GRADIENT BOOSTING ALGORITHM

This code uses the **Gradient Boosting algorithm** to predict the PRODUCT_LENGTH of a product based on its PRODUCT_TYPE_ID. Here is a step-by-step explanation of the code:

- The necessary libraries are imported: pandas to work with dataframes, GradientBoostingRegressor from sklearn.ensemble to build the gradient boosting model, and mean_absolute_percentage_error from sklearn.metrics to evaluate the model's performance.

- The train and test data are loaded using pandas read_csv() function. The data is stored in pandas dataframes train_df and test_df, respectively.

- The train and test data are concatenated using pd.concat() function to form a full_df dataframe. The NaN values are replaced with 0 in the full_df dataframe using the fillna() method.

- The features are selected from the train and test data, and X_train, X_test, y_train, and y_test are created as separate dataframes using pandas.

- A GradientBoostingRegressor() object called gb is created, and the fit() method is used to train the model using the X_train and y_train data. The model is used to make predictions on the test data using the predict() method. The predicted values are stored in y_pred.

- The mean_absolute_percentage_error() function is used to calculate the model's accuracy by comparing the actual values (y_test) with the predicted values (y_pred). The accuracy is stored in the score variable.

- The predicted values are saved in a new pandas dataframe called sub_df, along with their corresponding PRODUCT_IDs.The index of the sub_df dataframe is set to PRODUCT_ID using the set_index() method.The data in sub_df is written to a new CSV file called 'Gradient Boosting Result Submission.csv' using the to_csv() method.

In terms of feature engineering, the code only uses the PRODUCT_TYPE_ID feature to predict the PRODUCT_LENGTH. No additional feature engineering is performed.