# Project 2

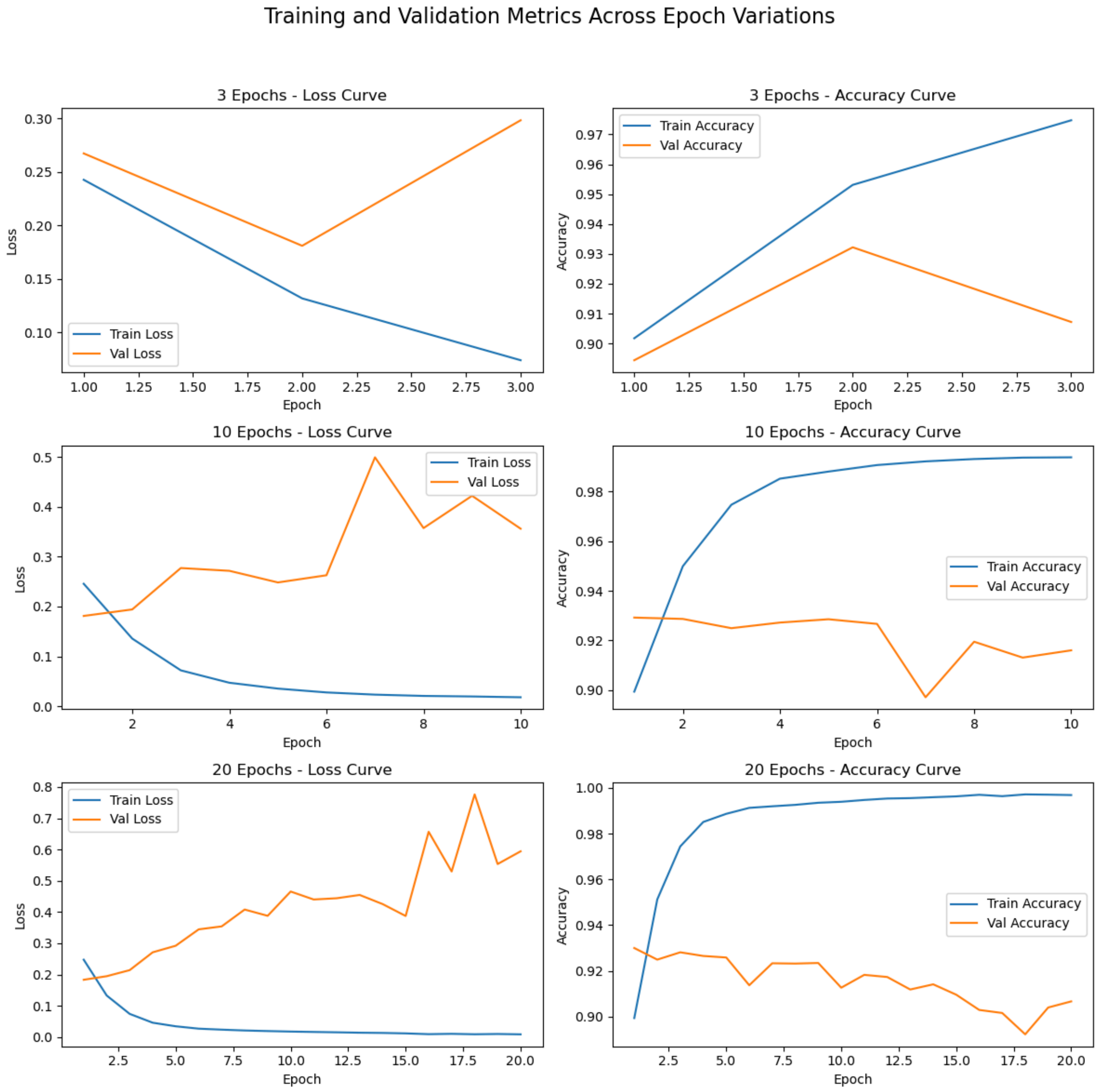
# Experiments:

| Models | Epochs | Model names | Runtime |
|---|---|---|---|
| DistilBERT fine-tuned | 3, 10, 20 | {‘model’: “distilbert-base-uncased”, ‘tokenizer’: “distilbert-base-uncased”} | 3 epochs: ~25 min<br>10 epochs: ~1 hr 15 min<br>20 epochs: ~2hr 25 min |
| GPT2 | 10, 20 | {‘model’: “gpt2”, ‘tokenizer’: “gpt2”} | 10 epochs: ~2 hr 16 min<br>20 epochs: ~4hr 25 min |
| DistilBERT vanilla | N/A | {‘model’: “distilbert-base-uncased”, ‘tokenizer’: “distilbert-base-uncased”} | Testing time on 15% of the data: ~3 minutes |
| Logistic Regression | N/A | sklearn.linear_mode.LogisticRegression<br>CountVectorizer (Bag of Words) | Testing time on 30% of the data: ~1 minute |

1. What do the accuracy and loss curves tell you about the fine-tuning process?

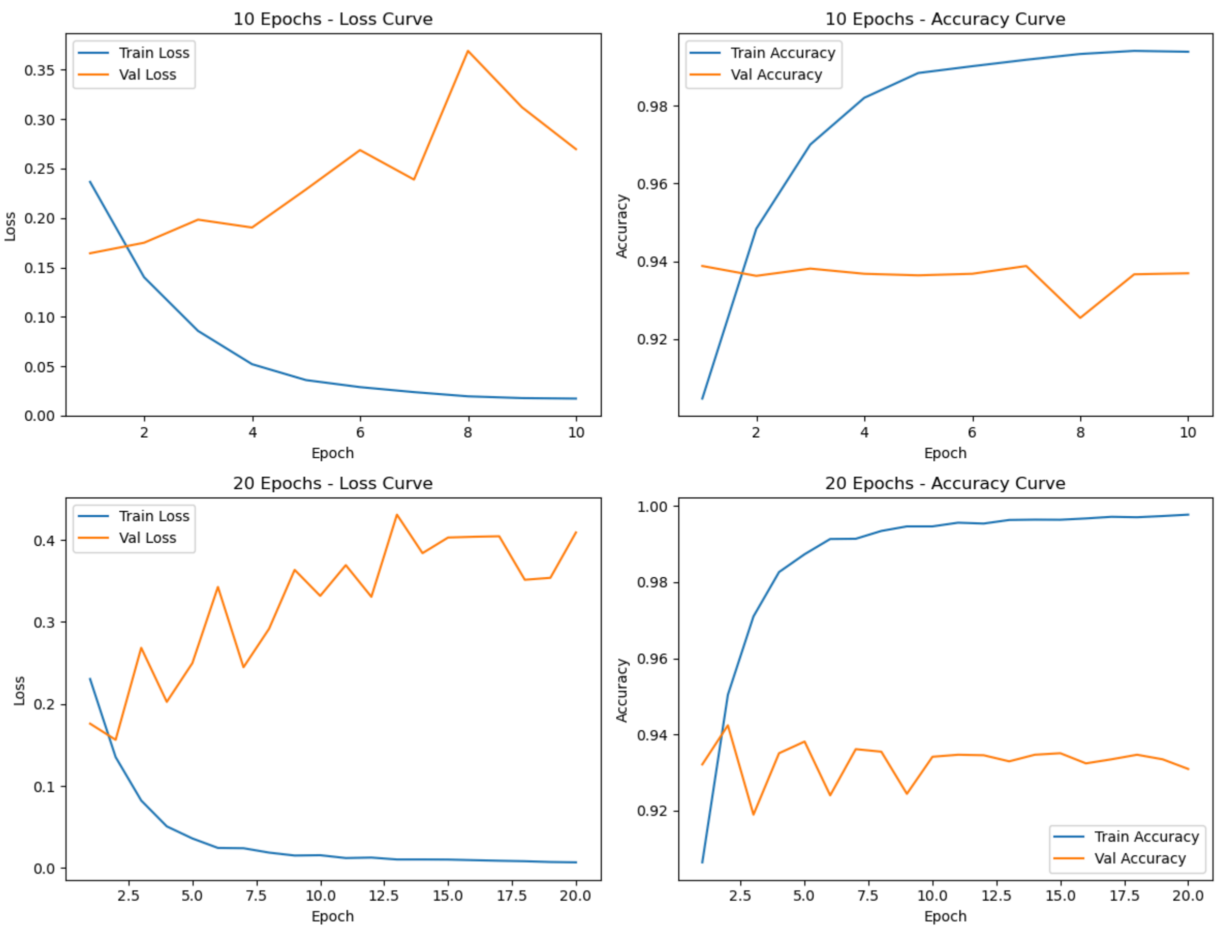
The training and validation results indicate that the model consistently overfits as the number of epochs increases. In the case of 3 epochs, the model begins to generalize initially, but validation loss increases, and accuracy drops after the first epoch, signaling the onset of overfitting. For 10 and 20 epochs, while training loss approaches zero and accuracy nears 100%, validation loss rises steadily, and validation accuracy stagnates or declines, further highlighting overfitting. These results suggest that training for too many epochs causes the model to memorize the training data rather than generalize to unseen validation data.

Training & validation accuracy and loss over epochs for the fine-tuned DistilBERT model.



Comparatively, GPT2's training & validation accuracy and loss are presented below:

Training and Validation Metrics Across Epoch Variants for GPT-2



2. How does the fine-tuned DistilBERT model compare to the classical ML model? What advantages or limitations do transformers present over classical algorithms?

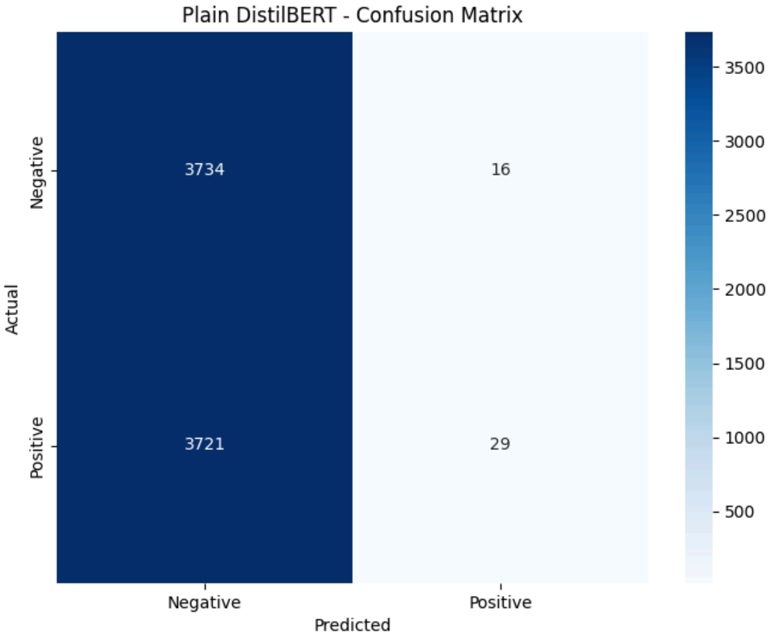
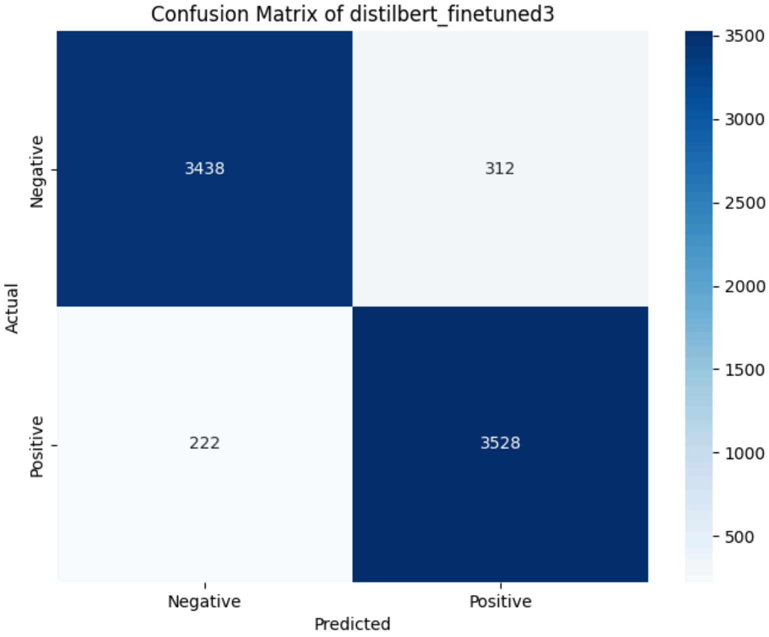
The fine-tuned DistilBERT model generally outperforms the classical machine learning model in terms of accuracy, precision, recall, and F1-score, especially when handling complex text data like sentiment analysis. This is because transformers like DistilBERT leverage pre-trained embeddings, attention mechanisms, and large-scale language understanding to capture nuanced relationships within the text, in this case sentiment. Classical machine learning models, like logistic regression, rely heavily on feature engineering and simpler representations like Bag-of-Words(which was used here), which often fail to capture deeper semantic or contextual information.

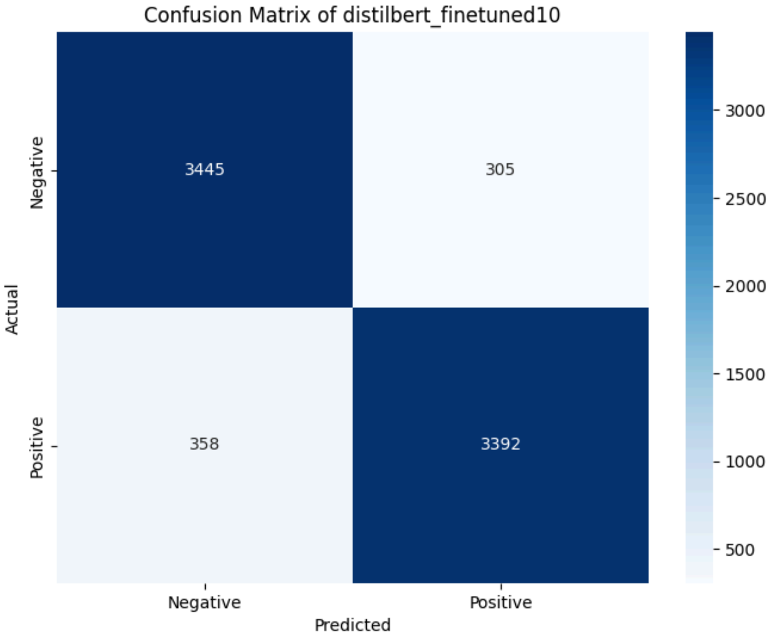
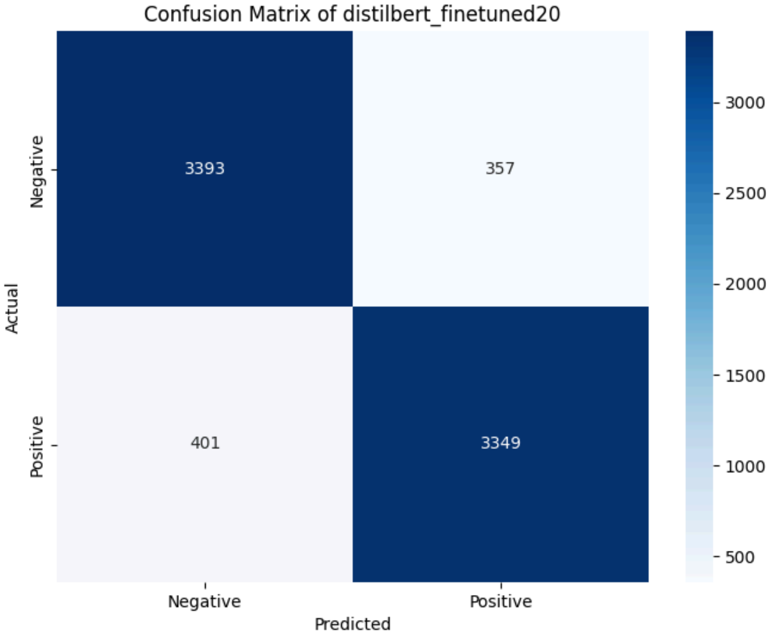
Transformers have the advantage of being highly scalable and capable of learning from large, pre-trained datasets, making them adaptable to a wide variety of tasks without extensive feature engineering. However, they come with notable limitations: they require significant computational resources (e.g., GPUs or TPUs) and large amounts of labeled data for fine-tuning, making them less accessible for smaller-scale projects. Classical algorithms, on the other hand, are lightweight, computationally efficient, and perform well on smaller datasets, making them more practical for resource-constrained environments. Ultimately, the choice depends on the task's complexity, dataset size, and available resources.

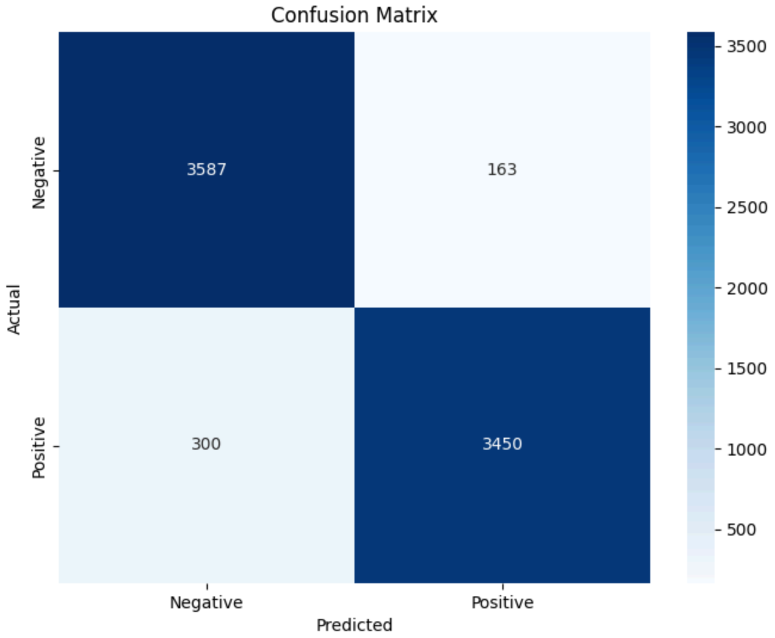
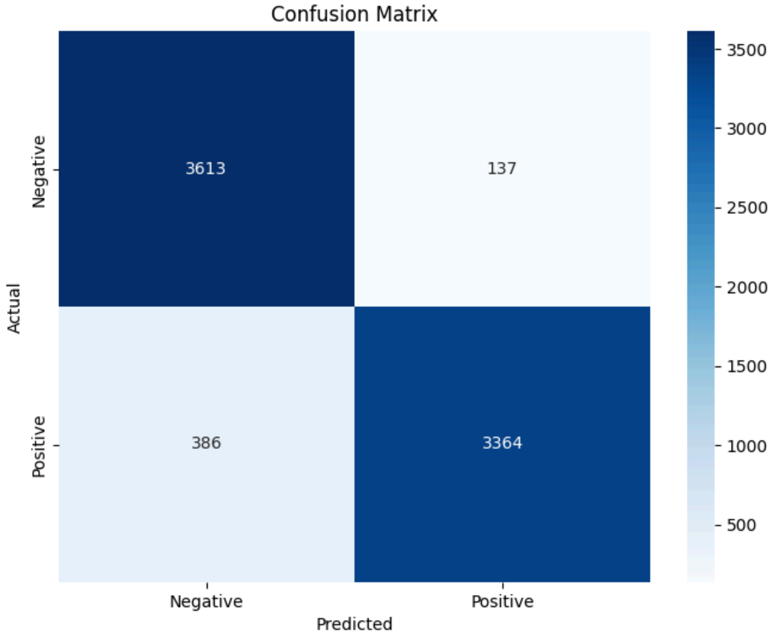
3. What insights can you draw from the confusion matrix? Are there any patterns in the misclassifications?

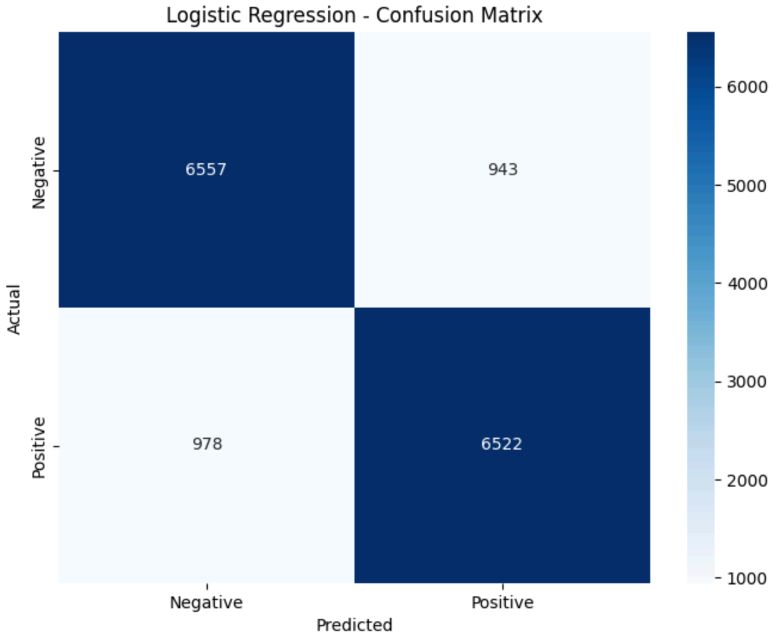
DistilBERT without fine-tuning gave 50% accuracy results whereas the fine-tuned model reached 92% accuracy. DistilBERT (without fine-tuning) tried to predict every sentiment/review as negative and wound up with 50% accuracy. Comparatively, GPT2's validation confusion matrices for 10 and 20 epochs had a slightly higher number of correct predictions than the fine-tuned DistilBERT model's 10 and 20 epoch variants. I reckon GPT2 without train will have a higher yield than vanilla DistilBERT. I wasn't able to save the fine-tuned GPT2 model and then ran it on the test dataset which I was doing for the fine-tuned DistilBERT model. That's because HuggingFace's model saving is tricky and you can't just simply load the model, especially if the model is an old one. I tried debugging but there wasn't enough time.
Below are the confusion matrices of validation results. The first one is for 10 epochs of fine-tuning and the next one is 20 epochs of fine-tuning.

| Model Names | Epochs | Confusion Matrix |
|---|---|---|

| | | |
|---|---|---|
| **DistilBERT (without fine-tuning)** | **N/A** |  |
| **DistilBERT (fine-tuned)** | **3** |  |

| | | |
|---|---|---|
| | **10** | Confusion Matrix of distilbert_finetuned10<br>3445<br>305<br>358<br>3392<br>Actual<br>Negative<br>Positive<br>Predicted<br>Negative<br>Positive<br>3000<br>2500<br>2000<br>1500<br>1000<br>500 |
| | **20** | Confusion Matrix of distilbert_finetuned20<br>3393<br>357<br>401<br>3349<br>Actual<br>Negative<br>Positive<br>Predicted<br>Negative<br>Positive<br>3000<br>2500<br>2000<br>1500<br>1000<br>500 |

| | | |
|---|---|---|
| **GPT2** | **10** |  Confusion Matrix<br>Negative<br>Positive<br>Actual<br>3587<br>163<br>300<br>3450<br>Negative<br>Positive<br>Predicted<br>3500<br>3000<br>2500<br>2000<br>1500<br>1000<br>500 |
| | **20** |  Confusion Matrix<br>Negative<br>Positive<br>Actual<br>3613<br>137<br>386<br>3364<br>Negative<br>Positive<br>Predicted<br>3500<br>3000<br>2500<br>2000<br>1500<br>1000<br>500 |

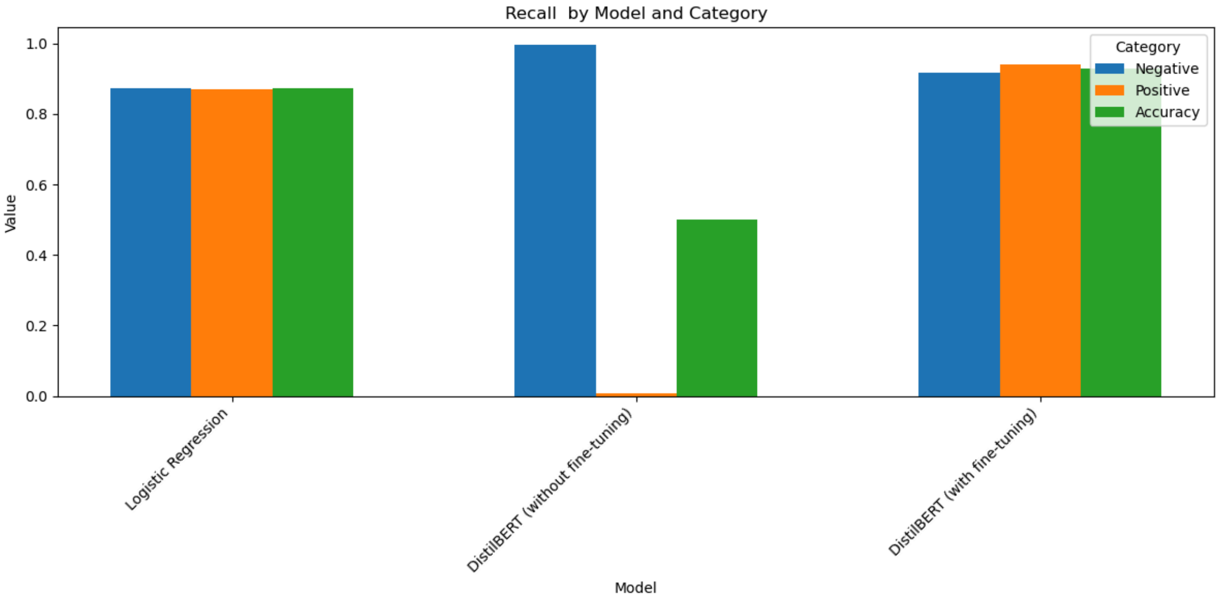
| | | |
|---|---|---|
| **Logistic Regression** | **N/A** |  |

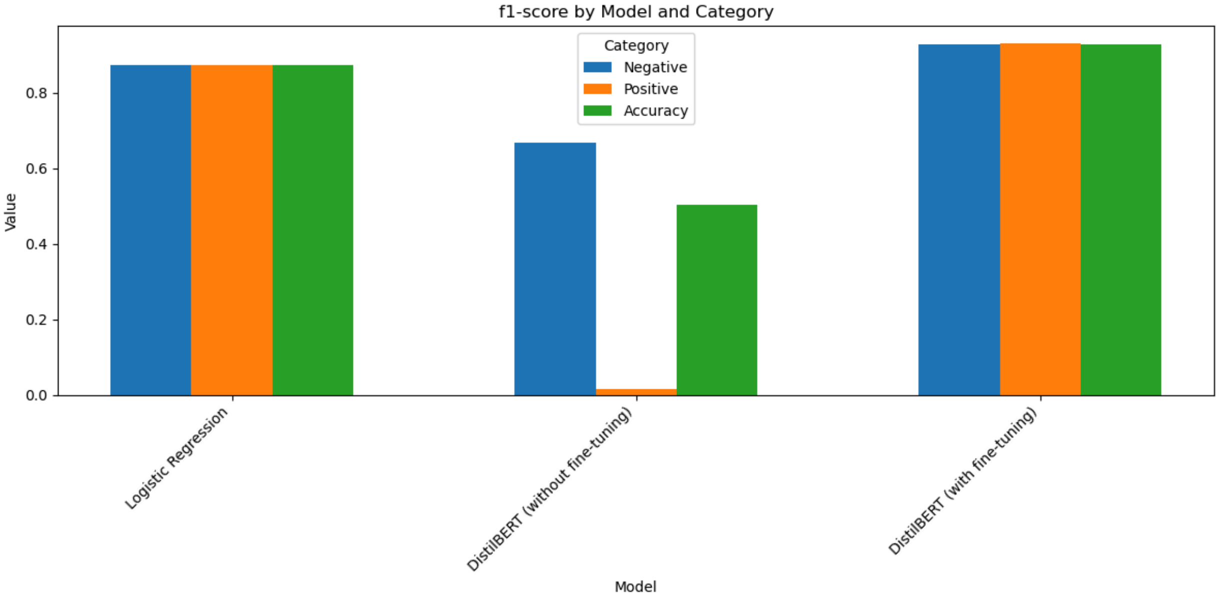
4. Why might the fine-tuned model outperform the base model?

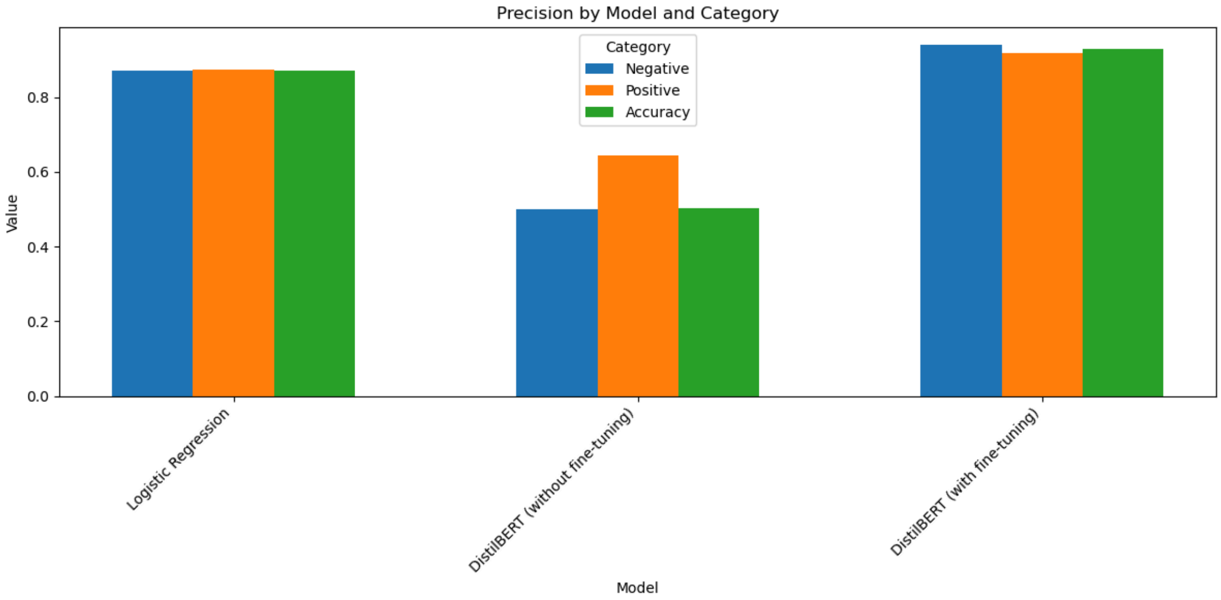
The fine-tuned DistilBERT model significantly outperforms the base DistilBERT model across all metrics because fine-tuning allows the pre-trained transformer to adapt to the specific nuances of the task, in this case, sentiment analysis. The base model performs poorly, achieving near-random classification results (around 50% accuracy), as it lacks task-specific optimization and relies solely on general language pretraining. Fine-tuning refines the model's weights on labeled data, enabling it to learn task-specific patterns such as sentiment polarity, resulting in substantial improvements in precision, recall, and F1-score. For instance, the fine-tuned model achieves an accuracy of 92.88% with 3 epochs, compared to 50.17% for the base model. However, as training continues to 20 epochs, performance slightly declines due to potential overfitting, highlighting the importance of balancing fine-tuning to retain generalization capabilities.

| Models | Epoch | Categories | Precision | Recall | f1-score | Support |
|---|---|---|---|---|---|---|
| Logistic Regression | N/A | Negative | 0.870205706 | 0.874266666 | 0.872231459 | 7500 |
| | | Positive | 0.873677160 | 0.869600000 | 0.871633812 | 7500 |
| | | Accuracy | 0.871933333 | 0.871933333 | 0.871933333 | 0.871 |
| | | Macro avg | 0.871941433 | 0.871933333 | 0.871932636 | 15000 |
| | | Weighted avg | 0.871941433 | 0.871933333 | 0.871932636 | 15000 |
| DistilBERT (without fine-tuning) | N/A | Negative | 0.500871898 | 0.995733333 | 0.666488175 | 3750 |
| | | Positive | 0.644444444 | 0.007733333 | 0.015283267 | 3750 |
| | | Accuracy | 0.501733333 | 0.501733333 | 0.501733333 | 0.501 |
| | | Macro avg | 0.572658171 | 0.501733333 | 0.340885721 | 7500 |
| | | Weighted avg | 0.572658171 | 0.501733333 | 0.340885721 | 7500 |
| DistilBERT (with fine-tuning) | 3 | Negative | 0.939344262 | 0.9168 | 0.927935223 | 3750 |
| | | Positive | 0.91875 | 0.9408 | 0.929644269 | 3750 |
| | | Accuracy | 0.9288 | 0.9288 | 0.9288 | 0.928 |
| | | Macro avg | 0.929047131 | 0.9288 | 0.928789746 | 7500 |
| | | Weighted avg | 0.929047131 | 0.9288 | 0.928789746 | 7500 |
| | 10 | Negative | 0.905863792 | 0.918666667 | 0.91222031 | 3750 |
| | | Positive | 0.917500676 | 0.904533333 | 0.910970861 | 3750 |
| | | Accuracy | 0.9116 | 0.9116 | 0.9116 | .9116 |
| | | Macro avg | 0.911682234 | 0.9116 | 0.911595585 | 7500 |
| | | Weighted avg | 0.911682234 | 0.9116 | 0.911595585 | 7500 |
| | 20 | Negative | 0.8943068 | 0.9048 | 0.8995228 | 3750 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | Positive | 0.903669725 | 0.893066667 | 0.89833691 | 3750 |
| | | Accuracy | 0.898933333 | 0.898933333 | 0.898933333 | .9116 |
| | | Macro avg | 0.898988262 | 0.898933333 | 0.898929855 | 7500 |
| | | Weighted avg | 0.898988262 | 0.898933333 | 0.898929855 | 7500 |


Recall  by Model and Category
Value
1.0
0.8
0.6
0.4
0.2
0.0
Category
Negative
Positive
Accuracy
Logistic Regression
DistilBERT (without fine-tuning)
DistilBERT (with fine-tuning)
Model


f1-score by Model and Category
Value
0.8
0.6
0.4
0.2
0.0
Category
Negative
Positive
Accuracy
Logistic Regression
DistilBERT (without fine-tuning)
DistilBERT (with fine-tuning)
Model



5. Which model would you recommend for deployment in a real-world scenario, and why? Consider both performance and efficiency in your answer.

For deployment in a real-world scenario, I would recommend the fine-tuned DistilBERT model with 3 epochs. This model achieves high performance with an accuracy of 92.88% and strong precision, recall, and F1-scores for both positive and negative classes, indicating excellent generalization to unseen data. Additionally, DistilBERT is a lighter and more computationally efficient transformer compared to larger models like BERT, making it more suitable for deployment scenarios where latency and resource constraints are considerations. Training beyond 3 epochs, as seen with the 10- and 20-epoch versions, leads to diminishing returns or slight overfitting, suggesting that the 3-epoch fine-tuned model balances performance and efficiency effectively. While classical models like logistic regression are more efficient, their significantly lower performance (e.g., around 87% accuracy) makes them less suitable for tasks requiring nuanced language understanding. Therefore, the 3-epoch fine-tuned

DistilBERT model offers the best trade-off between accuracy, efficiency, and deployment feasibility.