INDIVIDUAL PROJECT REPORT
ANALYZING THE SENTIMENT OF FINANCIAL DATA

Ketan Kapse - Group 6

Introduction

For this project, we are working on the financial sentiments dataset and developing models to predict the sentiments of financial textual data.

My contribution to this project lie in the following:

- Preprocessing the dataset.
- Implementation of the DistilBert model.
- Implementation of the roBERTa model.
- Implementation of the GBC model.
- Helping set up streamlit to host our models.

Description of Individual Work

For the GBC model, the workflow of my code is as follows:

Certainly! Here's the workflow presented in the past tense:

1. **Imported Necessary Libraries:** I started by importing the required libraries, especially Scikit-learn (sklearn), which provided the tools needed for machine learning tasks.
2. **Prepared the Data:** I loaded and organized my dataset into features (X) and corresponding target labels (y). Ensured that I split the dataset into training and testing subsets for model evaluation. I used TF-IDF to vectorize the data.

3. **Instantiated the Gradient Boosting Classifier:** I created an instance of the GradientBoostingClassifier from Scikit-learn's ensemble module. During initialization, I set hyperparameters such as:
   - n_estimators: 300.
   - learning_rate: 0.1
   - max_depth: 10
4. **Trained the Model:** I trained the Gradient Boosting Classifier by fitting it to the training data (X_train and y_train) using the .fit() method. This step allowed the model to learn and understand patterns present in the training data.
5. **Evaluated the Model:** I evaluated the trained model's performance using the testing dataset (X_test and y_test). Metrics such as accuracy, precision, recall, F1-score, or confusion matrix were employed to assess the model's generalization to unseen data.

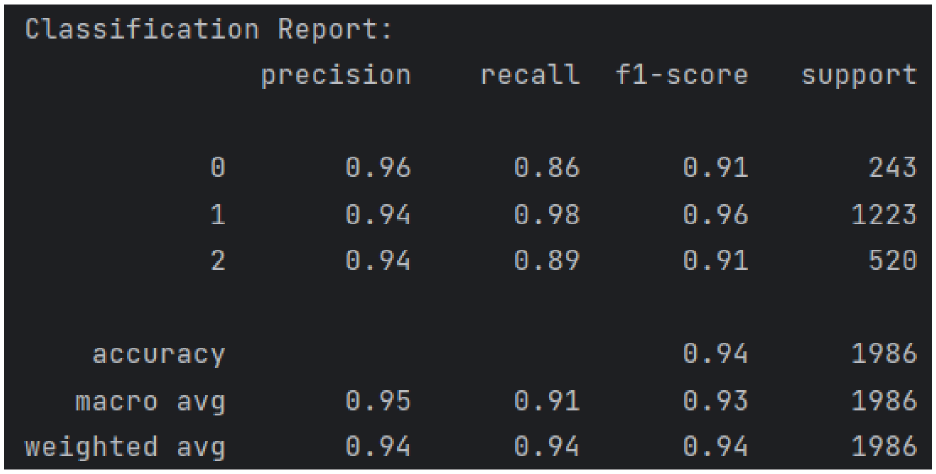
For the DistilBert and roBERTa models my workflow was as follows:

1. **Data Loading and Preprocessing:** I loaded the data from a CSV file named 'combinedData.csv' into a Pandas DataFrame. I converted the 'sentence' column to strings and applied a preprocessing function preprocess_df to clean and preprocess the text. After that, I selected the 'label' and 'sentence' columns for further processing.
2. **Data Splitting:** I split the dataset into training and evaluation sets using the train_test_split function from Scikit-learn. The splitting was done with 80% for training and 20% for holdout. I used stratification based on labels to maintain the class distribution in both sets.
3. **Model Initialization:** I utilized the Roberta tokenizer (RobertaTokenizer) and a pre-trained Roberta model for sequence classification (RobertaForSequenceClassification) for the roberta model. I used the DistilbertTokenizer and the pre-trained distilbert model for sequence classification(DistilbertForSequenceClassification).
4. **Model Training with K-Fold Cross-Validation:** To train the model robustly, I employed K-Fold Cross-Validation (StratifiedKFold) with 3 folds. For each fold:

    - I split the training set further into train and validation subsets.
    - Encoded the text data using the Roberta/Distilbert tokenizer and created PyTorch DataLoader objects for training and validation.
    - Trained the model for a specified number of epochs while monitoring training and validation loss and accuracy.
5. **Evaluation on Holdout Set and Metrics Calculation:** After completing K-Fold Cross-Validation, I evaluated the model on the holdout set (eval_sentences and eval_labels). I calculated the accuracy on this unseen dataset and calculated the F1-score for further performance assessment. Additionally, I generated a classification report to gain insights into model performance per class.
6. **Visualization of Training and Validation Metrics:** Finally, I plotted graphs to visualize the training and validation accuracy alongside the training and validation loss over epochs.

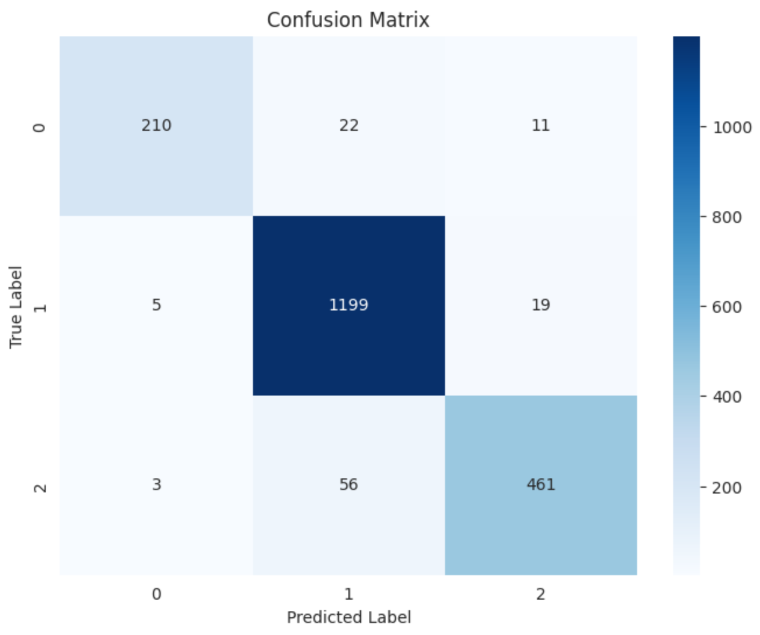
Results

<u>GBC:</u>



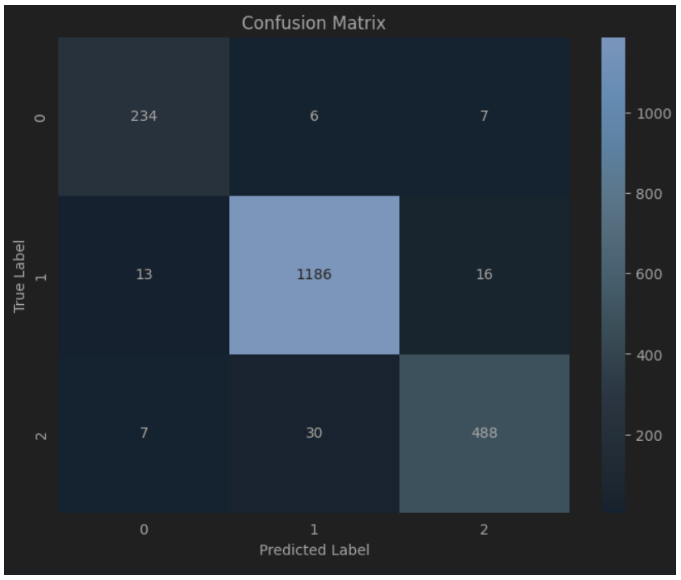
```
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.86      0.91       243
           1       0.94      0.98      0.96      1223
           2       0.94      0.89      0.91       520

    accuracy                           0.94      1986
   macro avg       0.95      0.91      0.93      1986
weighted avg       0.94      0.94      0.94      1986
```

Classification Report for GBC on the test set.

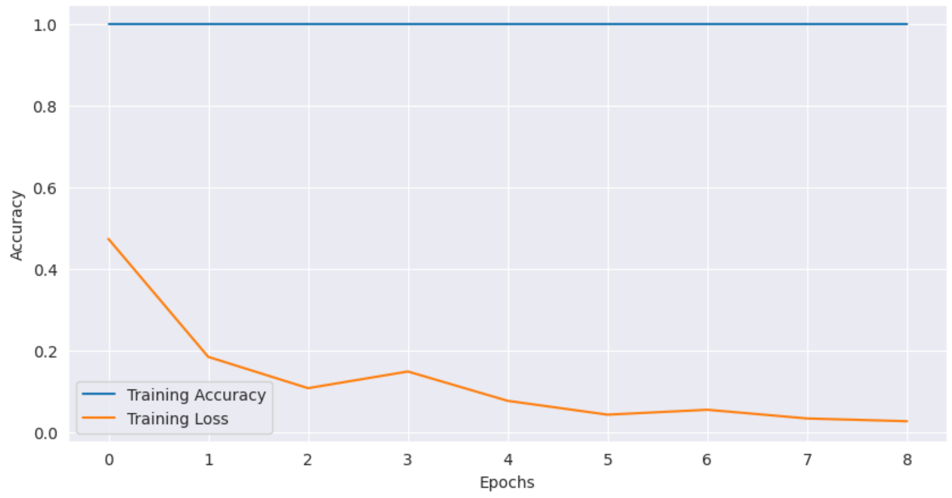


Confusion Matrix for GBC on the test set

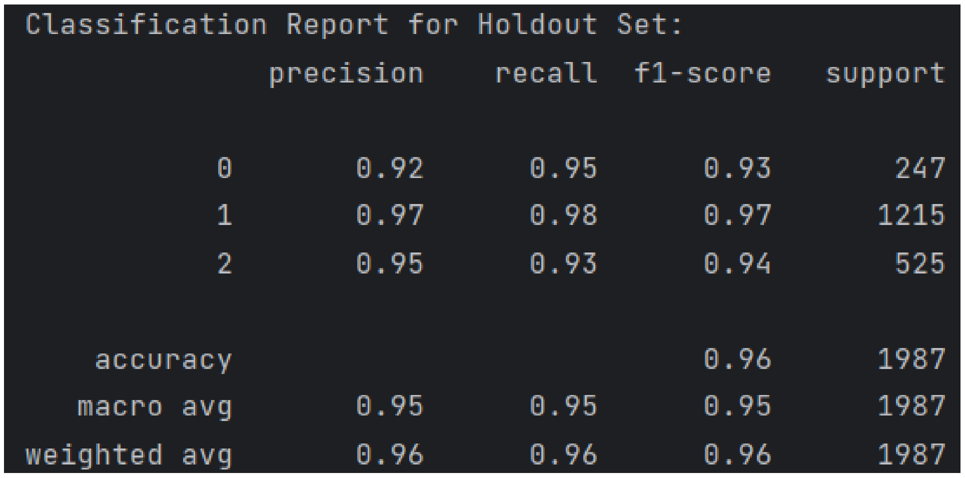
DistilBert:



Confusion Matrix for DistilBert on the holdout set.



Training Accuracy and Loss for Distilbert



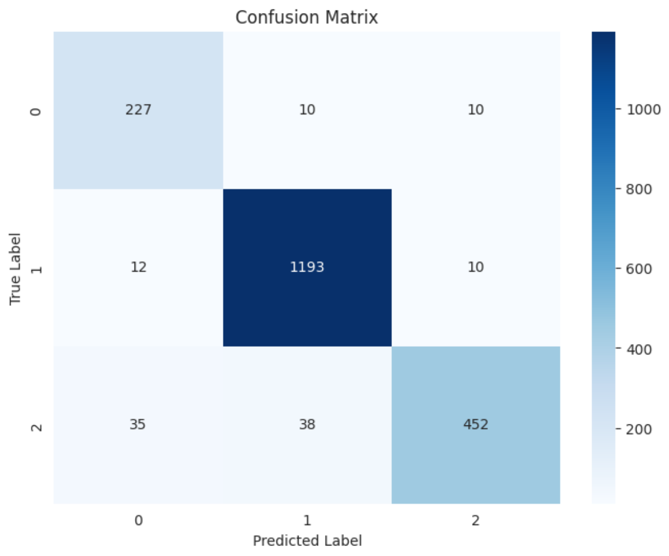
```
Classification Report for Holdout Set:
              precision    recall  f1-score   support

           0       0.92      0.95      0.93       247
           1       0.97      0.98      0.97      1215
           2       0.95      0.93      0.94       525

    accuracy                           0.96      1987
   macro avg       0.95      0.95      0.95      1987
weighted avg       0.96      0.96      0.96      1987
```

Classification report for distilbert on the holdout set.

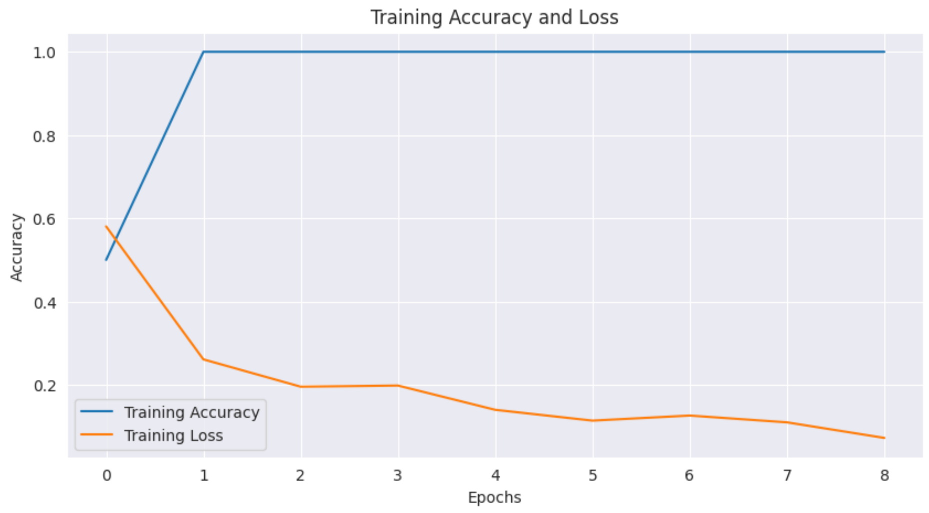
The distilbert model performs amazingly with F1 scores being 0.96 on the holdout set.

The learning rate of the model was 2e-5, loss was crossentropy and optimizer was AdamW with betas (0.9, 0.999) and epsilon 1e-5.

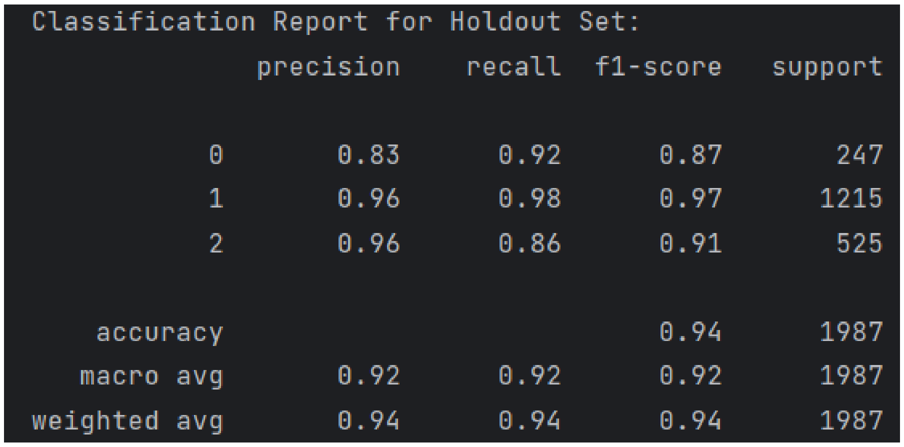
roBERTA:



Confusion matrix on the holdout set



Training Accuracy and Loss


```
Classification Report for Holdout Set:
              precision    recall  f1-score   support

           0       0.83      0.92      0.87       247
           1       0.96      0.98      0.97      1215
           2       0.96      0.86      0.91       525

    accuracy                           0.94      1987
   macro avg       0.92      0.92      0.92      1987
weighted avg       0.94      0.94      0.94      1987
```

Classification report on the holdout set.

The learning rate of the model was 2e-5, loss was crossentropy and optimizer was AdamW with betas (0.9, 0.999) and epsilon 1e-5.

It performs similarly to distilbert.

Streamlit:

In this part of our project, I contributed to loading up the models and creating the input fields.

Summary and Conclusion

In summary, I implemented the GBC, DistilBert and Roberta models. They all have F1 scores above 90 indicating that they are performing very well.
For future work, I would consider implementing textual augmentations such as back translations to address the imbalance in the class data.

Percentage of Original Code

For this project, we wrote the code on our own wherever we could. However, we took the help of online resources and AI assisted tools in Google Colab and ChatGPT when we required assistance.