Hackathon Report

Manish Nayak

November 21, 2025

1 Introduction

The objective is to predict ‘metric score‘ (from 0 to 10) for a prompt and its response, for the corresponding metric being used for evaluation. A analysis of the training data shows severe class imbalance, as shown in Table 1.

Table 1: Initial and Modified Score Distribution in Training Data

Initial Distribution		**Modified Distribution**	
Score	**Count**	**Score**	**Count**
9.0	3123	9.0	3123
10.0	1442	10.0	1443
8.0	259	8.0	259
7.0	95	7.0	95
6.0	45	6.0	46
0.0	13	0.0	13
3.0	7	1.5	11
1.0	6	3.0	10
2.0	5		
4.0	3		
5.0	1		
9.5	1		

2 Data Preprocessing and Feature Engineering

- **Text Embedding:** The state-of-the-art `google/embeddinggemma-300m` model from the `SentenceTransformer` library was used to convert all text inputs into 768-dimensional numerical vectors.
- **Feature Creation:** Three distinct embeddings were generated for each data point:
  1. **Metric Embedding:** Retrieved from the pre-computed embeddings file.
  2. **Prompt Embedding:** The `system_prompt` and `user_prompt` were concatenated and encoded to represent the full conversational context.
  3. **Response Embedding:** The ‘response‘ text was encoded separately.

- **Data Splitting:** A stratified 80/20 split was performed to create training and validation sets. Stratification on binned score values ensured that the distribution of rare scores was preserved in both sets.

3 Modeling Methodology

3.1 Model 1: Hierarchical Attention with Optimized Weighted Loss

This model was made to deeply understand the semantic relationships between the prompt, response, and metric.

3.1.1 Architecture

The `HierarchicalAttentionScorer` model utilizes a two-layer cross-attention mechanism:

1. **Prompt-Response Interaction:** The prompt embedding (Query) attends to the response embedding (Key, Value) to produce a "context-aware" prompt that highlights aspects relevant to the given response.

2. **Context-Metric Alignment:** The metric embedding (Query) then attends to this new representation to align the conversation's context with the specific evaluation criterion. The output is fed to a prediction head to generate the final score.

3.1.2 Handling Imbalance

To combat data imbalance, a custom weighted Mean Squared Error (MSE) loss was implemented. This function applies significantly higher penalties for errors on low-score samples. The weights for different score tiers (e.g., "rare", "mid-rare") were treated as hyperparameters.

3.1.3 Optuna Hyperparameter Search

The **Optuna** framework was used to conduct an extensive search (150 trials) to co-optimize both the model's hyperparameters (e.g., number of attention heads, learning rate) and the loss weights. The best parameters found were:

- `num_heads`: 4
- `dropout`: 0.2986
- `lr`: 6.26e-05
- `weight_rare` (scores less than 6): **1001.36**
- `weight_mid_rare` (scores 6-8): **87.92**
- `weight_8`: **11.96**
- `weight_most_common`: **1**

The model achieved a best weighted validation RMSE of **9.627**.

3.2 Model 2: VQ-VAE, SMOTE-NC, and XGBoost

This model uses a feature-engineering-centric approach to address the imbalance at the data level.

3.2.1 Methodology

This was a three-stage pipeline:

1. **Feature Discretization with VQ-VAE:** Three separate Vector Quantized Variational Autoencoders (VQ-VAEs) were trained to learn a discrete "codebook" (size 256) for each embedding type. This converted the high-dimensional, continuous embeddings into single categorical codes, performing a powerful non-linear dimensionality reduction.

2. **Data Balancing with SMOTE-NC:** With the features now categorical, the Synthetic Minority Over-sampling Technique for Nominal and Continuous features (SMOTE-NC) was applied. A tiered strategy synthetically generated new data points for minority classes, with the rarest scores (e.g., less than 6.0) being oversampled by a factor of 40 times.

3. **XGBoost for Regression:** An XGBoost regressor was trained on the new, balanced dataset of discrete codes. The model's inherent ability to handle categorical features was leveraged.

4 Ensembling Strategy and Final Results

The final prediction was generated by ensembling the outputs of both models.

- **Rationale:** Model 1 is a specialist, highly focused on identifying low-scoring samples. Model 2 provides a more generalized perspective from training on a synthetically balanced feature space. By combining their predictions, we leverage the specialized knowledge of Model 1 while regularizing it with the balanced view of Model 2.

- **Method:** A simple average of the predictions from both models was used to produce the final submitted score.

5 Conclusion

The Hierarchical Attention Network with an Optuna-tuned weighted loss (Model 1) and the VQ-VAE + SMOTE-NC + XGBoost pipeline (Model 2) represent two distinct methods for handling this challenge.