OSE Project: Essay

Name: Biswajit Palit
Matriculation Number: 50071214

The primary objective of my project is to perform 'Text Classification' on a uni-label, multi-class dataset using Natural Language Processing (NLP). I have employed various approaches, including logistic regression, decision trees, and naive Bayes, to evaluate the model's accuracy in predicting new and unseen data. The project's goal is to determine whether extensive fine-tuning of models results in better performance compared to feature extraction and scikit-learn models.

Dataset

I have utilized the PolyAI/banking77 dataset from the HuggingFace library, which comprises customer queries related to online banking and their corresponding intents. These queries are categorized into classes representing specific online banking issues such as 'card activation,' 'exchange rate,' and 'account balance issues.' The dataset contains 10,003 data points in the training set and 3,080 data points in the test set. I divided the test set in half to create three subsets: the training set, the test set (containing 1,540 data points), and the validation set (also containing 1,540 data points).

Reproducibility

Ensuring reproducibility was a significant challenge in this project. To address this, I established an environment at the beginning of the notebook, imported necessary dependencies, and set a random state. I also applied seed values to various sources of randomness, including CPU and GPU. Additionally, I assigned a seed to each hyperparameter tuning process to obtain consistent results. This approach resulted in perfect reproducibility for scikit-learn models. However, for transformer models, slight variations at the third decimal place could not be eliminated. Initially, I considered putting the hyperparameter tuning in the markdown for runtime efficiency, but encountered low initial accuracy scores, when the model is running for the first time in the notebook. To enhance reproducibility, I decided to run the hyperparameter tuning code in the notebook and then train the dataset with the best model. Notably, the best hyperparameters remained consistent across multiple notebook runs.

Approaches and Observations

A. Scikit-learn Models

At the outset, I performed feature extraction using the TF-IDF method and vectorized the dataset, resulting in X_train_tfidf, X_val_tfidf, and X_test_tfidf, which consist of numerical features based on the text data. This allowed me to explore different approaches.

1. **Logistic Regression**: This method was chosen because it suited the dataset's requirements. After performing a grid search, I found the best parameters for logistic regression (C=5, max_iter=500). This model achieved 88.37% accuracy on the validation set, which improved to 89% on the test set. It accurately predicted the class of custom text inputs.

2. **Decision Tree**: Using a grid search, I determined the best hyperparameters for the decision tree model. Despite being a simpler model than logistic regression, it achieved 75.32% accuracy on the validation set, improving to 75.71% on the test set. The predictions on custom text were consistent with the previous model.
3. **Naive Bayes**: I explored two approaches for Naive Bayes: multinomialNB and gaussianNB. Given the discrete vectorization of text queries in my dataset, multinomialNB was more relevant. The best parameters were alpha=0.1 and fit_prior=False, resulting in an accuracy and f1 score of 86.49 on the validation set, which improved to 87.20 on the test set. It provided consistent results on custom text inputs.

B. Transformer Models

Moving on to more sophisticated models, including BERT, RoBERTa, DistilBERT, and XLNet, I employed fine-tuning to potentially achieve better results. These models required separate tokenization of the data using AutoModelForSequenceClassification and AutoTokenizer.

1. **BERT**: On the validation set, the model achieved the best accuracy of 92.20% and an f1 score of 92.20%. Cross-checking with the test set improved the accuracy to 93.11%. BERT outperformed scikit-learn models and provided consistent predictions on custom text inputs.
2. **ROBERTA**: RoBERTa, a more robust adaptation of BERT, achieved a validation accuracy of 93.24% and an f1 score of 93.25%. On the test set, it improved to 93.79% accuracy, surpassing BERT. The model's performance on custom text remained consistent.
3. **DISTILBERT**: Despite being a lighter version of BERT, DistilBERT achieved a validation accuracy of 92% with optimal hyperparameters. It improved to 93.11% accuracy on the test set, maintaining consistency with other models on custom text inputs.
4. **XLNET**: XLNet, chosen for its different approach to capturing bidirectionality in text, achieved an accuracy of 91.9% with the best hyperparameters. Although not as good as BERT, it reached 93% accuracy on the test set and provided consistent results on custom text inputs.

Custom Text and Results

For a deeper understanding of how the models perform on real-world examples, I tested them with custom text inputs. One such example is:

Custom Text: “I have returned an item at ZARA, they say my refund is made, my bank account does not show it.”

Here are the results from various models:

1. Logistic Regression: Predicted Result - Refund_not_showing_up
2. Decision Tree: Predicted Result - Refund_not_showing_up
3. Naive Bayes: Predicted Result - Refund_not_showing_up
4. BERT: Predicted Result - Refund_not_showing_up
5. RoBERTa: Predicted Result - Refund_not_showing_up
6. DistilBERT: Predicted Result - Refund_not_showing_up
7. XLNet: Predicted Result - Refund_not_showing_up

It's worth noting that all models, including both scikit-learn and transformer models, provided consistent predictions for this custom text input. This might suggest that the text I chose for testing contains keywords or patterns that are easily identifiable by the models.

In previous runs of the notebook, I tested another custom text input: "I have made a transfer of 300 euros but it is not showing in my bank account transaction history." The result for this text was consistently predicted as "balance_not_updated_after_bank_transfer" across all models.

Conclusion

Among the transformer models, RoBERTa yielded the highest accuracy score at 93.79%, demonstrating a significant improvement over the best scikit-learn model, Logistic Regression, which achieved 89%. This supports the idea that transformer models offer greater depth and sensitivity, capturing subtle nuances in the data more effectively. However, the choice of custom text failed to reveal distinct performance differences among the models. Other custom text inputs also resulted in consistent predictions across models.