- **KNN method**

  The choice of KNN is mainly for simplicity and serve as a base line. We can quickly modify KNN based on project_baseline.py to test our preprocessing and have our first working model.

  There is no optimizer implementation for KNN model since it’s too simple to apply optimizer. We tried a weighted voting version of KNN that assign each “neighbor” with a weight of 1/d, where d is the calculated distance. The list of KNN hyperparameter tuned are 1) words list THRESHOLD; 2) k; 3) distance metric; 4) voting method

  We used grid search to search for the combination that produce maximum validation accuracy. The range of grid search for words list THRESHOLD was [1, 15]. We start with THRESHOLD = 200 but the result validation accuracy is very low (~ 50%). Consider knn typically perform suboptimal with high dimensional data, we manually increment THRESHOLD from 1 and notice the validation accuracy approximately drops under 60% after threshold = 15. The range of grid search for k was [1, 50], since validation accuracy drops under 60% after k = 50. The choice of distance metric was either Euclidean or Manhattan. Lastly search either non-weighted or weighted voting inference.

  The best combination found was THRESHOLD = 1; k = 26; distance = Euclidean; voting = non-weighted, result in validation accuracy = 0.657. Technically speaking, only k, distance and voting are KNN’s hyperparameter. THRESHOLD is pre-processing parameter, but we specifically tuned it because KNN is sensitive to dimension.

  Since three class were balanced in dataset, and all mistakes have same consequences, we choice to mainly rely on accuracy. The final test accuracy with best combination found was 0.657. ChatGPT have the highest precision (0.75), recall (0.77) and f1-score (0.76), indicate ChatGPT is the easiest to predict. Claude is moderately accurate indicate by recall = 0.65, 65% of actual Claude were found. Gemini is the hardest class with recall = 0.55.

  Numpy and pandas were imported for basic data manipulation, sklearn.metrics was imported for evaluation matrices. Two distance metrics were implemented separately and called by main knn_predict() function wrapped in an if chunk to determine which distance to use. Main body of hyperparameter tuning was validation set inference within four nested for loop, correspond to four hyperparameter.

- **Result**

Across all three modes, ChatGPT is the easiest to predict, conserve the highest precision, recall and f1-score across all three model. Claude and Gemini were frequently miss classified as each other indicated by the relatively higher [Claude, Gemini] and [Gemini, Claude] entry in confusion matrix, consist with our exploratory data analysis.

// TODO compare model family's evaluation/confusion metrix

After evaluating KNN, Logistic Regression, and Random Forest, we selected Random Forest as our final model, achieved accuracy of ???% on test set. Although Logistic Regression achieved accuracy comparable to Random Forest on the validation and test sets, its performance was highly inconsistent across splits—Logistic Regression reached ~80% accuracy on the training set, but only ~70% on validation and test, indicating clear overfitting. In contrast, Random Forest showed the smallest variance among training, validation, and test accuracy (consistently ~70–75%), which suggests that it generalizes more reliably and is less sensitive to fluctuations in data distribution. Compared to KNN, Random Forest achieved ~5-10% higher validation accuracy and was less affected by noisy or high-dimensional features—an important advantage given the text-based nature of our inputs. Additionally, Random Forest provides richer hyperparameter tuning potential: even with six hyperparameters explored in our tuning, we limited the search range due to computational constraints. With a more exhaustive search (e.g., tuning tree depth, number of estimators, feature subsampling strategy), Random Forest is expected to improve further. Overall, Random Forest offers the best balance of predictive performance, model stability, and tuning flexibility, making it a stronger and more reliable final choice than both KNN and Logistic Regression.