Predicting Human Preferences for LLM Responses: A Multi-Stage Machine Learning Project

1. Introduction

This project aims to develop models that predict human preference ratings between two candidate LLM responses given a prompt, using the dataset from the Kaggle "LLM Classification Finetuning" competition. Unlike simple correctness classification, the task relies on nuanced human judgment, introducing unique challenges and relevance for current AI evaluation research.

2. Dataset and Task Description

The dataset consists of prompt-response pairs, each labeled with one of three preference outcomes: A Wins, B Wins, or Tie. For each instance, a human annotator is presented with a prompt and two distinct candidate responses—these may vary in length, style, factual accuracy, or tone. The annotator is then asked to select the response that better fulfills the prompt, or to indicate if both are equally suitable, resulting in the "Tie" label. Both the diversity of prompts and responses, ranging from factual queries to open-ended and conversational tasks, and the subjective nature of labeling (which reflects human preferences rather than absolute correctness) make the task particularly challenging. Additionally, exploratory analysis revealed that the distribution of classes is moderately imbalanced: "Tie" cases occur less frequently than clear A/B preferences, which further complicates model training and performance assessment.

3. Model Development and Experimental Stages

Step 1-2: Baseline Modeling and Feature Engineering

Developed a pipeline based on textual and statistical features such as sentence length, word count, lexical richness, and punctuation per response.

- Used logistic regression for multiclass classification and implemented stratified train/validation splits with fixed random seeds.
- The baseline results indicated strong performance for "A Wins" and "B Wins", but failed to reliably identify "Tie" samples. Validation accuracy was approximately 44%, and multiclass log loss was 1.07.

Step 3: Enhanced Features, Embeddings, and Ensembling

- Added pretrained sentence embeddings (MiniLM, E5) and derived relationship features between prompts and responses.
- Introduced bias-aware features (position, verbosity, formatting complexity, etc.).
- Explored ensemble classifiers (Random Forest, XGBoost, LightGBM), soft voting, weighted averaging, and probability calibration.
- Achieved improvement with embedding-based models and ensembles: accuracy up to 51%, log loss down to 0.98, and recall for "Tie" up to 35~39%.

Step 4: Error Analysis and Diagnosis

- Conducted confusion matrix, F1-score, precision/recall, and error case reviews.
- Identified "Tie" as the main bottleneck due to annotation subjectivity and feature limitations, offering targeted remedies (oversampling, binary chain, richer features).
- Provided clear visualizations and diagnostic comments, leading to concrete improvement strategies.

Step 5: Final Model Integration and Submission

- Tuned ensemble weights and finalized submission.
- Documented environment (Python 3.11, GPU type, package list, fixed seeds), ensuring full reproducibility in code and results.
- Labelled the best model function, included Kaggle leaderboard screenshot, and summarized submission protocol.

4. Validation Strategy and Comparative Analysis

Metric	**Baseline**	**Embedding**	**Ensemble**	**Notes**
Log Loss	1.07	1.02	0.98	Gradual improvement
Accuracy	0.44	0.49	0.51	Improvement observed
Recall (Tie Class)	0.05	0.35	0.39	Major bottleneck

Class-wise performance confirmed the “Tie” identification as the main limitation, while other classes benefited from feature engineering and ensembling.

5. Key Lessons and Technical Gains

- Text-based features alone are insufficient for modeling human linguistic nuance; embeddings and bias-aware designs proved highly effective.
- The ability to identify “Tie” samples depends on advanced feature design, careful engineering, and appropriate model choices.
- Maintaining clarity and reproducibility in pipelines, documentation, and environment setup is essential in applied ML.
- External validation with Kaggle competition protocols highlights the need for repeatable experimental methodologies.ml-tp3.ipynb+1

6. Error Analysis, Limitations, and Future Directions

The current model achieves noticeable improvements in log loss and overall accuracy but remains limited in “Tie” class detection due to feature expressiveness, class imbalance, and subjectivity in human annotation.

Directions for further optimization:

- Specialized binary tie classifiers, oversampling techniques, and domain-specific feature designs.
- Transformer-based self-supervised and meta-learning methods, improved label curation, and annotator bias management.
- Building robust, reproducible codebases and automated experiment management for future scalability and collaboration.

7. Project Summary and Report Writing

Each stage contributed major advancements:

- Feature engineering, embeddings, calibration, and ensemble strategies iteratively improved model performance and interpretation.
- Diagnostic analysis and technical documentation ensured transparent reporting and allowed reproducibility for future research and application.

- The report organizes methodology, experimental progression, key findings, and actionable future directions clearly, enabling presentation as an experiment report or academic summary.

8. Author Contributions

Yang Haochen completed Step 1 and Step 2 of the project Baseline Modeling and Feature Engineering.Wang Haoze was responsible for Step 3 Enhanced Features, Embeddings, and Ensembling.Jin Junwu worked on Step 4 Error Analysis and Diagnosis.Pan Lubiao handled Step 5 Final Model Integration and Submission.Li Wenbin prepared and wrote the final project report.