1. Problem Statement
   The project aims to develop a market sentiment analysis application that uses machine learning to classify the sentiment of financial news headlines as positive, negative, or neutral. Using the Kaggle Financial Sentiment Dataset and the FinBERT model, the application will allow users to input a stock ticker and time period, automatically fetch related news headlines, and provide a sentiment classification with confidence scores, along with a sentiment trend graph over the chosen period.
2. Data Preprocessing

   I confirm that I will use Kaggle Financial Sentiment Dataset :
   https://www.kaggle.com/datasets/sbhatti/financialsentiment-analysis

   The Kaggle Financial Sentiment Dataset contains approximately 5322 financial news headlines and sentences labeled as positive (1), negative (-1), or neutral (0). The dataset is relatively imbalanced with roughly 32% positive, 54% neutral, and 15% negative labels. It is specifically designed for financial sentiment analysis.

   My preprocessing approach included:

   1. Text cleaning: Remove URLs, special characters, and extra whitespace while preserving financial symbols like $ and % which carry important meaning in financial contexts. At first, I thought I should clean the data in a more robust way with stop word removal and lemmatization, but I realized this would affect the model (FinBERT) performance since it was trained to understand grammatical structure.
   2. Handling missing values: No missing values were found in the headline/sentence column, so no data was dropped for this reason.
   3. Duplicate handling: Find and remove 525 duplicate entries to prevent the model from being biased toward repeated content.
   4. Tokenization: Use FinBERT's specialized financial tokenizer which properly handles financial terminology, numbers, and symbols.
   5. Preserved text case: Maintain the original case structure as financial abbreviations, tickers, and company names often have specific capitalization patterns that carry meaning.

      These preprocessing steps were designed to clean the data while preserving financial-specific terminology and structure, maximizing the relevant information available to the FinBERT model.

3. Machine learning model

a. Framework and Tools

I implemented the sentiment analysis model using PyTorch and the Hugging Face Transformers library. The architecture is based on FinBERT, a BERT-based model specifically fine-tuned for financial text. FinBERT consists of:

- 12 transformer encoder layers
- 12 attention heads per layer
- 768 hidden dimensions
- 109M parameters total

b. Training Decisions

- Data Split: 80% training, 20% validation using stratified sampling to maintain class distribution
- Hyperparameters: Learning rate of 2e-5, batch size of 20, 4 epochs, and max sequence length of 160
- Regularization: Applied weight decay (0.01) to prevent overfitting
- Optimization: Used AdamW optimizer with warmup steps (500) to stabilize early training

c. Validation Methods

I evaluated the model using stratified cross-validation on held-out data. The eval_loss of 0.354 and accuracy of 85.4% indicate the model is neither overfitting nor underfitting. The balanced precision (84.9%) and recall (85.4%) scores further support this conclusion.

d. Challenges

The main implementation challenges were:

1. Text preprocessing incompatibility with FinBERT's expected input format (resolved by using minimal cleaning)
2. Handling textual sentiment labels in the dataset (solved by creating a robust mapping function)
3. Path inconsistencies when saving the model (fixed with absolute path handling)

4. Preliminary results

   The model achieved strong performance with:

   - Accuracy: 85.4% (so much better than the baseline I expected)
   - F1 Score: 84.9%
   - Precision: 84.9%
   - Recall: 85.4%
   - Eval Loss: 0.354

   These metrics demonstrate balanced performance across all sentiment classes. The relatively low eval_loss indicates well-calibrated probability predictions, essential for financial decision-making. The model successfully classifies financial headlines with high accuracy while maintaining a good balance between precision and recall.

5. Next steps

   A huge pro is that the metrics are well balanced (no surprise since FinBERT was trained before, I am just trying to fine tune it.

   A huge con is that the data set is dominated with neutral label, so the model may default to neutral prediction in ambiguous cases and struggle with subtle positive/negative sentiment, which in the world of finance, is not very useful since that would mean there is no action can be taken.

   Future work:

   - Fetch financial news headlines and perhaps manually label some of them to continue to train the model even more.
   - Deploy and enhance one page web application with additional visualization options