DA5401 Kaggle data challenge

M Aravindhan - DA25S006

November 21, 2025

Contents

1 Introduction

This project aims to evaluate the quality of multilingual text generated from LLM chatbot, especially for Indian languages with 145 diverse evaluation metrics including multilingual comprehension, accuracy and factual correctness whose embeddings are available but not the actual metric definitions, and the goal is to assign a performance score ranging from 0-10 .

1.1 Problem Statement

Predict quality scores (0-10) for text samples across 145 metrics while handling:

- Severe class imbalance (91% samples at 9-10 scores)
- Code-mixed multilingual text
- Limited training data (5,000 samples, 145 metrics)

1.2 Dataset Overview

Training: 5,000 samples — Test: 3638 samples — Metrics: 145 unique dimensions

2 Data Description and Preprocessing

2.1 Raw Data Structure

The dataset was loaded with 5000 training samples containing:

Table 1: Dataset Schema

Column	**Description**
metric_name	Evaluation metric identifier (145 unique)
score	Target variable (0.0 - 10.0)
user_prompt	User's query or request
response	Model's generated response
system_prompt	System instructions for model

2.2 Multilingual Composition

Considering the multilingual nature of the dataset, particularly the presence of code-mixed text and Indian native languages, I designed a multi-model ensemble language-detection pipeline to achieve the best performance in identifying languages.

2.2.1 Language Detection Approaches

To ensure high accuracy across various language types, I used following six language identification methods:

1. **LangID (Statistical Method):** A Naïve Bayes probabilistic model that predicts language using character-level statistics.
2. **Lingua (Machine Learning Detector):** A machine-learning detector that combines multiple linguistic cues for high-accuracy language prediction.

3. **XLM-RoBERTa Language Classifier:** A transformer-based multilingual model that identifies languages across 100+ languages.

4. **FastText Language Identification:** A neural network using character n-grams to classify text in 176 languages.

5. **Unicode Script-based Detection:** A rule-based heuristic that maps Unicode codepoint ranges to writing scripts for language inference.

6. **Qwen-based LLM Language Detection:** A generative LLM that outputs ISO language codes and handles noisy or mixed-language text.

7. **Majority-vote Ensemble Method:** A voting-based ensemble that selects the most frequent prediction across all detectors.

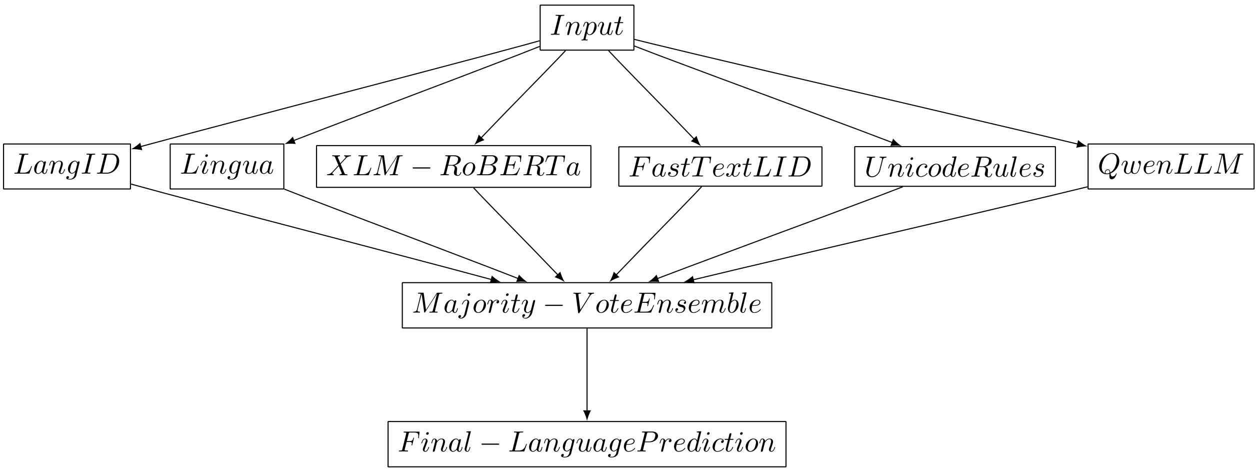

Figure 1: Pipeline of Language Detection Approaches

2.2.2 Language Distribution (Preprocessing)

The dataset includes multilingual content across 30 languages:

Table 2: Top Languages in Dataset (Preprocessing)

Language	**Count**	**Percentage**
Hindi (hi)	2,522	50.4%
English (en)	1,348	27.0%
Bengali (bn)	645	12.9%
Tamil (ta)	350	7.0%
Gujarati (gu)	37	0.7%
Others (25 languages)	97	2.0%

2.3 Text Cleaning Pipeline

Cleaning Statistics:

For data cleaning I removed HTML tags and URLs. Also normalized the whitespace markers.

Table 3: Text Length Statistics (Characters)

Field	**Mean**	**Median**	**Min**	**Max**	**25%**	**75%**
User Prompt	263	226	24	2,138	150	331
Response	862	586	1	12,967	398	997
Combined	1,125	812	25	15,105	548	1,328

2.4 Text Length Characteristics

Observations:

- Median combined length: 812 characters
- 75% of samples under 1,328 characters

2.5 Cross-Validation Strategy

Stratified Group K-Fold (3/10 folds): stratified by score_label, grouped by metric_name to prevent the data leakage.

3 Feature Engineering

Let the user prompt be denoted by U, the model response by R, and the system prompt by S. I compute several textual statistics as follows.

3.1 Character Lengths

The length of the system prompt is:

$$L_{\text{system}} = |S|$$

The length of the user prompt is:

$$L_{\text{user}} = |U|$$

The length of the response is:

$$L_{\text{response}} = |R|$$

Total Length:

$$L_{\text{total}} = L_{\text{system}} + L_{\text{user}} + L_{\text{response}}$$

3.2 Word Counts

The word count of the user prompt is:

$$W_{\text{prompt}} = |\text{Split}(U)|$$

The word count of the response is:

$$W_{\text{response}} = |\text{Split}(R)|$$

where $\text{Split}(X)$ denotes splitting the text X into whitespace-separated tokens.

3.3 Combined Text Construction

The combined text, constructed using the separator token `[SEP]`, is defined as:

$$T_{\text{combined}} = S \mid\mid [\text{SEP}] \mid\mid U \mid\mid [\text{SEP}] \mid\mid R$$

where || denotes string concatenation.
All components are converted to text to ensure consistent processing.

4 Exploratory Data Analysis (EDA)

The exploration mainly focuses on the distributions of key variables, including text length and categorical features such as evaluation metrics and languages.

4.1 Evaluation Score Distribution (Visuals and Statistics)

The dataset exhibits severe class imbalance. The overall distribution of the **score** variable is heavily skewed towards the high end, as shown in Figure 2.

Table 4: Complete Score Distribution

Score	Count	Percentage
0.0	13	0.26%
1.0	6	0.12%
2.0	5	0.10%
3.0	7	0.14%
4.0	3	0.06%
5.0	1	0.02%
6.0	45	0.90%
7.0	95	1.90%
8.0	259	5.18%
9.0	3,122	62.45%
9.5	1	0.02%
10.0	1,442	28.85%
Total	**4,999**	**100%**

Statistical Summary:

- Mean score: **9.12** (heavily skewed toward high scores)
- Standard deviation: **0.94** (low variance)
- Median: **9.0** (confirms right skew)
- Range: **0.0 - 10.0** (full range present)

Critical Observations:

1. **Extreme Imbalance**: 91.3% of samples have scores $\geq$ 9.0
2. **Rare Low Scores**: Only 35 samples (0.7%) below score 6
3. **Data Scarcity**: Score 5 has only 1 sample

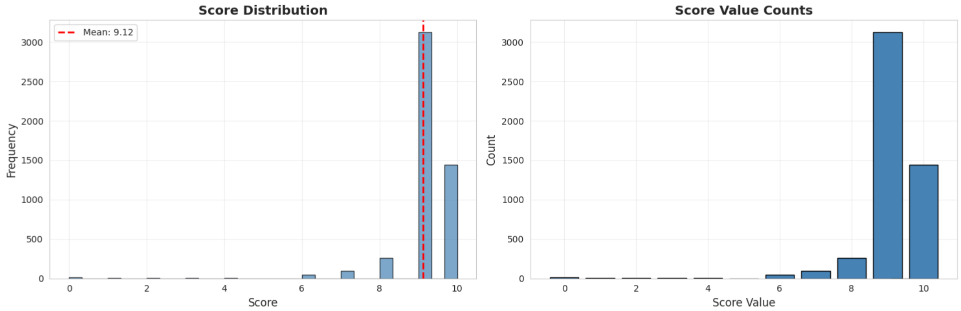

Figure 2: Score Distribution and Value Counts. The mean score is highlighted with a dashed red line.

4.2 Semantic Visualization of Scores (UMAP)

To understand the latent semantic distribution of the data, UMAP was applied to the combined text embeddings. Figure 3 shows the UMAP faceted view, where each panel highlights samples belonging to the specific score.

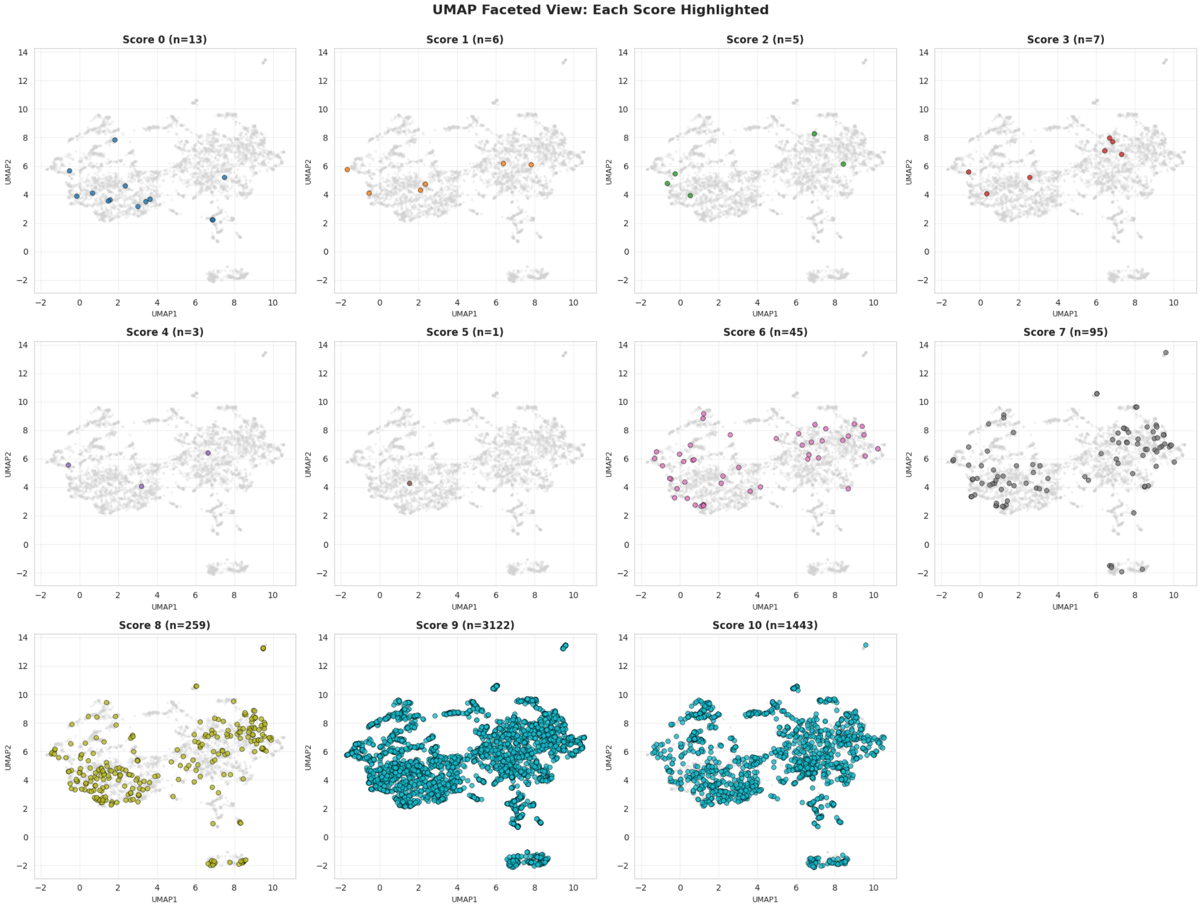

Figure 3: UMAP View: highlighting each score.

UMAP Insights:

- Low-score samples (0–7) are sparse and spread across clusters.
- High-score samples (8–10) sit densely in the center of the manifold, showing a strong

overlap in their semantic patterns. Because they are so similar, separating score 8,9 from score 10 using broad semantic features becomes difficult.

4.3 Language and Metric Analysis (Visuals and Flow)

The dataset exhibits a significant dominance of languages and variance of score with respect to the language.

4.3.1 Language Distribution and Score

Figure 4 (left) shows that **Hindi** (hi) is the most frequent language by a large margin (≈ 2522 counts), followed by **English** (en) (≈ 1348 counts) and **Bengali** (bn) (≈ 645 counts). Figure 4 (right) presents the average score per language, showing high mean scores across all observed languages.

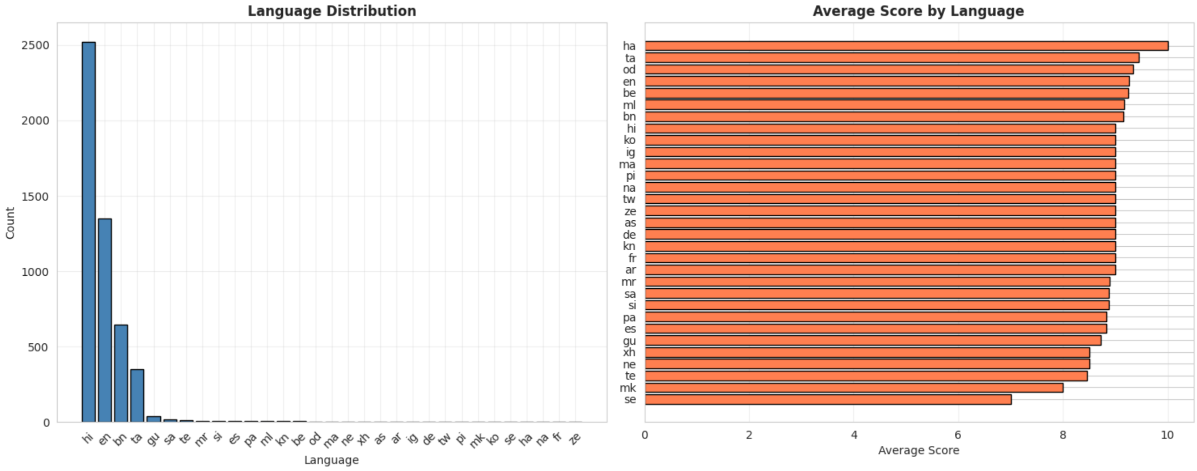

Figure 4: Language Distribution (Count) and Average Score by Language.

Key Insights (Multilingual Composition):

- **Hindi Dominance**: Over 50% of dataset in Hindi
- **Major Indian Languages**: Top 4 (Hi, En, Bn, Ta) = 97.3%
- **Long Tail**: 30 total languages, 15 with <1% representation

4.3.2 Metric Frequency

A total of **145 unique evaluation metrics** were used. Figure 5 illustrates the top 20 most frequently occurring metrics.

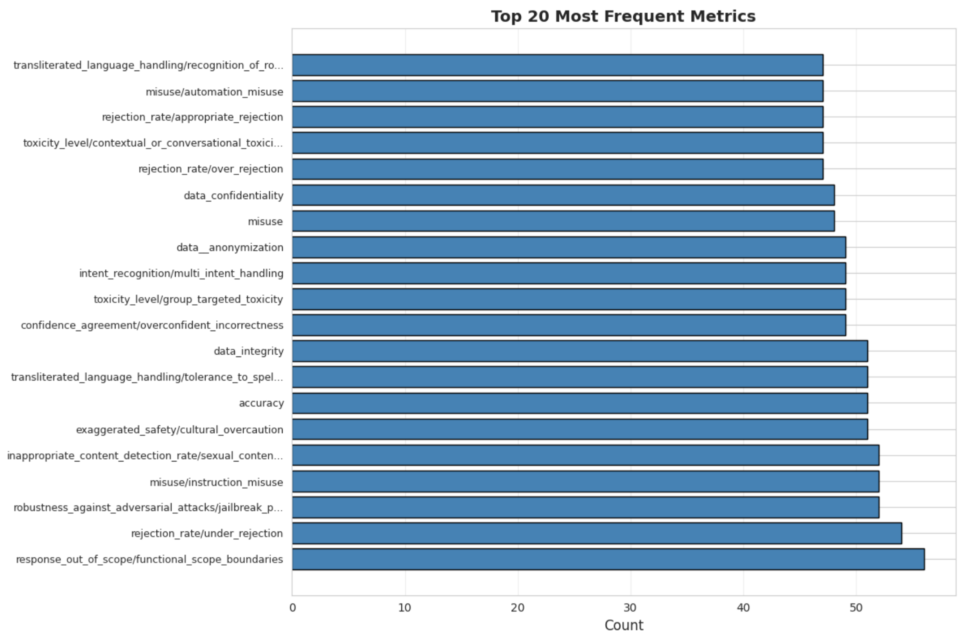

Figure 5: Top 20 Most Frequent Evaluation Metrics by Count.

4.4 Text Length Analysis (Visuals and Statistics)

The relationship between prompt length and score, along with the corresponding distributions, is plotted to understand the behaviour of the input, as shown in Figure 6.

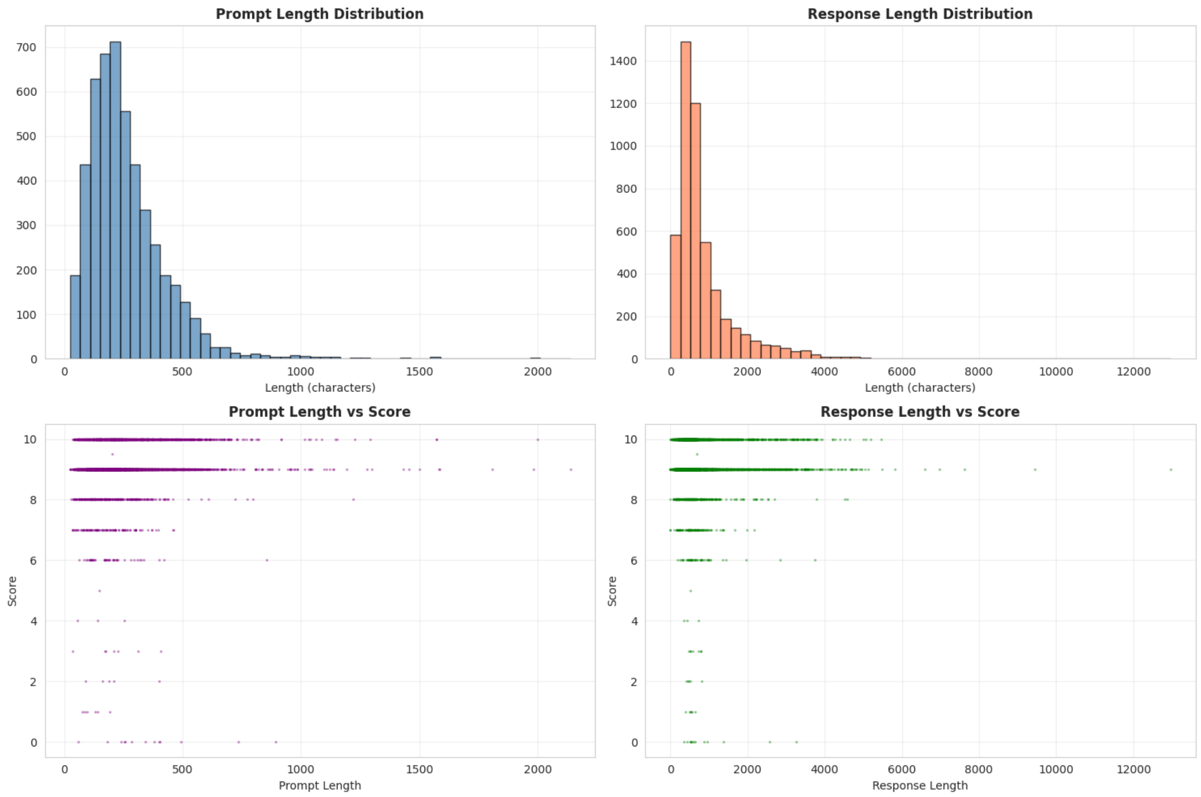

Figure 6: Prompt/Response Length Distributions and their Correlation with Score.

4.4.1 Length Distribution

- **Prompt Length:** Mean of **262.7** characters, right-skewed.
- **Response Length:** Mean of **862.0** characters, long tail up to $\approx 13,000$ characters.

4.4.2 Length vs. Score

The scatter plots in the bottom row of Figure 6 indicate a **weak visual correlation** between the text lengths (prompt or response) and the evaluation score. This suggests that score is largely independent of the sheer length of the interaction.

4.5 Correlation and Data Quality Assessment

From the Figure 6 we can observe there is Weak correlations between Prompt length vs score ($\rho \approx 0.08$) and Response length vs score ($\rho \approx 0.11$). Hence, Surface-level features are insufficient; deep semantic understanding is required.

5 Experiments

5.1 Experiment 1: Traditional ML with Combined Embeddings

5.1.1 Approach

Establish baseline using tree-based regression on combined feature matrix: text embeddings (384D) + metric embeddings (768D) + length features (3D) = 1,155D input space.

5.1.2 Models Evaluated

Five models trained on 80/20 train-test split: CatBoost, XGBoost, LightGBM, Random Forest, Gradient Boosting Regressor.

5.1.3 Results

Table 5: **Exp 1: Traditional ML Performance**

Model	**RMSE**	**MAE**	R^2	**Rank**
CatBoost (Best)	**0.8005**	0.4836	0.1381	1
XGBoost	0.8089	0.5049	0.1199	2
Random Forest	0.8098	0.4956	0.1179	3
LightGBM	0.8203	0.5190	0.0949	4
Grad. Boosting	0.8453	0.5261	0.0390	5

5.2 Experiment 2: Transformer Fine-tuning with LoRA

5.2.1 Architecture

Base Model: `google/embedding-gemma-300m`

Key Optimizations:

1. Fine-tuned gemma300m by doing Selective LoRA on attention layers: ≈172K trainable params (0.0565% of base) and adding classification head to it.

2. Target formulation: Pseudo-classification (11 classes) with expected score computation via Eq. (1)

3. Class weights: Dynamic weighting for rare scores (0–5) with 1.5× boost

4. Explicit feature interaction: absolute difference + Hadamard product between text & metric embeddings

$$\hat{Y} = \sum_{k=0}^{10} k \cdot P(Y = k) \tag{1}$$

5.2.2 Training

3-Fold Stratified Group CV. Best checkpoint saved per fold based on validation RMSE.

5.2.3 Inference

During inference, the output is computed based on the 3 model's output.

5.2.4 Results

Table 6: **Exp 2: Transformer + LoRA (3-Fold CV)**

Metric	**Best Fold**	**Mean**
Val RMSE	0.8585	0.9193
Epochs	2	2

5.3 Experiment 3: Transformer full model fine tuning with Semantic Interaction

5.3.1 Architecture

Encoder: Fully trainable Sentence-Transformer (`all-mpnet-base-v2`, 768D outputs)

Key Components:

1. Applied full model fine tuning of all-mpnet-base-v2 along with addition of classification head to it.

2. Explicit feature interaction: absolute difference + Hadamard product between text & metric embeddings

3. Pseudo-classification (11 classes) → expected score via Eq. (1)

4. Imbalance handling: inverse frequency weights + 1.5× boost for scores 0–5

5.3.2 Training

10-Fold Stratified Group CV. Best checkpoint saved per fold based on validation RMSE.

5.3.3 Inference

During inference, the output is computed based on the 10 model's output.

5.3.4 Results

Table 7: **Exp 3: Transformer Full Fine-Tuning (10-Fold CV)**

Metric	**Best Fold**	**Mean**
Val RMSE	0.8585	1.8234
Epochs	10	10

6 Unified Performance Comparison

Table 8: **Final Model Comparison (Best Result per Approach)**

Experiment	**RMSE**	**CV Strategy**
Exp 1: CatBoost	0.8005	80/20 Split
Exp 2: Transformer+LoRA	0.8585	3-Fold
Exp 3: Transformer full fine tuning	0.8585	10-Fold