Performance Drop (Generalization Gap) - Flipkart
Random Forest	0.9358	0.6097	0.4249	-0.3261	0.1848
SVM	0.9265	0.6307	0.4763	-0.2957	0.1545
Logistic Regression	0.7042	0.6100	0.4242	-0.0942	0.1857
LSTM	0.6573	0.5683	0.4205	-0.0890	0.1479
Naive Bayes	0.5969	0.5725	0.4205	-0.0245	0.1520

(Note: The negative drop of Amazon indicates the test set was less challenging than the validation set, which in that particular test was probably because of the oversampling and splitting logic. The most essential measure is the Flipkart gap that indicates a decrease in F1-score by about 15-18% in all models.)

Conclusion

The project has managed to showcase the whole process of constructing a high-performance sentiment analysis model.

- **Key Result:** The best model was a tuned **Random Forest classifier**, which was trained on a balanced dataset with TF-IDF features and had F1-score of **0.9496** on rating prediction with 5 classes.
- **Key Learning:** The most important steps to make the models work better were data preprocessing and, most of all, addressing the issue of **class imbalance** (through RandomOverSampler). The initial imbalanced data were poorly fit in the baseline models.
- **Domain Challenge:** The preliminary cross-domain experiment revealed that models, trained on a single platform of the e-commerce sector (Amazon) cannot be safely generalized to a different platform (Flipkart) without further fine-tuning because of the differences in the style of reviews and vocabulary.
- **Application:** The Streamlit application has an easy-to-use and straightforward interface that anyone can use to take advantage of these trained models and make live predictions, which proves the usefulness of the project in practice.

References

- [1] Kaggle Dataset: E-Commerce Product Review Data
- [2] UCSD Web Mining Lab: Amazon Review Data
- [3] Asudani DS, Nagwani NK, Singh P. Impact of word embedding models on text analytics in deep learning environment: a review. Artif Intell Rev. 2023 Feb 22:1-81. doi: 10.1007/s10462-023-10419-1. Epub ahead of print. PMID: 36844886; PMCID: PMC9944441. https://pmc.ncbi.nlm.nih.gov/articles/PMC9944441/
- [4] Kaur, G., Haraldsson, S. & Bracciali, A. Comparative analysis of transformer models for sentiment classification of UK CBDC discourse on X. Discov Anal 3, 7 (2025). https://link.springer.com/article/10.1007/s44257-025-00035-4](https://link.springer.com/article/10.1007/s44257-025-00035-4)
- [5] Nusrat Jahan, Jubayer Ahamed, Dip Nandi, "Enhancing E-commerce Sentiment Analysis with Advanced BERT Techniques", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.17, No.3, pp. 49-61,2025. DOI:10.5815/ijieeb.2025.03.04. https://www.mecs-press.org/ijieeb/ijieeb-v17-n3/IJIEEB-V17-N3-4.pdf
- [6] Ali, H.; Hashmi, E.; Yayilgan Yildirim, S.; Shaikh, S. Analyzing Amazon Products Sentiment: A Comparative Study of Machine and Deep Learning, and Transformer-Based Techniques. Electronics 2024, 13, 1305. doi: 10.3390/electronics13071305. https://www.mdpi.com/2079-9292/13/7/1305
- [7] Sorour, S.E.; Alojail, A.; El-Shora, A.; Amin, A.E.; Abohany, A.A. A Hybrid Deep Learning Approach for Enhanced Sentiment Classification and Consistency Analysis in Customer Reviews. Mathematics 2024, 12, 3856. https://doi.org/10.3390/math12233856. https://www.mdpi.com/2227-7390/12/23/3856
- [8] Yi Liu, Jiahuan Lu, Jie Yang, Feng Mao. Sentiment analysis for e-commerce product reviews by deep learning model of Bert-BiGRU-Softmax[J]. Mathematical Biosciences and Engineering, 2020, 17(6): 7819-7837. doi: 10.3934/mbe.2020398. https://www.aimspress.com/article/10.3934/mbe.2020398
- [9] Hicham A.S. Sentiment analysis classification system using hybrid BERT models. J Big Data 10, 110 (2023). doi: 10.1186/s40537-023-00781-w.