Data Analytics Lab - Final Project

Sree Varshini V P (ED22B082)

November 21 2025

1 A Brief on the Problem Statement

We aim to predict the fitness score from 1-10 which tells us how well the prompt response pair (which is also the test data) aligns with the metric definition which has been embedded. Hence, we have 2 inputs (metric embedding and prompt-response pair and one output).

Here, the 2 inputs are provided are the metric definitions which were converted to text embeddings using the sentence transformer google/embeddinggemma-300m and the prompt response pairs given as raw files. The output is scores vs ID table for all the 3638 test samples.

The problem statement reads that the ability to accurately model this fitness is crucial for automatically generating and curating high-quality test datasets.

We finally aim to reduce RMSE error (the metric used).

2 Major Challenges

- **Class Imbalance (Skewed data)**: The dataset provided was found to have a class imbalance with higher number of higher valued scores (7-10 approximately). This resulted in high RMSE scores when methods such as sample weights/ SMOTE was not applied.

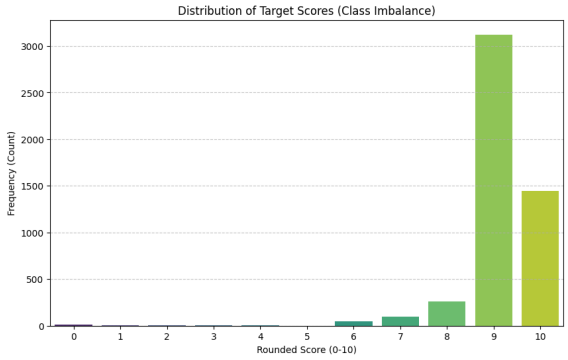

Figure 1: Skewed class distribution

- **Prediction of the LLM judge's predictions**: Since the ground truth here was the predictions of an LLM judge, and no ground truth of actual correlation exists, understanding of whether the LLM's biases led to high RMSEs was challenging.

3 Methodologies

Three main methodologies emerged from the multiple methods used to solve the challenge:

1. A LightGBM Model- based method
2. An ensemble of LightGBM, Ridge Regression and XGBoost
3. A Neural Network based method with Negative Contrastive Loss

Here is the methodology used in each of them: (ater data loading)

3.1 1. LightGBM + Sample Weights

This methodology utilised the LightGBM Regressor, considering the problem as a regression. This was the simplest approach involving similarity features and class imbalance strategies.

3.1.1 Reasoning: Why LightGBM was chosen

Of several models, LightGBM was chosen (and provided the best results since):

- It is known to perform Gradient based one side sampling (it filters out a significant portion of the data with small gradients which are the easy examples and focus on training the remaining data instances with large gradients which are the hard samples. It also does exclusive feature bundling to bundle many exclusive features into a single one. This makes it faster and efficient.
- It is also known to handle the high dimensional data provided well.

3.1.2 Methodology explained

- **Embedding**: The SentenceTransformer('paraphrase multilingual mpnet base v2'), the multilingual Sentence Transformer model to convert the combined text of the Prompt-Response Pair into a dense numerical text embedding.

  Note: Other attempts: Other embeddings such as all-MiniLM-L12-v2 and EmbeddingGemma-300M were tried over which multilingual mpnet base v2 gave the best scores.

- **New features computed**:

$$\mathbf{X} = [\,\mathbf{E}_M,\ \mathbf{E}_T,\ |\mathbf{E}_M - \mathbf{E}_T|,\ \mathbf{E}_M \odot \mathbf{E}_T,\ \|\mathbf{E}_M - \mathbf{E}_T\|_2\,]$$

  (Em: metric embedding; Et: Text embedding)

  These features were computed and fed, which includes Euclidean distance and dot product **to capture complex non linear relationships** in the data to improve performance.

- **Concatenation**: The user prompt, response and optional system prompt are concatenated into a single text string, which provides the full conversational context for embedding.
- **Sample weights**: In order to account for class imbalance, sample weights are assigned inversely proportional to occurrence frequency:

$$w_i = \frac{1}{f_{y_i}}$$

- **Note: Other attempts**: Other attempts include oversampling and SMOTE of which sample weights gave the best results.
- **Train test split**: Train test split is performed with stratify = y to ensure equal representation across both sets.
- **LightGM Paramters**: The model is trained on the LightGBM Model with the following set of parameters which turned out to be the most optimum after tuning:

  objective='rmse', metric='rmse', $n_estimators = 2000, learning_rate = 0.0005, num_leaves = 15, lambda_l1 = 0.5, lambda_l2 = 0.7, feature_fraction = 0.7, bagging_fraction = 0.7, min_child_samples = 50, bagging_freq = 1, verbose = -1, n_jobs = -1, seed = 42$

- **Final prediction**: The scores are predicted in the range 1-10. They are rounded off if RMSE scores proved lower.

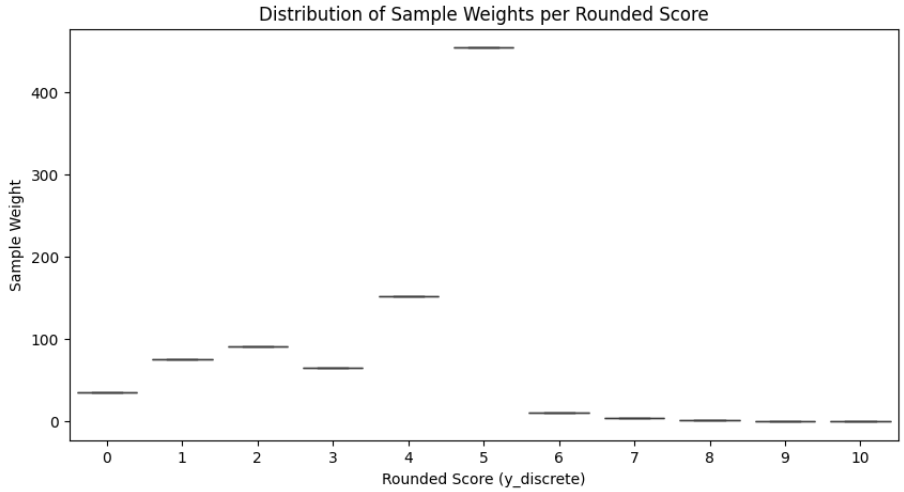

Figure 2: Sample weights according to

3.2 Ensemble of LightGBM + Ridge Regression + XGBoost

3.2.1 Reasoning: Why the ensemble was chosen

- **XGBoost** is known to use both L1 Lasso and L2 Ridge regularization terms in its objective function. This controls the complexity of the final model and prevents overfitting of noise
- Sample weights are also natively enabled in XGBoost which is further advantageous.
- **Ridge Regression** This is known to provide a simple and stable linear model hat is less sensitive to noise or local fluctuations as compared to the tree based models. When averaging its predictions with the more complex models, it effectively reduces the overall variance of the ensemble.
- **Ensemble** Hence, ridge regression was expecte

3.2.2 Methodology explained

- The hyperparameters of the LightGBM Model were retained as they provided the best results.
- **XGBoost Hyperarameters**: xgbmodel = xgb.XGBRegressor(objective='reg:squarederror', $\mathrm{n}_{e}stimators = 1000, learning_{r}ate = 0.01, max_{d}epth = 5, min_{c}hild_{w}eight = 1, subsample = 0.7, colsample_{b}ytree = 0.7, reg_{l}ambda = 1, reg_{a}lpha = 0.5, random_{s}tate = 42, n_{j}obs = -1, tree_{m}ethod =' hist', verbosity = 0$)

Early stopping = 100 was used to prevent overfitting. hyperparameters were tuned to arrive at the above.

- **Ridge Regression**: sklearn.linearmodel.Ridge was used. It was intended to smoothen out the predictions of complex tree based methods.
- **The Ensemble**: Simple averaging was used to combine all the 3 models

  Note: Other attempts: Other methods such as of combining did not lead to an improvement of scores. Hence, the results of the simple average ensemble are provided.

3.3 Neural Network and Negative Contrastive Labels Based Method

3.3.1 Reasoning: Why NNs

- We seek to evaluate the semantic fitness between a text pair and a metric. A Dual projector network projects the two high dimensional input vectors into a lower dimensional latent space where their dot product directly corresponds to the final fitness score. So this projection is a non linear mapping which is basically optimized to capture relevance. This is better than relying on a linear combination of raw concatenated features.
- The negative contrastive labelling produced significant results. This is because the network learns not only that the correct metric is similar to the text (which is a positive pair) but also other incorrect metrics (which are negative samples) which are dissimilar.

  So the distance between a prompt response pair and its correct metric is minimized, while distance to all sampled incorrect metrics is maximised. So this makes it far more reliable.

3.3.2 Methodology

The same embedding 'paraphrase multilingual mpnet base v2 is used' (which provided the best results here as well).

- **The negative contrast class**: This creates training batches for negative contrastive learning. A positive pair, which is a prompt response embedding and its correct metric embedding and other hard negative metric embeddings are sampled. Hence, both correctly and in
- **The soft layer norm**: This performs standard layer normalisation.
- **Dual Projector**: This is used to learns the similarity function. It takes two embeddings (x for sample and m for the metric) and outputs a similarity score. **Structure**: It basically uses two separate soft norm layers and two separate projection head sequences (the fx and fm), where each consists of two linear layers with a ReLU activation in between.
- **Loss function**: The loss function is modeled as a function of 3 terms to get maximum results. These include regression loss, positive binary cross entropy lo

4 Results

Here are the RMSE score from the 3 types of models

1. LightGBM : 3.633
2. Ensemble: 3.716
3. Neural Network: 2.837

   Other similar combinations produced scores similar/close to the LightGBM and Ensemble scores given here.

5 Inferences

(a) **Why the neural network performs the best**: The dual projector takes the text and metric embeddings and non linearly compresses them into a new space. So it is like a custom filter for the two embeddings. Since, in this new space, the network learns to make the embeddings of the correct pairs group together and pushes dissimilar ones apart, it is highly efficient for the problem at hand. This customisation is not performed by the tree based models.

(b) **Why the ensemble RMSE may be close/ slightly higher than the LightGBM RMSE**: Ridge regression may have made the model too simple. Hence, the dataset may have several non-linear relationships. Moreover, lightgbm performed better than XGBoost even individually since its specific regularisation may not have helped generalise well to this specific dataset. LightGBM proves most relevant in this dataset. **Inference**

(c) **Why methodologies such as sample weights, metric similarity features and negative contrast decreased RMSE significantly** Sample weights helped model focus on underrepresented data as data is skwewed. Metric similarity features also highlight that interaction between features is more important than just knowing raw features. Further, this dataset is sk