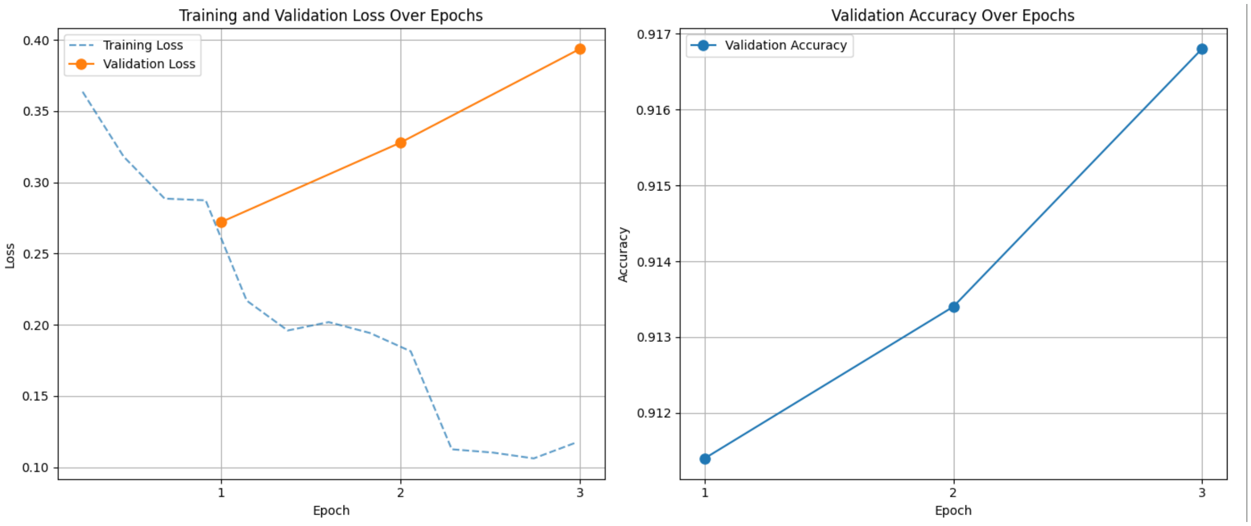
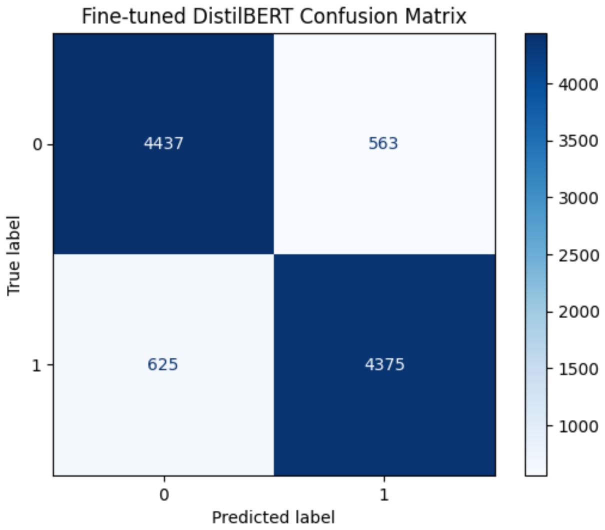
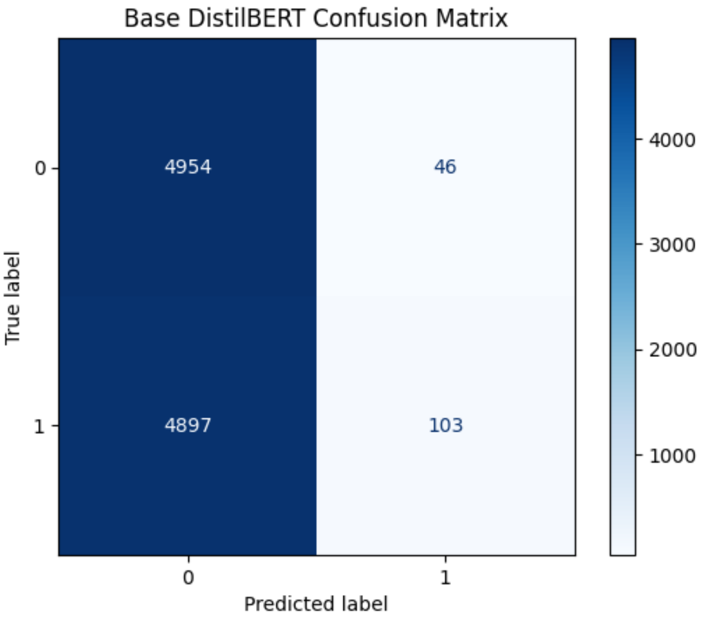
Project B Report – Kyle Hix

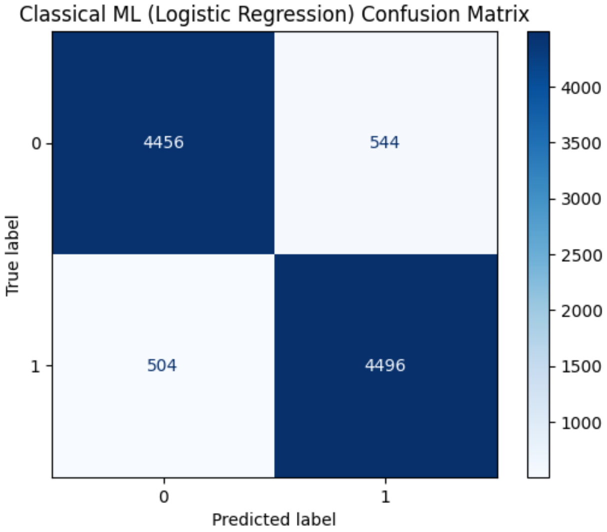
Accuracy and loss curve plots



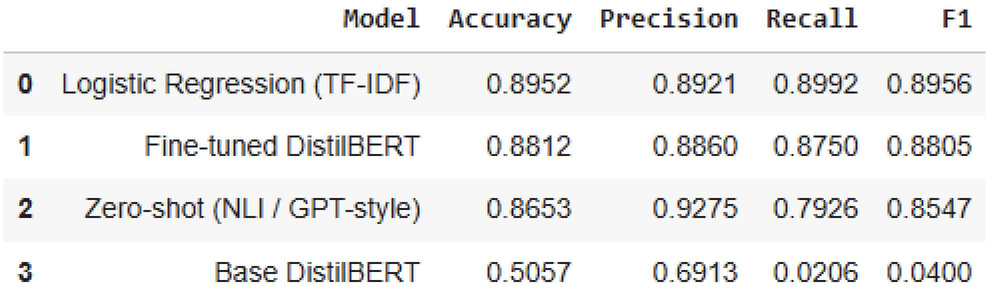
Confusion Matrix



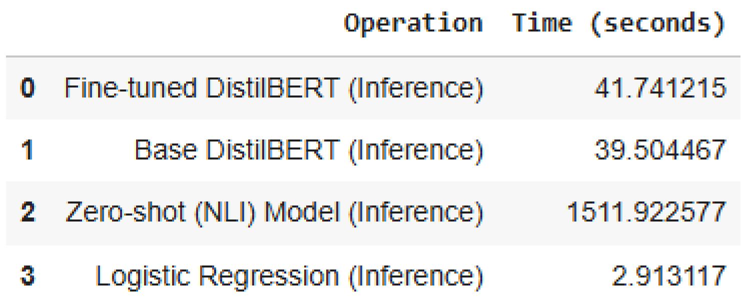




Comparative Tables



	Model	Accuracy	Precision	Recall	F1
0	Logistic Regression (TF-IDF)	0.8952	0.8921	0.8992	0.8956
1	Fine-tuned DistilBERT	0.8812	0.8860	0.8750	0.8805
2	Zero-shot (NLI / GPT-style)	0.8653	0.9275	0.7926	0.8547
3	Base DistilBERT	0.5057	0.6913	0.0206	0.0400



	Operation	Time (seconds)
0	Fine-tuned DistilBERT (Inference)	41.741215
1	Base DistilBERT (Inference)	39.504467
2	Zero-shot (NLI) Model (Inference)	1511.922577
3	Logistic Regression (Inference)	2.913117

Questions

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

   The plots show that as the fine-tuning process trains, accuracy increases and the loss decreases. This shows that our model is getting better after every epoch.

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 model trains exponentially faster than the classical model. The run time for classical is very slow. More GPU is needed for the classical model. On the other hand, the fine-tuned model is susceptible to overfitting. Eventually the fine-tuned model will not be a reliable source.

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

   The fine-tuned model and the classical model produce very similar results. Their differences are marginal. The misclassifications are minuscule, but it highlights that both models can give improper results. The baseline model produced mainly negative results regardless of the review type.

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

   The fine-tuned model outperforms the base model because it has been trained to work on this specific type of data, whereas the base model has no previous understanding.

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

   The fine-tuned model would be best for a real-world scenario. It is more accurate and is quicker to train. However, this would be quick for many smaller datasets, because this model is prone to overfitting.