Sentiment Analysis Project Documentation

By Khadeeja Arshad Ali (B20102057)

1) Problem Statement

Objective: The goal of this project is to develop a model that accurately classifies text as either positive or negative sentiment. This is particularly useful for applications like product reviews, social media sentiment analysis, and customer feedback, where understanding users' sentiments can provide valuable insights. The challenge is to capture nuances in language, sarcasm, and informal expressions that may affect sentiment classification accuracy.

2) Dataset Description

- **Dataset Source**: https://www.kaggle.com/datasets/endofnight17j03/bert-sentiment-analysis
- **Dataset Overview**: The dataset consists of text samples labeled as either positive or negative sentiment. Each sample is a short sentence or review with one of two labels:
  - **Label 1**: Positive
  - **Label 0**: Negative
- **Data Size**: 5668 (70% for training and 30% for testing)
- **Sample Data:**

Sentence	Label
Ok brokeback mountain is such a horrible movie.	0
Brokeback Mountain was so awesome.	1
friday hung out with kelsie and we went and saw The Da Vinci Code SUCKED!!!!!	0
I am going to start reading the Harry Potter series again because that is one awesome story.	1

3) Method

1. BERT for Sequence Classification

In our code, we used **BERT for Sequence Classification** to classify sentences as positive or negative. Here's a summary of the process:

- **Tokenization**: The BERT tokenizer splits the text into tokens and maps each token to a numerical ID from BERT's vocabulary. This step allows BERT to interpret the sentence in a structured way that it can process.
- **Model Initialization and Fine-Tuning**: The ***`BertForSequenceClassification`*** model is initialized with two output labels (for binary classification) and fine-tuned on your specific dataset. Fine-tuning BERT involves training only a few layers on top of pre-trained weights, allowing the model to adapt to your task without extensive retraining.
- **Prediction**: After training, the model predicts the sentiment for new text samples by outputting probabilities for each class. The class with the highest probability is chosen as the predicted sentiment.

2. Google Gemini Model (Gemini Pro)

In our code, we used the **Gemini Pro model for few-shot learning** to predict sentiment. Here's a summary of the approach:

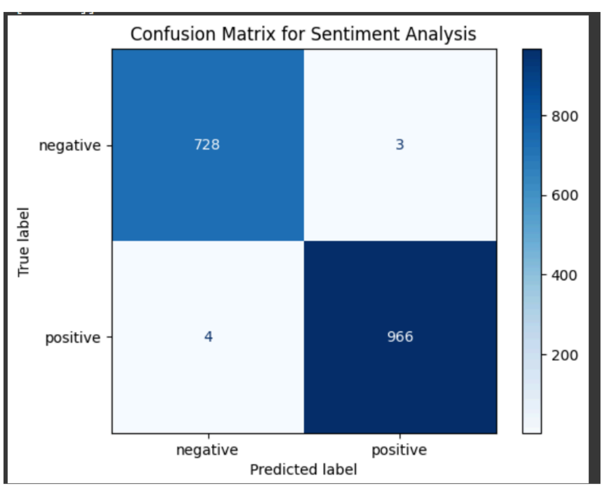
- **Few-Shot Learning**: Few-shot learning is a technique where a language model is provided with a small number of example inputs along with their labels (positive or negative in this case). This prompt setup gives the model context for the task to predict the label of a new, unseen input. Few-shot learning is beneficial when limited labeled data or a quick model setup is needed.
- **Prompt Creation**: The code prepares a custom prompt with a few example sentences labeled as positive or negative. This helps guide the model's understanding of how to classify similar sentences.
- **Inference**: Once the prompt is set up, the model generates a response to classify the input sentence. It can understand and mimic patterns from the examples provided, making it versatile for tasks where deep contextual knowledge or flexibility is required.

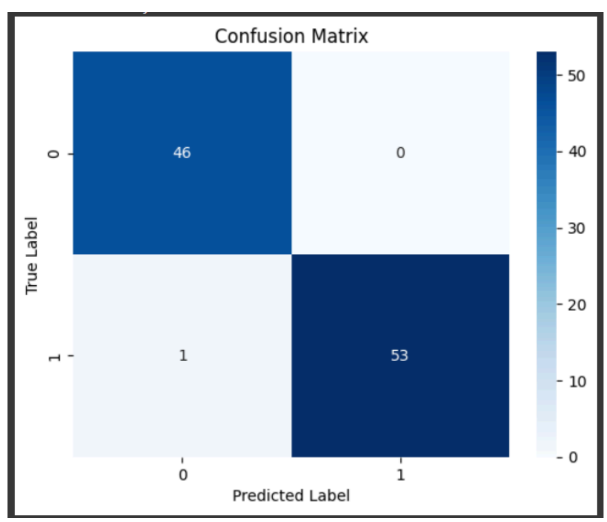
4) Results

Evaluation Metrics

To assess model performance, the following metrics were used:

- **Accuracy**: Measures the proportion of correct predictions among the total number of predictions.
- **Precision**: Indicates how many of the positive predictions were actually positive.
- **Recall**: Shows how many of the actual positive samples were correctly identified by the model.
- **F1-Score**: The harmonic mean of precision and recall, providing a balance between the two.

<table>
<tr><th>BERT</th><th>LLM</th></tr>
<tr><td>Accuracy</td><td>Accuracy</td></tr>
<tr><td>Accuracy: 0.9959</td><td>Model Accuracy: 99.00%</td></tr>
<tr><td>Classification Report</td><td>Classification Report</td></tr>
<tr><td>
<table>
<tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr>
<tr><td>negative</td><td>0.99</td><td>1.00</td><td>1.00</td><td>731</td></tr>
<tr><td>positive</td><td>1.00</td><td>1.00</td><td>1.00</td><td>970</td></tr>
<tr><td>accuracy</td><td></td><td></td><td>1.00</td><td>1701</td></tr>
<tr><td>macro avg</td><td>1.00</td><td>1.00</td><td>1.00</td><td>1701</td></tr>
<tr><td>weighted avg</td><td>1.00</td><td>1.00</td><td>1.00</td><td>1701</td></tr>
</table>
</td><td>
Classification Report:
<table>
<tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr>
<tr><td>0</td><td>0.98</td><td>1.00</td><td>0.99</td><td>46</td></tr>
<tr><td>1</td><td>1.00</td><td>0.98</td><td>0.99</td><td>54</td></tr>
<tr><td>accuracy</td><td></td><td></td><td>0.99</td><td>100</td></tr>
<tr><td>macro avg</td><td>0.99</td><td>0.99</td><td>0.99</td><td>100</td></tr>
<tr><td>weighted avg</td><td>0.99</td><td>0.99</td><td>0.99</td><td>100</td></tr>
</table>
</td></tr>
<tr><td>Confusion Matrix</td><td>Confusion Matrix</td></tr>
<tr><td>


</td><td>


</td></tr>
</table>

5) Discussion & Error Analysis

BERT

1. **Sentence**: "st nd harry potter movies clearly best funniest"
   - **True Label**: 1 (Positive)
   - **Predicted Label**: 0 (Negative)
   - **Error Type**:
     - **Contextual Misunderstanding Errors**: The sentence is brief and lacks a clear positive signal; words like "best" and "funniest" may not appear as strong indicators of positivity without surrounding context, which could lead the model to misclassify it as neutral or negative.

2. **Sentence**: "sick books right now must say da vinci code awesome"
   - **True Label**: 1 (Positive)
   - **Predicted Label**: 0 (Negative)
   - **Error Type**:
     - **Synonym and Polysemy Errors**: The word “sick” is polysemous—it could mean “cool” (positive) or “ill” (negative). The model might misinterpret this term if it hasn't been seen it used positively during training, leading to an incorrect classification.
     - **Contextual Misunderstanding Errors**: The informal phrasing (e.g., "sick books right now") may not be understood as a positive sentiment due to the model’s limited understanding of conversational language structure. Without a clear indicator of sentiment, BERT may treat this as ambiguous.

3. **Sentence**: "personally neither hate love da vinci code"
   - **True Label**: 1 (Positive)
   - **Predicted Label**: 0 (Negative)
   - **Error Type**:
     - **Label Ambiguity Errors**: The phrase “neither hate love” creates a neutral or ambiguous tone that could be confusing for the model. The wording suggests a lack of strong emotion, which may cause the model to lean towards a neutral or negative prediction rather than a positive one.
     - **Noise and Labeling Errors**: The sentence’s neutrality could lead to labeling challenges, as it doesn’t express a clear sentiment either way. If such phrases were inconsistently labeled in the training data, the model might struggle to interpret them accurately.

4. **Sentence**: "harry potter dumb"
    - **True Label**: 0 (Negative)
    - **Predicted Label**: 1 (Positive)
    - **Error Type**:
        - **Length and Complexity Errors**: This sentence is too short and contains little information, which may dilute the sentiment.
        - **Class Imbalance Errors**: If positive examples are overrepresented in the dataset, the model may have a tendency to predict “positive” more often. Simple negative phrases like “dumb” may be overshadowed, and the model could classify such examples as positive due to bias from the class imbalance.

5. **Sentence**: "da vinci code sucked ballz"
    - **True Label**: 0 (Negative)
    - **Predicted Label**: 1 (Positive)
    - **Error Type**:
        - **Noise and Labeling Errors**: The informal slang “ballz” might be unfamiliar to the model, which could lead it to ignore this part of the sentence. Slang can introduce noise that confuses the model’s understanding, leading it to overlook negative sentiment.
        - **Out-of-Distribution Errors**: The model may not have encountered similar slang or informal expressions in its training data. Unfamiliar phrases like “sucked ballz” could be incorrectly classified due to their unusual structure, which the model interprets as positive by default.

6. **Sentence**: "ah nothin love heather shes great hate will brokeback mountain stites"
    - **True Label**: 0 (Negative)
    - **Predicted Label**: 1 (Positive)
    - **Error Type**:
        - **Contextual Misunderstanding Errors**: The sentence expresses mixed sentiments (“love” and “hate”), which could confuse the model. Without a clear indicator that the sentiment is negative overall, the model may lean toward a positive classification due to the mention of “love.”
        - **Label Ambiguity Errors**: The presence of both positive and negative expressions creates ambiguity. Such mixed sentiments may be inconsistently labeled, making it difficult for the model to generalize accurately.

7. **Sentence**: "its hard decide movie to see good ones playing would love see da vinci code hand over the hedge looks funny"
    - **True Label**: 1 (Positive)
    - **Predicted Label**: 0 (Negative)
    - **Error Type**:
        - **Length and Complexity Errors**: This sentence is long and contains multiple topics, which may dilute the sentiment. Longer sentences can present too much information for the model to process, leading it to miss the main positive sentiment.
        - **Label Ambiguity Errors**: The sentence has mixed signals with both positive (e.g., "love") and neutral expressions (e.g., "looks funny"), which may make it difficult for the model to assign a positive label confidently.

LLM

Sentence: "He's like, 'YEAH I GOT ACNE AND I LOVE BROKEBACK MOUNTAIN'"
True Label: 1 (Positive)
Predicted Label: 0 (Negative)
Error Type:

- **Contextual Misunderstanding Errors**: The capitalized "LOVE" may be interpreted as sarcasm or exaggeration, which could cause the model to misinterpret the sentiment. The informal tone may not align with typical positive examples seen by the model.
- **Noise and Labeling Errors**: The unusual structure and mixed capitalization introduce noise that may mislead the model. If the training data lacked similar formats, the model might not handle this input properly.
- **Label Ambiguity Errors**: The sentence can appear ambiguous since sarcasm could be read as either positive or negative. Without clear context, the model may find it difficult to resolve the true sentiment.

Summary of Error Types in Misclassified Items

Sentence	True Labels	Predicted Label	Error Types
"st nd harry potter movies clearly best funniest"	1	0	Label Ambiguity Errors, Contextual Misunderstanding
"sick books right now must say da vinci code awesome"	1	0	Synonym and Polysemy, Contextual Misunderstanding
"personally neither hate love da vinci code"	1	0	Noise and Labeling Errors, Label Ambiguity Errors
"harry potter dumb"	0	1	Class Imbalance Errors
"da vinci code sucked ballz"	0	1	Noise and Labeling, Out-of-Distribution
"ah nothin love heather shes great hate will brokeback mountain stites"	0	1	Contextual Misunderstanding, Label Ambiguity
"its hard decide movie to see good ones playing would love see da vinci code hand over the hedge looks funny"	1	0	Length and Complexity, Label Ambiguity
"He's like, 'YEAH I GOT ACNE AND I LOVE BROKEBACK MOUNTAIN'"	1	0	Contextual Misunderstanding, Label Ambiguity Errors