Model reporting

To evaluate the effectiveness of the K-Nearest Neighbor (KNN) classifier, I selected the Reddit r/Korea Subreddit Dataset from Kaggle (https://www.kaggle.com/datasets/bwandowando/reddit-rkorea-subreddit-dataset).

The target for prediction is the is_submitter column, a binary variable indicating if the comment's author is the thread's original submitter:

- True (1): The author is the thread's original submitter.
- False (0): The author is not the thread's submitter.

The objective is to predict whether a comment's author is the thread submitter based on features like comment content, score, and other metadata.

Implementation part

✓ The implementation begins by importing necessary libraries for data analysis, visualization, preprocessing, and model evaluation.

✓ During data preprocessing, I analyzed the distinguished column, which contained 64 non-null values labeled as "moderator" and 23,649 missing values. Given the small number of non-null entries, the column was removed from the dataset.

✓ In the data preprocessing step, I counted the unique values in the is_submitter and stickied columns, which both contained only boolean values (True and False). To prepare the data for modeling, I mapped these boolean values to integers (1 for True and 0 for False) and converted the columns to integer format.

✓ During data analysis, I evaluated the dataset for low and high cardinality features. I did not find any low cardinality columns. As a result, I labeled all categorical (objective) columns as numerical values. Afterward, I converted all columns to integer type for consistency and model compatibility.

✓ Before training the model, I scaled the feature datasets by using StandardScaler(). This was done after splitting the data into training, validation, and test sets to ensure that the scaling is applied consistently across all sets.

✓ The KNeighborsClassifier(n_neighbors=5) initializes the K-Nearest Neighbors (KNN) model with 5 neighbors. The fit(x_train, y_train) method trains the KNN model using the training data (x_train) and corresponding target values (y_train).

The accuracy of the model is indicating that the model correctly predicted 95% of the test data.

KFold cross-validation with 5 splits is applied to evaluate the KNN model's performance more robustly. The data is scaled using StandardScaler() before performing the cross-validation. The cross_val_score() function calculates the mean squared error (MSE) for each fold. The results of the cross-validation show the following MSE values for each fold:

[0.945, 0.939, 0.943, 0.949, 0.949]

✓ These values indicate consistent performance across all folds, with the model achieving a relatively high accuracy in each validation split.

10-fold cross-validation is applied to the KNN model using the entire dataset (x and y). The cross_val_score() function computes the accuracy for each fold.

The cross-validation scores for each fold are:

[0.946, 0.942, 0.944, 0.946, 0.945, 0.946, 0.946, 0.944, 0.937, 0.911]

The mean accuracy across all folds is 0.94, indicating stable model performance, and the standard deviation of the scores is 0.01, suggesting low variability between the folds.