CSE 256_FA24: An Exploration into Wine Review Classification through Sentiment Analysis

Author - Isheta Bansal
i1bansal@ucsd.edu

1 Introduction

The goal of this project was to predict user sentiment (Liked or Not Liked) from wine reviews by fine-tuning various machine learning models. The challenge was to handle the nuanced language in reviews and build an accurate model while optimizing computational resources.

Project Objectives:

1. Preprocess the wine reviews dataset and label the sentiment based on review scores: Completed

2. Fine-tune a pre-trained Logsitic Classifier model as baseline on the processed dataset for sentiment classification: Completed

3. Implement other machine learning models on the dataset and compare it to the baseline: Completed

4. Optimize hyperparameters (e.g., learning rate, batch size) to improve model accuracy: Completed

5. Evaluate the model using metrics like accuracy, confusion matrix, and classification report: Completed

2 Related work

The classification of wine reviews has historically been approached through various text classification methods. Past studies in this field have used a variety of techniques, from simple machine learning models to advanced deep learning approaches.

Early work in sentiment analysis relied on methods like bag-of-words and TF-IDF with classifiers such as Naïve Bayes, Support Vector Machines (SVMs), and Logistic Regression. For example, Pang and Lee (Pang and Lee, 2004) showed how machine learning could be used for sentiment analysis, focusing on the importance of labeled data and good feature selection. Logistic Regression, a simple yet effective method, is often used as a starting point for sentiment tasks because it works well with sparse data. Researches (Pedregosa et al., 2011) have highlighted its usefulness when combined with techniques like feature engineering.

In the context of wine reviews, researchers (Panicheva et al., 2018) have used NLP techniques to predict ratings and classify sentiments, using methods like bag-of-words and sentiment lexicons. TensorFlow has also been widely used for building sequential neural network models, which are simpler models designed for tasks like sentiment analysis. Goodfellow et al. (Goodfellow et al., 2016) explained how even basic dense networks can perform well if fine-tuned properly.

Deep learning techniques such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs) became popular as they could handle more complex text patterns. Kim (Kim, 2014) introduced a CNN model that became a standard for many text classification tasks. More recently, models like BERT, introduced by Devlin et al. (Devlin et al., 2019), have revolutionized text analysis with their ability to understand context better through a transformer-based architecture. Xu et al. (Xu et al., 2022) showed how BERT could be applied to wine reviews, significantly improving predictions by using the context of reviews. Sun et al. (Sun et al., 2019) also demonstrated that tweaking hyperparameters like learning rates and batch sizes can further improve the performance of BERT on tasks like this.

In short, the field has evolved from simple machine learning to highly sophisticated deep learning models, making wine review classification a fascinating and challenging topic in NLP.

3 Dataset

The dataset consists of wine reviews from Winemag, containing textual descriptions and numerical ratings. Dataset Statistics: Total samples: 150,000

3.1 Data preprocessing

The dataset is loaded from a CSV file for each of the implemented models. It contains various columns but ones we will be using for this project are ’description’ and ’points’.
Sentiment Labeling: A new column ’Liked’ is created by converting the points to binary labels. If the points score is 90 or above, it’s labeled as 1 (Liked), otherwise, it’s labeled as 0 (Not Liked).The dataset is then reduced to only two columns, ’Review’ and ’Liked’. The class distribution is highly imbalanced, refer 1.

Class	Distribution
0	0.68058
1	0.31942

Table 1: Class Distribution in Imbalanced Dataset

Data Split: The dataset is split into three, train/validation/test as 70/10/20. Stratified sampling has been used to maintain the class distribution in this split as well.
The Models of Logistic Regression, Sequential Neural Network and Random Forest were able to run on the original dataset but running complex ML models like DistilBert Transformer posed a problem due to less computational power. To resolve this, the author along with comparing the various ML models on the imbalanced dataset of 150k samples, has also used a smaller dataset by slicing the dataset and creating a balanced dataset with 50-50 distribution between the ’Liked’ and ’Not Liked’ classes as that would give us another layer of comparison between the models. This *Balanced* dataset has 3000 samples.

4 Baseline

We chose Logistic Regression Classifier as the baseline for this task due to its simplicity. The dataset split was done using train_test_split from sklearn. TF-IDF was used to convert the text data into numerical features.

- Hyperparameters tuning for TfidfVectorizer:
  1. max_features = 1000 (This ensured that only the most relevant words are considered which prevents overfitting.)
  2. stop_words=’english’
  3. lowercase=True
  4. ngram_range=(1, 2)
- Hyperparameters tuning for Logistic Regression:
  1. random_state = 42
  2. max_iter=1000

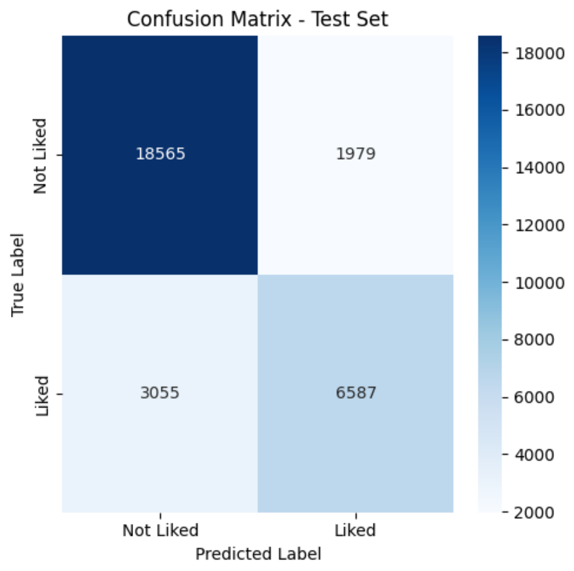

Figure 1: Confusion Matrix for Test Set - Logistic Regression

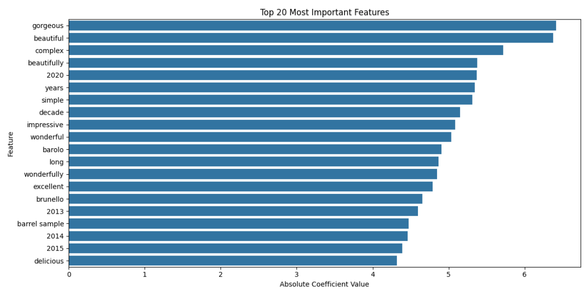

Figure 2: Top 20 features in Logistic Regression Classifier

For Balanced Dataset Observations, refer figures 6, 1, 2.

5 Approach

5.1 Sequential Neural Network

The author implemented a Sequential Model in Keras to classify the wine reviews as ’liked’ or ’not liked’. The layers in this model are stacked one after the other where the output of one layer is passed as input to the next.

- The Embedding Layer is converting the input text to dense vector representations. Words

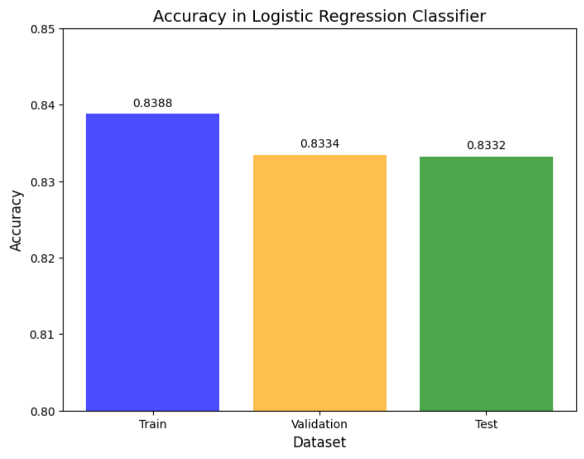

Figure 3: Accuracy of Logistic Regression Classifier across Datasets

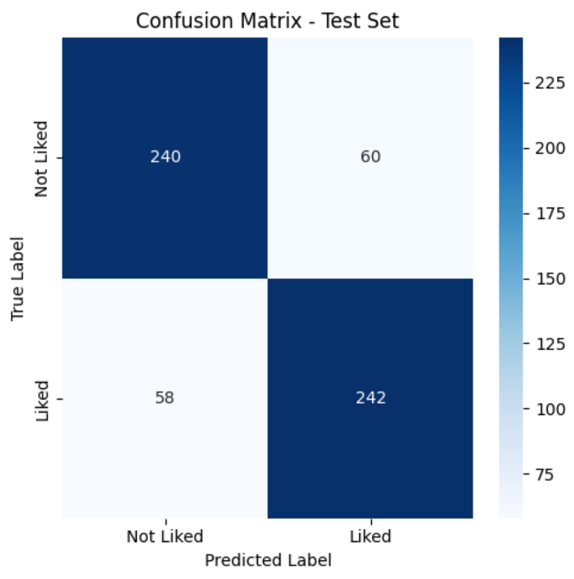

Figure 4: Confusion Matrix for Balanced Test Set - Logistic Regression

with similar meanings will have similar representations. Each word in the review is transformed into a 1000 dimensional vector.

- The GlobalAveragePooling1D layer takes the output of the Embedding layer and computes the average of all word vectors along the sequence dimension. This helps to convert variable-length sequences into a fixed-size vector, which can then be passed to the dense layer for classification. It outputs a 1-dimensional vector of size 1000.

- The Dense layer has neurons which are connected tp every neuron in the previous layer. This layer applies weights and biases to the

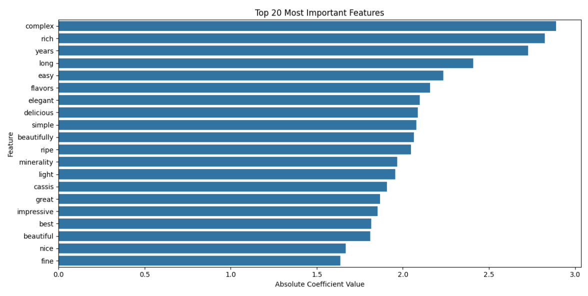

Figure 5: Top 20 features in Logistic Regression Classifier in Balanced Set

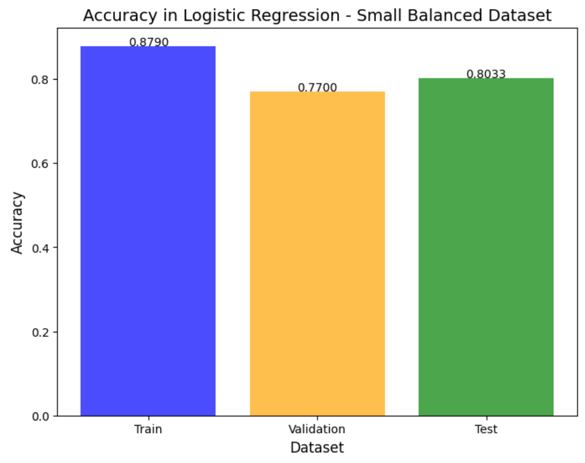

Figure 6: Accuracy of Logistic Regression Classifier across Balace Dataset

input and uses an activation function to produce an output. ReLU is used as the actuvation function as it allows the model to learn complex patterns.

- Hyperparameters:
  1. Embedding Dimension = 1000
  2. Number of neurons in Dense Layer = 128
  3. Learning Rate = 0.001
  4. Epochs = 50

- The final Dense Layer has a single output which can be either 0 or 1 for binary classification. The Sigmoid activation function has been used in this layer to map the output to a probability between 0 and 1. The output represents the probability of the wine review being 'Liked'.

- Adam optimizer has been used along with the Binary Cross-Entropy Loss function. The epochs have been reduced to 50 from 100 as the latter was making the model overfit the train data.

This model posed issues in running on personal machine and Colab. So, we will compare it on the small balanced dataset.

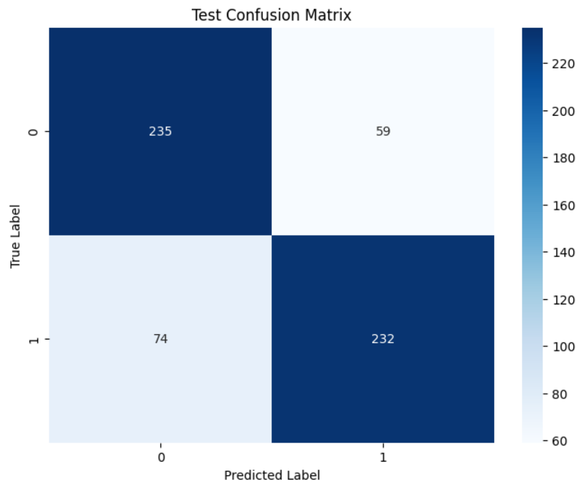

Figure 7: Confusion Matrix for Test Set - Sequential Neural Network

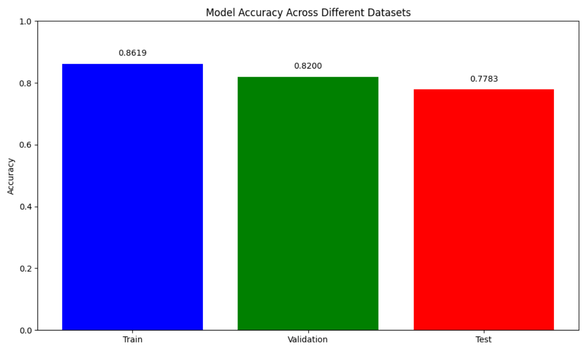

Figure 8: Accuracy of Sequential Neural Network across Datasets

5.2 Random Forest Classifier

As the original dataset is imbalanced, the author researched about methods that would work best in such a scenario. A Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class that has the higher number of votes.

- To handle the class imbalance, it uses the class_weight parameter which is taken as balanced. It automatically adjusts weights to compensate for class distribution which prevents bias towards majority class.

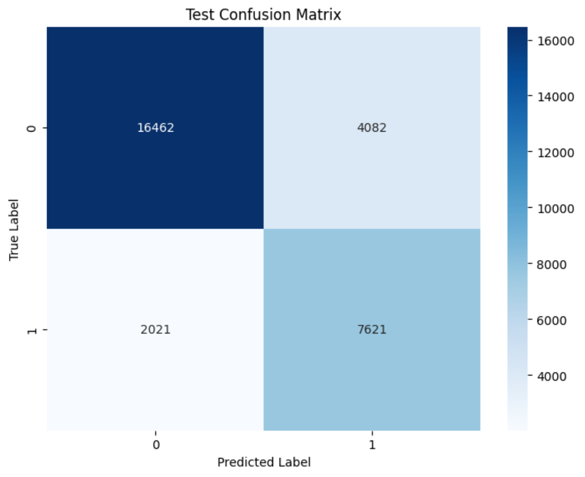

Figure 9: Confusion Matrix for Test Set - Random Forest

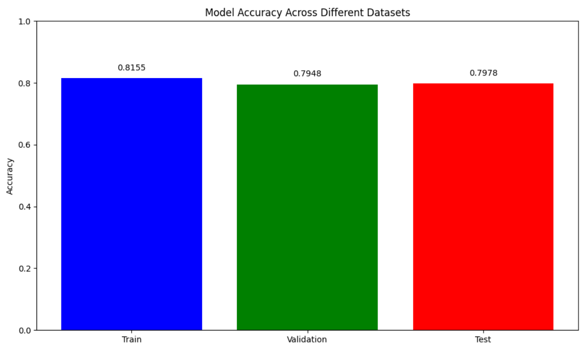

Figure 10: Accuracy of Random Forest Classifier across Datasets

- TF-IDF Vectorizer is being used to convert text into numerical features. It also captures word importance. It also removes English stop words and limits vocabulary to 5000 most important features.
- Each tree is trained on a random subset of data. This allows diverse tree perspectives. The randomness reduces the correlation between tress which in turn improves generalization of the data.
- Random Forest can handle non-linear relationships while also being less prone to overfitting.
- Hyperparameters:
  1. 300 decision trees (estimators)
  2. depth of 20
  3. Minimum 10 samples to split a node
  4. Minimum 5 samples in a leaf node

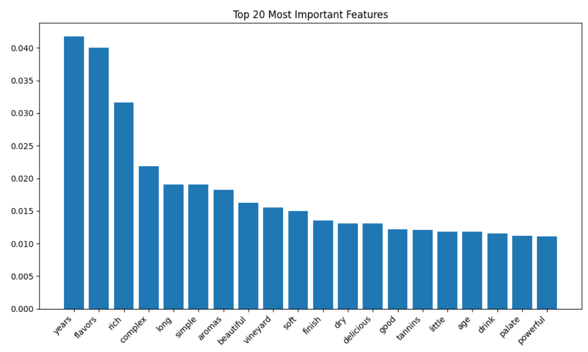

Figure 11: Top 20 Features in Random Forest Classifier

- The model took around 1-2 minutes to run, becoming the fastest out of our approaches.

Class	Precision	Recall	F1 Score
Not Liked (0)	89%	80%	84%
Liked (1)	65%	79%	71%

Table 2: Classification Report for Random Forest Classifier

The model achieved 80% accuracy on Test Set. On the other hand, when run on the small balanced dataset, the model was able to give 76% accuracy on Test Set.

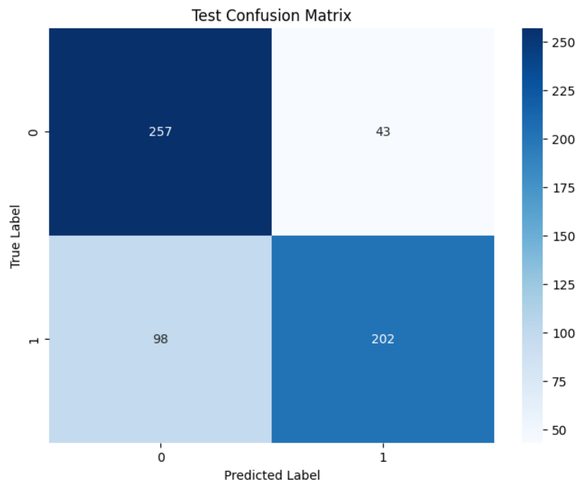

Figure 12: Confusion Matrix for Balanced Test Set - Random Forest

5.3 DistilBERT

The author implemented a DistilBERT-based model with custom pooling methods. DistilBERT is a smaller, faster, and more efficient version of BERT. The author tried various pooling techniques, namely, mean, max and CLS token pooling. This model learns contextual embeddings

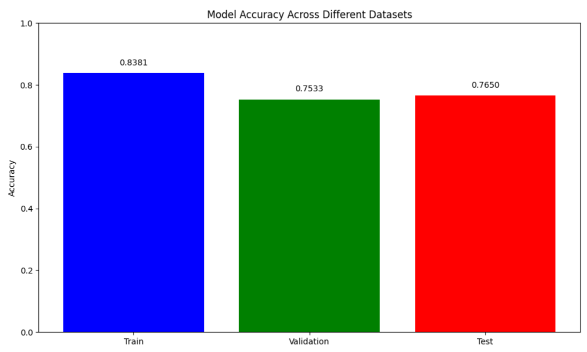

Figure 13: Accuracy of Random Forest Classifier across Balanced Dataset

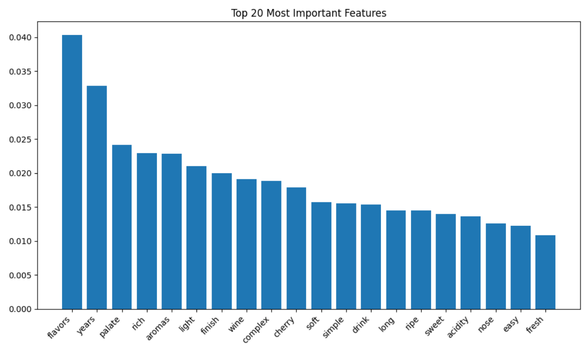

Figure 14: Top 20 Features in Random Forest Classifier for Balanced Dataset

for words (or tokens) and understands the semantic relationships between them. DistilBERT is a compressed version of BERT that retains 97% of its language understanding capabilities but has 60% fewer parameters, making it faster and more memory-efficient. This is the reason author chose this model instead of BERT but unfortunately, the computing constraints didn't allow this model as well to run on the original dataset which is why this will be compared on the small balanced dataset only. The author tried various values for the hyperparameters. The final used were:

- Hyperparameters:
  1. max_len = 128
  2. batch_size = 16
  3. Mean pooling was chosen out of the three pooling strategies
  4. Learning Rate = 0.001
  5. Dropout Rate = 0.2

6 Error analysis

Let's take the following examples for error analysis:

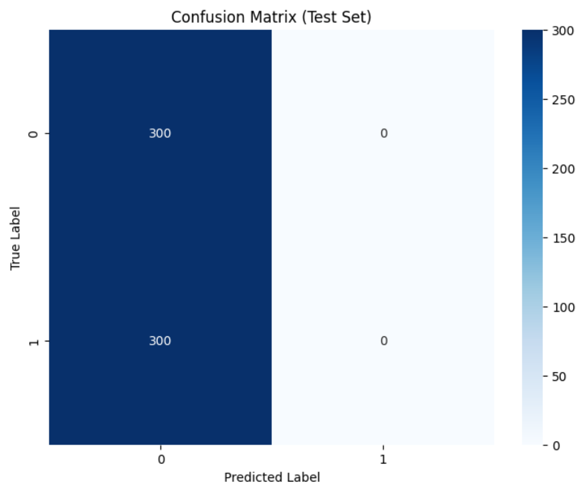

Figure 15: Confusion Matrix for Test Set - DistilBERT

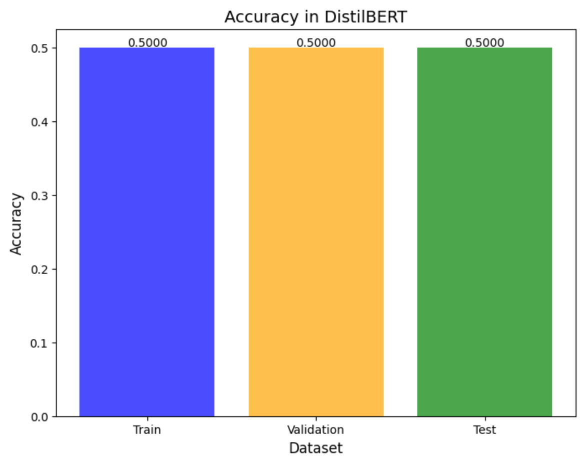

Figure 16: Accuracy of DistilBERT across Datasets

- Review 1: *An excellent wine with great balance and complexity. Rich flavors of dark fruit and oak*
- Review 2: *Amazing depth with perfect tannins. A masterpiece of winemaking*
- Review 3: *Simple and straightforward. Somewhat bland with little character*

1. The example predictions show that models struggle with nuanced wine reviews, particularly those with sophisticated language. For instance, Review 1 received mixed predictions across models:
    - Sequential Neural Network: Incorrectly predicted as "Not Liked"
    - Logistic Regression: Correctly predicted as "Liked"
    - Random Forest: Marginally predicted as "Liked"
    - DistilBERT: Incorrectly predicted as "Not Liked"
2. Reviews with seemingly positive descriptions can be misclassified, indicating the models' difficulty in capturing subtle sentiment nuances. Review 2 was often misclassified as "Not Liked" This suggests that the models might be over-relying on specific keywords rather than understanding the holistic sentiment. When trained on the smaller balanced dataset, Random Forest was able to classify it as 'Liked'.
3. Review 3 was more consistently classified as "Not Liked" across models, indicating better performance on more direct, negative descriptions.
4. The original dataset had a significant class imbalance (68% Not Liked, 32% Liked), which likely biased model predictions.

7 Conclusion

7.1 Project Takeaways

- Model Performance

  Logistic Regression emerged as the most consistent model while Random Forest showed robustness, especially with balanced datasets. Complex models like DistilBERT and Sequential Neural Network struggled with the nuanced wine review classification.

- Challenges Encountered

  Computational constraints limited running complex models on the full dataset. The binary sentiment labeling (90+ points) might be too simplistic for capturing wine review sentiments. Handling class imbalance was a significant challenge.

- Surprising Findings

  The performance variation across different dataset sizes (original vs. balanced) highlighted the importance of data preprocessing.

7.2 Future Prospect

- Refined Sentiment Labeling

  Develop a more granular sentiment scoring system beyond the binary 90-point threshold and incorporate more contextual features from wine reviews.

- Data Augmentation

  Synthetic reviews can be generated to balance the dataset. Author tried doing this through SMOTE but computational restraint is a huge challenge.

- Feature Engineering

  Incorporate domain-specific features relevant to wine reviews. Experiment with ensemble methods combining multiple model predictions.

- Final Thoughts

  This project demonstrated the complexity of sentiment analysis in a specialized domain like wine reviews. While machine learning models show promise, capturing the nuanced language of wine descriptions remains a challenging task that requires sophisticated natural language understanding techniques.

8 Acknowledgements

- Generative AI Tools were used to refine language for Error Analysis and Conclusion.

- Author only used the parts that improved the language with minor changes.

References

Devlin, J. et al. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Goodfellow, I. et al. (2016). *Deep learning*. MIT Press.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Pang, B. and Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting of the association for computational linguistics*, pages 271–278.

Panicheva, P. et al. (2018). Sentiment analysis in wine reviews using feature-based and lexicon-based approaches. *Journal of Food Science and Technology*.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Sun, C. et al. (2019). How to fine-tune bert for text classification? *arXiv preprint arXiv:1905.05583*.

Xu, W. et al. (2022). Bert for wine review sentiment and attribute analysis. *Computers and Food Studies*.