FINAL REPORT

TITLE:

Evaluating Sentiment Classification Techniques on Social and Movie Review Data

INTRODUCTION:

Sentiment analysis is widely used in various applications such as customer feedback systems, online reviews, and brand monitoring. It plays a significant role in understanding public opinion through textual data.

In this project, I explored and compared the performance of several models for binary sentiment classification using a sample of the IMDb movie review dataset. The goal was to assess how well traditional machine learning techniques perform in contrast to modern deep learning and transformer-based approaches, especially on a small, cleaned dataset.

The models included Logistic Regression, Linear Support Vector Classifier (SVC), an LSTM-based neural network, and a pretrained transformer model (BERT). Each model was trained and evaluated using consistent data splits and common metrics such as accuracy and F1 score.

DATA AND PREPROCESSING:

The IMDB dataset was used, consisting of 500 labeled movie reviews evenly split between positive and negative sentiments. Text preprocessing involved:

- Removing HTML tags via BeautifulSoup,
- Cleaning non-alphabetic characters using regular expressions,
- Converting to lowercase and stripping whitespace.

A label encoder transformed sentiment labels (positive, negative) into binary format (1, 0). For traditional ML models, TF-IDF vectorization with a vocabulary size of 2000 was applied. For deep learning models, tokenization and padding/truncation to standard lengths were used.

BASELINE MODELS:

Logistic Regression

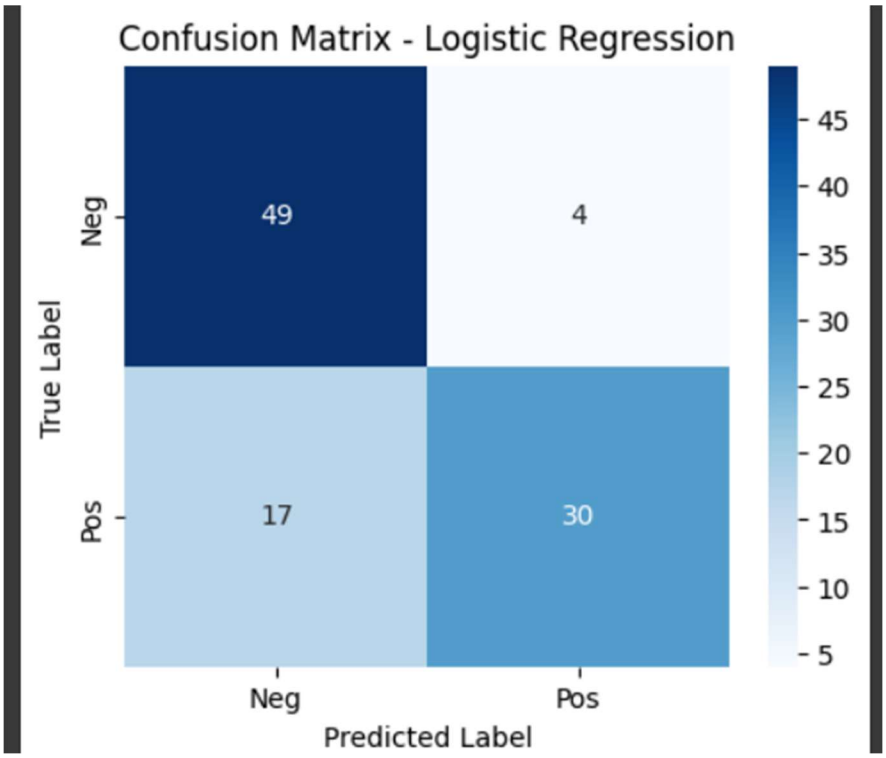
Logistic Regression served as our initial baseline. After training on TF-IDF features, it achieved:

- **Accuracy**: 0.79
- **F1 Score**: 0.78
- **AUC-ROC**: 0.8719

```
Logistic Regression Results
              precision    recall  f1-score   support

           0       0.74      0.92      0.82        53
           1       0.88      0.64      0.74        47

    accuracy                           0.79       100
   macro avg       0.81      0.78      0.78       100
weighted avg       0.81      0.79      0.78       100

AUC-ROC: 0.871938980329185
```

Confusion Matrix:

The model performed better on negative reviews, with a strong precision of 0.88 for positive samples, but lower recall (0.64).



3.2 Linear Support Vector Classifier (LinearSVC)

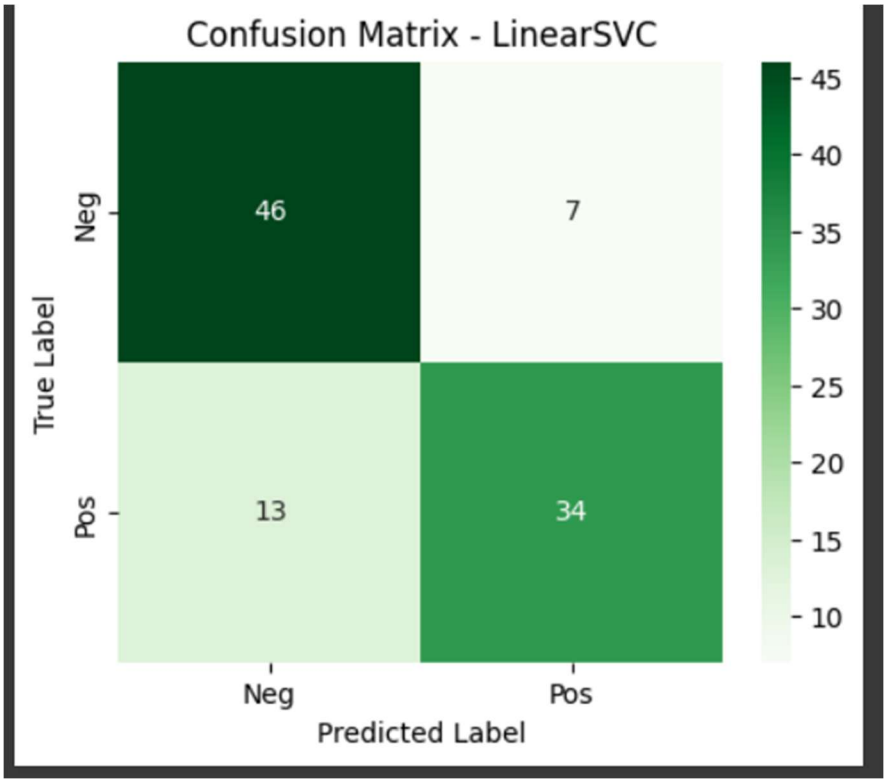
LinearSVC outperformed Logistic Regression slightly in both accuracy and F1:

- **Accuracy**: 0.80
- **F1 Score**: 0.78

```
LinearSVC Results
Accuracy: 0.8
F1 Score: 0.7727272727272727
              precision    recall  f1-score   support

           0       0.78      0.87      0.82        53
           1       0.83      0.72      0.77        47

    accuracy                           0.80       100
   macro avg       0.80      0.80      0.80       100
weighted avg       0.80      0.80      0.80       100
```

Confusion Matrix:

LinearSVC showed improved recall and overall better class separation, especially for the positive sentiment class.



DEEP LEARNING APPROACH- LSTM:

I built a sequential LSTM model using Keras. The architecture included:

- An embedding layer (128 dims),
- A 64-unit LSTM layer,
- Dropout regularization,
- A final Dense sigmoid output.

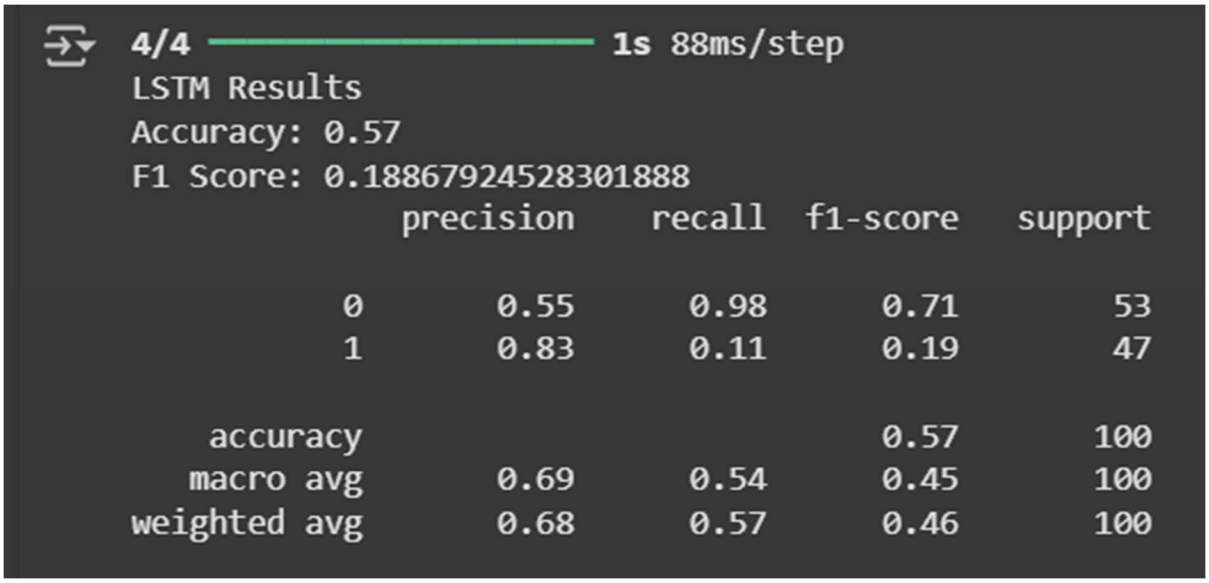
Training stats:

- Epochs: 2
- Batch size: 128
- Training accuracy: 0.69
- Validation accuracy: 0.51

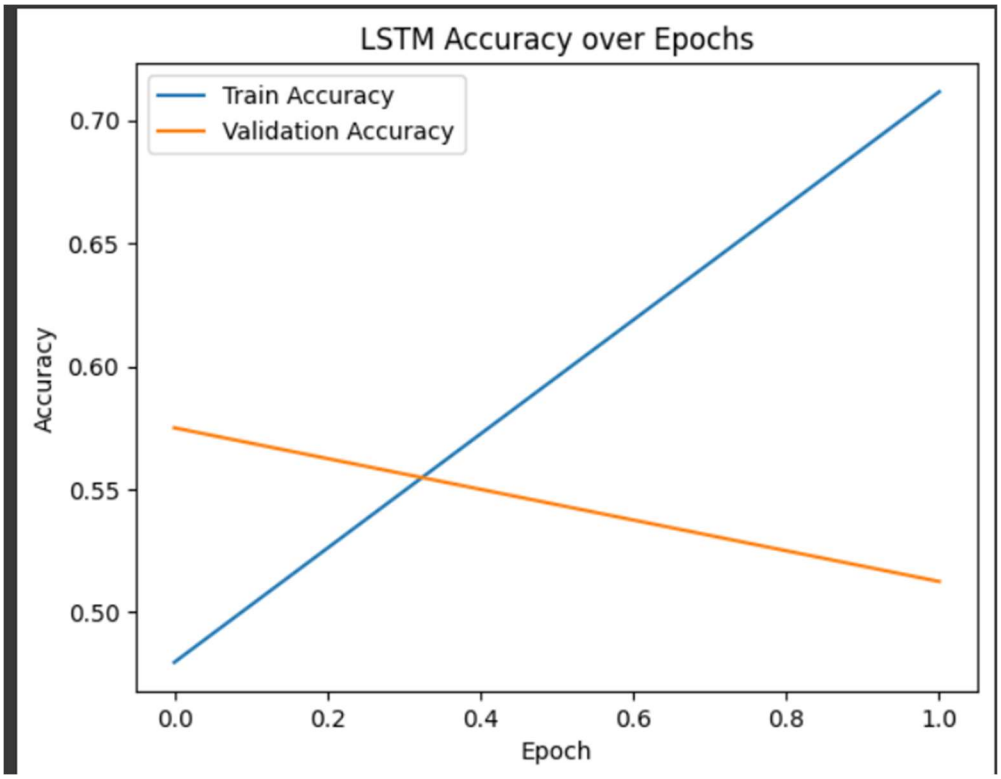
Test set evaluation:

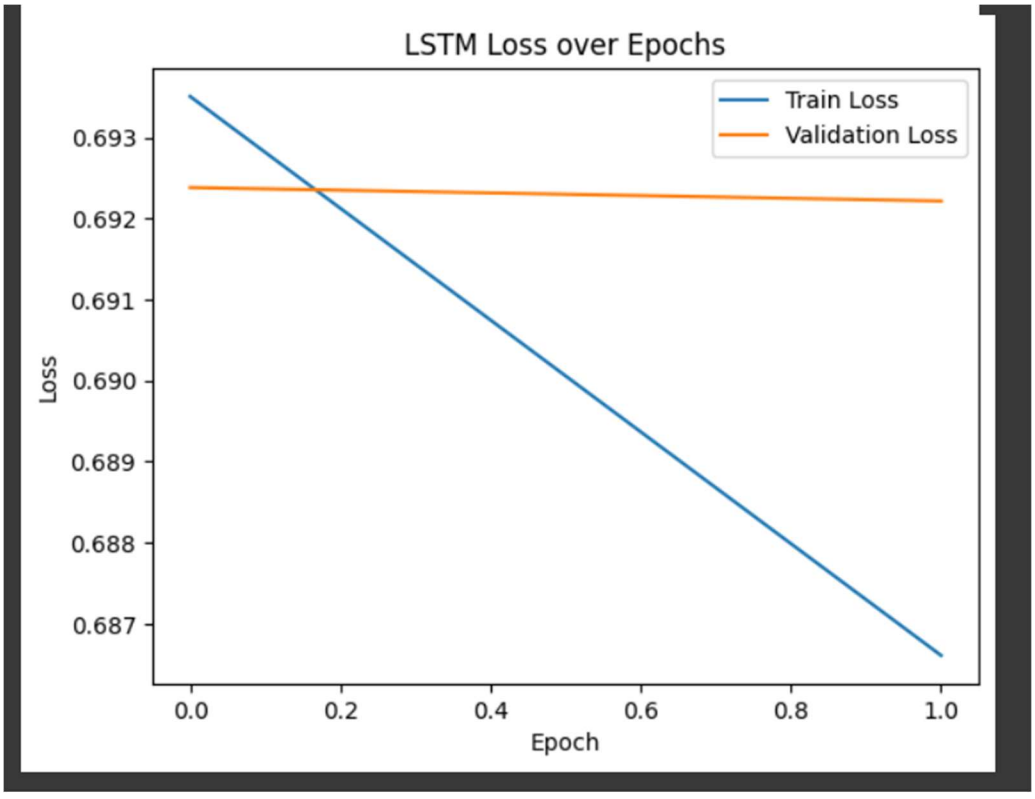
- Accuracy: 0.57
- F1 Score: 0.19

Performance dropped significantly, especially for positive class recall (0.11), suggesting underfitting or insufficient training due to the small dataset size.



```
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 88ms/step
LSTM Results
Accuracy: 0.57
F1 Score: 0.18867924528301888
              precision    recall  f1-score   support

           0       0.55      0.98      0.71        53
           1       0.83      0.11      0.19        47

    accuracy                           0.57       100
   macro avg       0.69      0.54      0.45       100
weighted avg       0.68      0.57      0.46       100
```


LSTM Accuracy over Epochs
Train Accuracy
Validation Accuracy
0.70
0.65
0.60
0.55
0.50
Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
Epoch


LSTM Loss over Epochs
Train Loss
Validation Loss
0.693
0.692
0.691
0.690
0.689
0.688
0.687
Loss
0.0
0.2
0.4
0.6
0.8
1.0
Epoch

TRANSFORMER-BASED MODEL - BERT:

BERT (bert-base-uncased) was fine-tuned using the Hugging Face Trainer API. Training output:

- **Train Loss**: 0.62
- **Training Time**: ~17 minutes
- **Steps**: 50

Evaluation results:

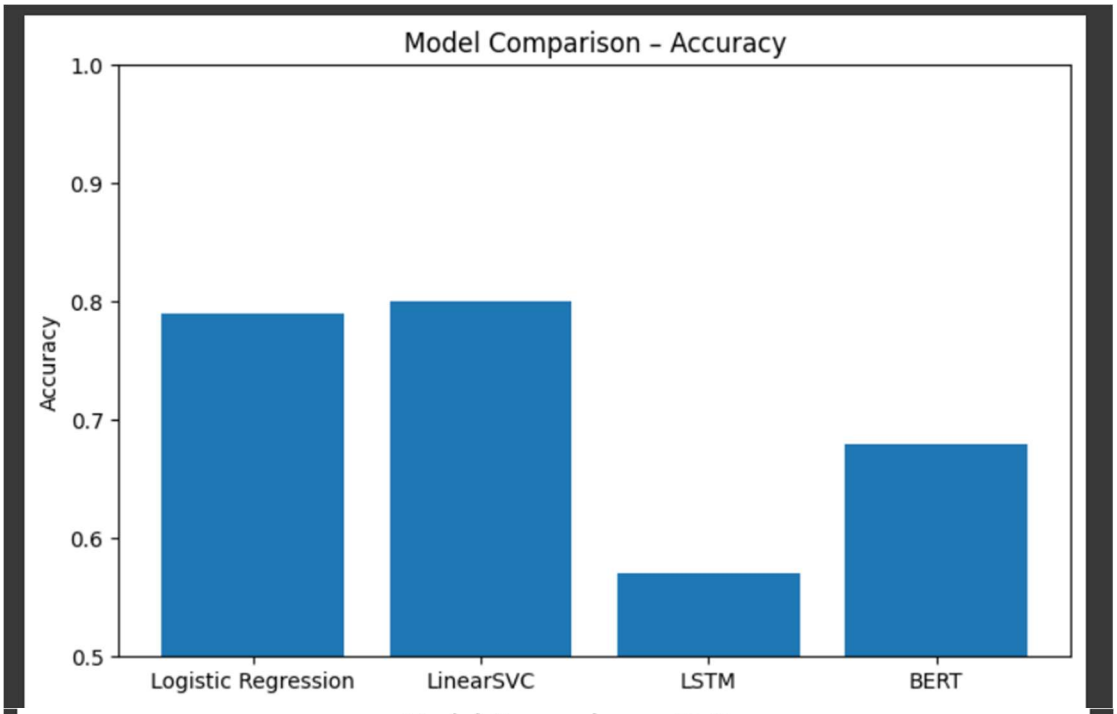
- **Accuracy**: 0.68
- **F1 Score**: 0.74
- **Precision (Class 0/1)**: 0.92 / 0.60
- **Recall (Class 0/1)**: 0.43 / 0.96

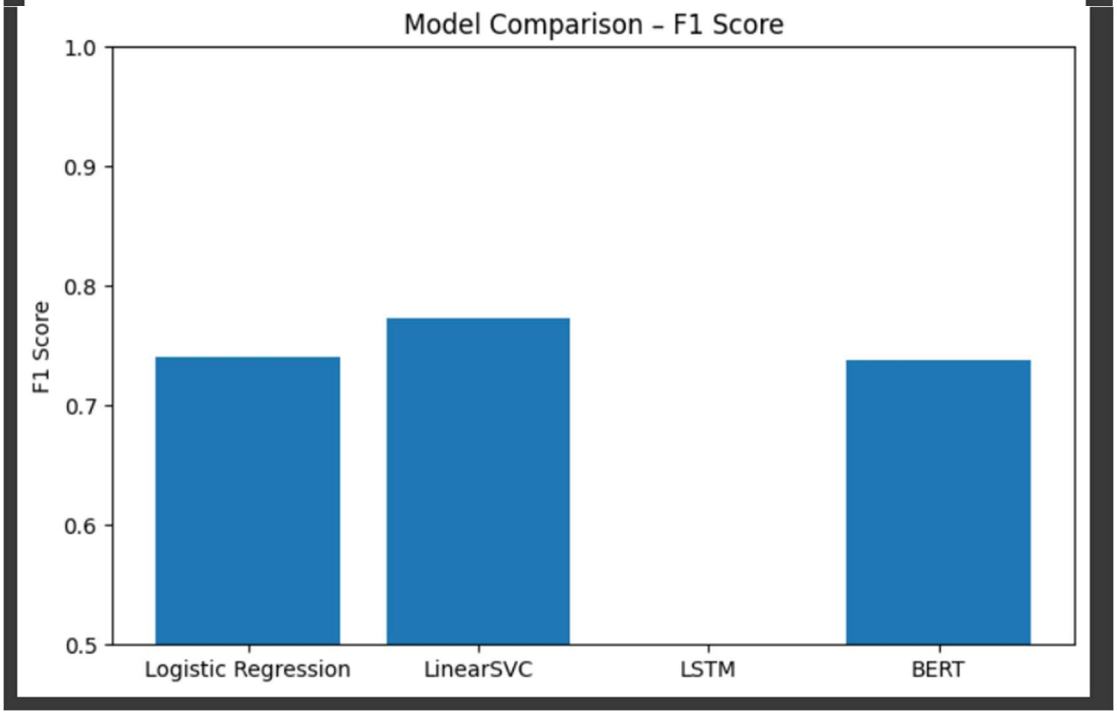
BERT displayed high recall for positive samples but over-predicted them, leading to low recall for negative class.

```
BERT Accuracy: 0.68
BERT F1 Score: 0.7377049180327869

Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.43      0.59        53
           1       0.60      0.96      0.74        47

    accuracy                           0.68       100
   macro avg       0.76      0.70      0.66       100
weighted avg       0.77      0.68      0.66       100
```

MODEL COMPARISON:

Bar charts and confusion matrices further visualize performance. LinearSVC consistently led across metrics, followed closely by Logistic Regression. While BERT showed promising F1 score, it underperformed in overall accuracy compared to classical models. LSTM, in this case, suffered from poor generalization due to data scarcity and short training.





CONCLUSION:

This project allowed me to compare a variety of techniques for sentiment classification, ranging from simple linear models to advanced neural architectures.

The best-performing traditional model was LinearSVC, which slightly outperformed Logistic Regression. These models proved to be reliable and efficient on small, vectorized datasets.

The LSTM model did not perform well, likely due to the limited number of training examples and only two training epochs. Despite its potential, it showed signs of underfitting and produced poor F1 scores, particularly for the positive class.

BERT, although trained on a reduced dataset due to resource limitations, produced promising results with a strong F1 score. However, it over-predicted positive samples, indicating class imbalance sensitivity.

In conclusion, this comparison highlights that well-tuned classical models are still very competitive for small-scale sentiment analysis. Future work could include experimenting with larger datasets, performing hyperparameter optimization, increasing training time for LSTM and BERT, and testing more recent transformer variants like RoBERTa or DistilBERT.

SOURCES:

- **IMDB Movie Reviews Dataset**
Pathi, L. N. (2018). *IMDB Dataset of 50K Movie Reviews*. Kaggle. https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

- **TF-IDF Vectorization**
Ramos, J. (2003). *Using TF-IDF to determine word relevance in document queries*. In Proceedings of the First Instructional Conference on Machine Learning.

- **Logistic Regression**
Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (3rd ed.). Wiley.

- **Support Vector Machines (SVM)**
Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297.

- **LSTM Networks**
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

- **BERT: Bidirectional Encoder Representations from Transformers**
Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of deep bidirectional transformers for language understanding*. NAACL-HLT. https://arxiv.org/abs/1810.04805

- **Hugging Face Transformers Library**
Wolf, T., Debut, L., Sanh, V., et al. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. https://huggingface.co/transformers/

- **Scikit-learn (for LR and SVM)**
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830. https://scikit-learn.org/

- **Keras / TensorFlow (for LSTM)**
Chollet, F. (2015). *Keras*. https://keras.io
TensorFlow Team. (2015). *TensorFlow*. https://www.tensorflow.org