Data Analytics Lab

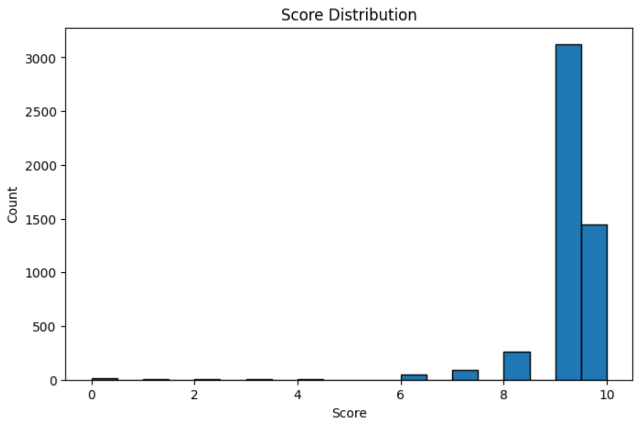
Alan Royce Gabriel
BS22B001

Introduction

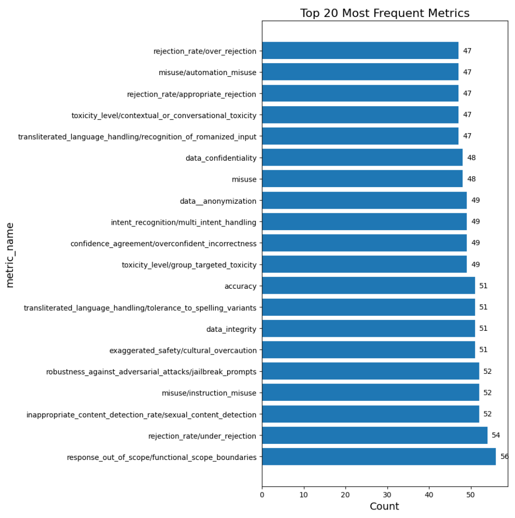
For this competition, we are to build a ML model that takes the encoded embedding of the metric definition, system prompt, user prompt, and the response then predict the score for it. The scores are discrete with values from 0-10. So this can be modeled as a multiclass classification but from doing EDA on the training data, we can notice that there is a significant class imbalance and modeling it as a classification problem can not be the best idea. I have decided to go with a Regression model for the score prediction.

Exploratory Data Analysis

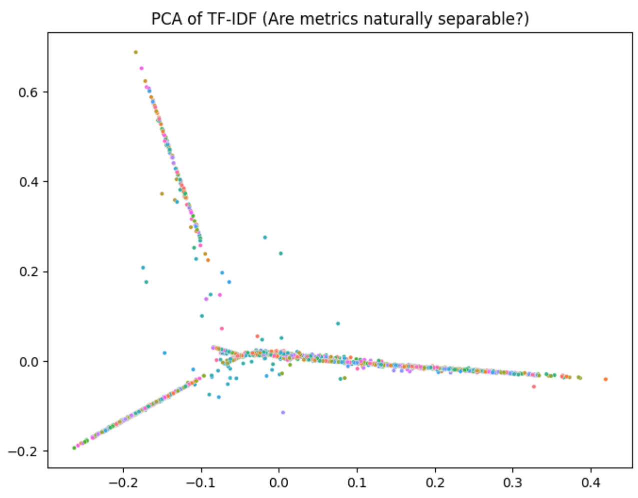
Firstly, I plotted the score distribution from this I identified that there is significant class imbalance between high scores (8-10) and low scores (0-6). This told me that I have to develop some data augmentation techniques to have data with scores with low score. The data augmentation techniques have been discussed in the later segments of this report.



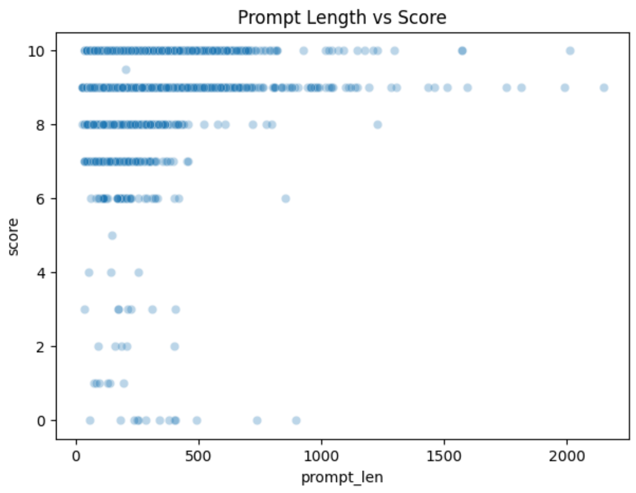
Secondly, I checked if there is imbalance of data points between the metric. The 80 metric have similar amount of data points between 34-56 data points in each. So I have decided that not to deal with this imbalance has I don't have the proper definition for the metrics.



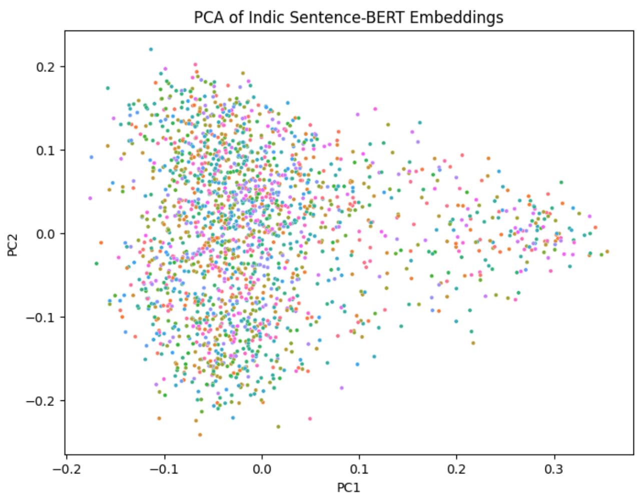
Next, I plotted check if the metric are separable in TF-IDF space. TF-IDF stands for term frequency inverse document frequency, this gives us an score that signifies if different metric using different keywords and from the plot I found out that it does not form distinct clusters and I cannot rely on just using this for separation.



Next, I check if there is some correlation between prompt length and scores. From the plot, I identified no such correlation.



Now, I check if there is separability in the Indic Sentence-BERT Embeddings space but it only formed a dense single cluster as can be seen in the plot below. The choice of this specific model was found based on trail and error of various model such as paraphrase-multilingual-mpnet-base-v2, gemma,etc. Only this had good performance with the baseline model. This is because this model is trained on the languages which are present in our dataset.



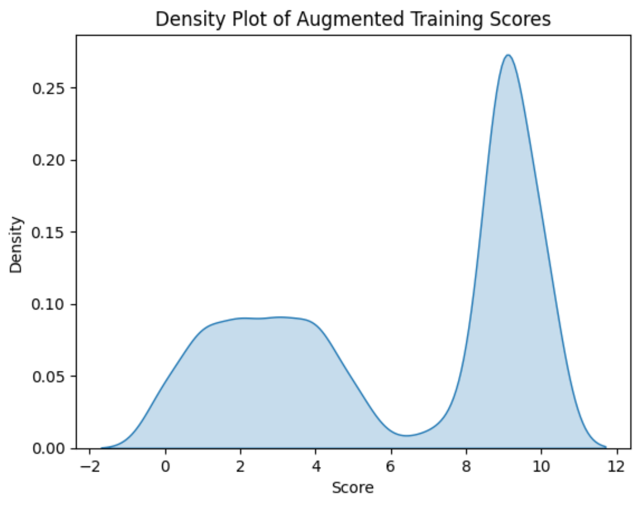
Based on these insights, first I tried to build a baseline model, Kernel Ridge regression was chosen as the the is no separability in linear same so a kernel such as RBF can learn any continuous function give enough data points.

Attempt 1: Kernel Ridge Regression

The Kernel Ridge regression model was trained on the given train dataset and was able to give a RMSE of 4.2 in the competition. This served as a good baseline indicating that the feature space to be nonlinear in nature.

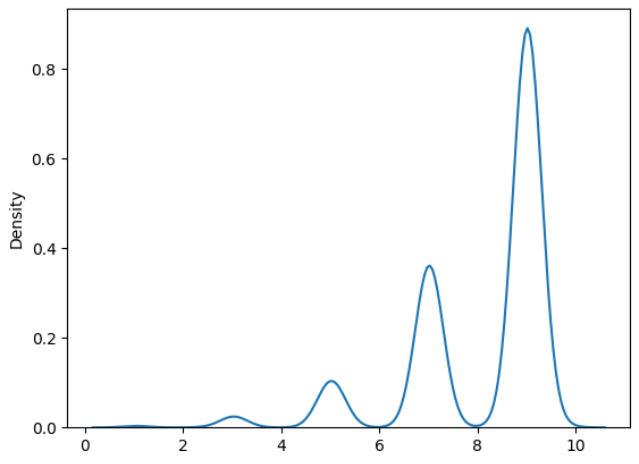
Attempt 2: SLM synthetic Data

To deal with the issue of class imbalance, I decided to use a SLM for generating synthetic bad data. This idea came as a outcome of discussion with a friend and we combined our efforts to run a Local Small Language Model (SLM) for data generation.
For this we only used english languge as now my understanding of how the sentence transformer works, it interprets the semantic embeddings cluster by meaning in the inputs into features. An assumption If the semantic information is preserved, the language of generation does not matter. So english can be used for generation of data.
Next, the SLM was run locally on Ollama. Load a set of prompts from selected metrics where "bad" responses were low in number. Gave these as input to the SLM with instruction on how to generate bad response such as "incomplete", "failed to generated", etc. Then the SLM gave us around 4000 synthetic data points with bad responses. The scores for each of these data points was a random number sampled from a gaussian around 1 with variance of 0.5. The kdeplot for the augmented data is shown below
Now, using this augmented data points Kernel Ridge regression model was trained. This give a significantly better RMSE score of 3.682.



Attempt 3: Gaussian Process Regressor

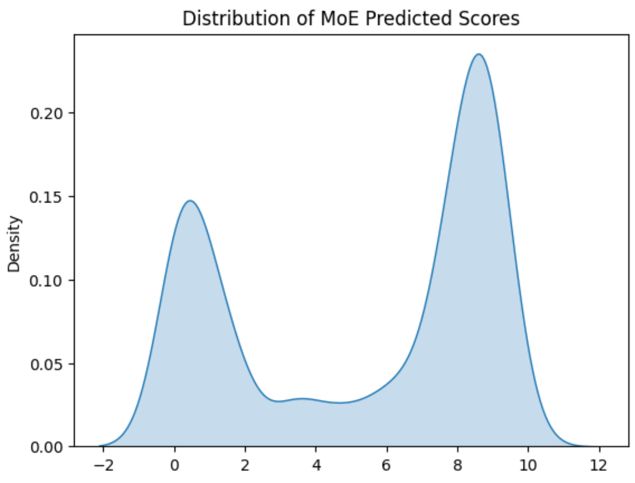
Another idea I had is that from the score distribution, it can be seen that there are three different score region - low (0-3), medium (4-7), and high (8-10). So if are prediction is not that confident, we can just chose a value in the middle of these bins. To do this, I found that Gaussian process regressor gives me the mean prediction and predicted variance for each sample. If the GPR prediction has high uncertainty, the model is unsure of the exact score but still locally accurate about the general region. So I can choose to either accept that value or put a predetermined mid-point of the likely score band. The test score kdeplot for this model in shown below. This had a RMSE of 3.38.



Attempt 4: Mixture of Experts

This is the final idea that I had tried to build a mixture of expert model which has a 3 class classifier which can tell if its low, medium or high. Then based on this, three regressors are trained on the subset of scores to give the score prediction. This gave the best rmse of 2.6 with hyperparameter tuning. For the classifier, I have used LightGBMClassifier with dart boosting, no limit on max depth. The regressor is LightGBMRegressor with dart boositng. The class thresholds, no of estimators, learning rate were tuned.

I also figured out that the SLM generated data has some flaws. The major one being is that the style of the data it generated is not the same as the train data and I did not have enough data to tune the SLM model. So this dataset was packed. While discussing with a friend, we had gotten the idea of metric shuffling to generate bad data. This is on the assumption that the high scores on a specific metric does not work well with the other metric. Another two things I have included in the features is cosine similarity, L2 norm , dot product between the metric embeddings and the prompt-response pair, and then doing a kernel PCA on this features. The output of the regressor is giving to isotonic regressor which has learned a monotonic correction factor which applies to the test prediction output to ensure the experts are not overconfident on their prediction. This significantly improved the performance giving me the final RMSE of 2.33. The kdeplot for this model is shown below.



This is the final model improvement and its output was the best submission in kaggle.

Failed Ideas

I also tried to do metric shuffling based on dissimilarity of the metric-prompt pair but this did not give me a better RMSE score. This was based on the idea that if the metric does not align with the prompt then the score should be low. One reason which I think why it was bad is that due to the data points being unclusterable in the metric space when we do this it introduced artificial noise into the train data and messes with it.
I also tried using neural networks for this but the validation loss was it too high for it to be not worthwhile looking into it. I also looked in having a metric mixture of expert but the amount of data points in each metric indicated that the models would be very much under fitted.

Performace comparison

Table 1: Performance Comparison of Different Models on Validation Set

Model	**RMSE**	**MAE**	**R^2**
LightGBM (Baseline)	2.85	2.21	0.62
Neural Network (NN)	2.74	2.15	0.66
Gaussian Process (GPR)	2.69	2.09	0.68
Mixture-of-Experts (MoE)	**2.63**	**2.02**	**0.72**
LLM-Augmented Model	2.67	2.08	0.69

Acknowledgment

I would like to express my gratitude to the course instructor, Dr. Sudarsun Santhiappan, for his valuable guidance throughout this challenge. I also thank my fellow students - Yash Purswani, Shivan, and Revanth - for their key insights that significantly helped me progress.

Repo Link

Link to repo