Another model family we used was tree-based models. Random forest ultimately became our primary model, since it naturally handled high-dimensional features, required minimal preprocessing, and consistently outperformed the other model families on validation performance while remaining relatively robust and stable.

Random forest was not optimized using gradient-based optimizers like SGD or Adam. Instead, each tree was grown greedily. At each split, the algorithm evaluated the threshold by selecting a subset of text features and chose the split that maximized information gain according to a chosen criterion (Gini, entropy, or log loss). Regularization was implemented via hyperparameters. We regularized the model by limiting the maximum depth of each tree, enforcing a minimum number of samples per leaf, and controlling the number of features considered at each split. The number of trees controlled the strength of the ensemble. For instance, more trees typically reduced variance but increased computational cost.

To evaluate models fairly and tune hyperparameters without leaking information, we used a train/validation/test split with proportions 50% / 25% / 25%, applied at the student level using the student_id column, making sure that all three responses from a given student were contained in exactly one of the three splits to maintain class balance. After that, a manual grid search was performed over a reasonably rich hyperparameter grid. We varied the number of trees $\in \{50, 100, 200, 300, 400, 500\}$, the maximum depth $\in \{1, 2, \ldots, 10\}$, the minimum samples per leaf $\in \{1, 2, \ldots, 10\}$, the feature subsampling strategy $\in$ {"sqrt", 0.3, 0.5, 0.7}, and the split criterion $\in$ {"gini", "entropy", "log_loss"}. For each combination, we trained the model on the training split and computed accuracy on both the validation and the test splits. The best model was the one with the highest validation accuracy. This grid search revealed that very deep trees with very small leaves achieved almost perfect training accuracy but worse validation and test performance (overfitting), while moderate depths and more samples in leaves produced better generalization. Crucially, we tuned all three model families before comparing them, rather than tuning only the winning model, so the final choice of random forest was fair.

The first evaluation metric was accuracy. It was simple to interpret and appropriate for this task because the three classes (ChatGPT, Claude, Gemini) were balanced. However, accuracy alone could hide systematic misclassifications, so we examined the confusion matrices for the train, validation, and test splits. The confusion matrices showed that for each true model label, how predictions were distributed across the three classes, which allowed us to see whether the classifier systematically confused, say, Gemini with ChatGPT more often than with Claude. These matrices also showed more detailed metrics such as recall, precision, and F1. They provided evidence that

model performance was reasonably balanced across classes rather than being driven by one easy-to-classify class (ChatGPT).