GSTN Analytics Hackathon

Developing a Predictive Model in GST

REPORT

TEAM ID - GSTN_435

MEMBERS - Adamya Gaur
Mohit Gupta
Harjot Singh
Gaurav Kumar Chaurasiya

Explanation of the key methodology and steps taken in

Model Development

1. Data Understanding and Preprocessing

Data Exploration

The first thing on our introductory tour of the dataset was to get familiar with its structure, distributions, and possible missing value issues. We then imported the data into a Pandas DataFrame and ran some summary statistics on every feature, where we could see the range, mean, and standard deviation on that variable; in addition, by plotting the distribution of the target variable, we looked for class imbalance issues. Importantly, we found that many of the features consisted of missing values. Particularly Column9 of where data was missing for over 93% of the feature. To look at correlations between features to see if there are some multicollinearity problems that could negatively affect the model. The size of the dataset is 785, 133 rows, and 24 columns - with the mix containing numerical and categorical data types. Summary statistics Different counts for each feature Column 3 Column 4 appears to have significantly fewer non-null entries, so possibly data quality issues The Target variable suffers from a class imbalance effect since this problem is dominated by entries belonging to one class, which can affect our model's training.

2. Data Preprocessing

Dropping Columns:
We dropped Column9. It had more than 93% missing data. To impute that much of missing data might result in bias and noise, so we decided to delete it completely.

Median Imputation: For columns having a moderate number of missing values, we made use of median imputation. Median is primarily outlier resistant and constitutes an effective mean that does not get dominated by the extreme value as well.

Standard Scaling: After handling missing values, we applied standard scaling to the remaining features. This step ensured that all variables were on a similar scale, facilitating better convergence during model training and improving the overall performance of our algorithms.

3. Model Selection:

In our analysis, we focused on selecting models that excel in handling structured/tabular data, particularly for the GST dataset. After evaluating several options (ML models, Neural Network, LLM etc), we identified three algorithms that demonstrate strong performance and adaptability: **XGBoost, CatBoost, and LightGBM.**

XGBoost is well-regarded for its robustness with both numerical and categorical data, as well as its ability to handle missing values internally. Its regularization capabilities (L1 and L2) help reduce overfitting, making it a suitable choice for this dataset.

CatBoost is particularly effective in managing categorical features and offers strong performance even in the presence of missing values and

LightGBM is optimized for speed and efficiency, making it suitable for large datasets while maintaining competitive accuracy.

To enhance our model performance further, we implemented a custom voting classifier that combines the predictions of these three models. The CustomVotingClassifier aggregates the outputs from individual models using **majority voting**.

The custom voting mechanism not only combines the strengths of each individual model but also enhances overall predictive accuracy.

4. Model Training Training Process

The data was split into training and validation sets to ensure the model's performance would generalize well to unseen data. We used an 80/20 split, where 80% of the data was used for training and 20% for validation. This split allowed the model to learn from a sufficient amount of data while also providing a validation set to evaluate its performance during training.

5.Hyperparameter tuning

Grid search and Optuna Usage

To optimize our model, we have used hyperparameter tuning using both GridSearchCV and Optuna. First, we have used GridSearchCV to search over various parameters including n_estimators, max_depth, learning_rate, subsample, and colsample_bytree. This was a grid search in which we specified values for each hyperparameter, and all these combinations were exhaustively evaluated for the best-performing configuration based on validation set performance.

We placed Optuna on top of this to further enhance our tuning process. Optuna is a hyperparameter optimization framework that employs a more efficient approach by using TPE in sampling the hyperparameters. This search in the hyperparameter space adapts and evolves, mainly focusing on promising regions of the hyperparameter space, thus speeding up the optimization.

6. Model Evaluation Validation and Testing

After tuning the model, We evaluated it on both the validation and test sets using a comprehensive set of metrics:

- **Accuracy:** The proportion of true results (both true positives and true negatives) among the total number of cases examined.

  $$\text{Accuracy} = \text{TP+TN} / \text{TP+TN+FP+FN}$$

- **Precision:** The ratio of true positive predictions to the total predicted positives, indicating the accuracy of positive predictions.

  $$\text{Precision} = \text{TP} / \text{TP+FP}$$

- **Recall:** Also known as sensitivity, it measures the proportion of actual positives that were correctly identified.

  $$\text{Recall} = \text{TP} / \text{TP+FN}$$

- **F1 Score:** The harmonic mean of precision and recall, providing a single metric that balances both concerns.

  $$F1 = 2 \times \text{Precision} \times \text{Recall} / \text{Precision+Recall}$$

- **ROC AUC:** The area under the Receiver Operating Characteristic curve, which measures the model's ability to distinguish between classes across various threshold settings.
- **Balanced Accuracy:** The average of recall obtained on each class, useful in cases of imbalanced datasets.

  $$\text{Balanced Accuracy} = (\text{TPR+TNR}) / 2$$

- **Log Loss:** A performance metric that evaluates the probabilities of the predicted classes, penalizing incorrect classifications.
- **Confusion Matrix:** A table that summarizes the performance of a classification model, showing true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).
- **Classification Report:** A detailed report providing precision, recall, F1 score, and support for each class, giving insights into model performance for multi-class problems.

Final Model Selection and Submission

Final Training

Once we decided on the suitable hyperparameters, we utilized all the training data for retraining of the model. This, in turn, provided it with the capability to utilize all available data, making it learn the best-fit relationships between the features and the target variable. Using the model trained finally, we used the model for predictions over the test set.

Submission Preparation

We then formatted the output into the required format for the final submission to the competition by merging the predictions with the original ID column of the test set to trace and ensure accuracy. Once saved as a CSV file, the predictions were ready for submission. Further checks were run to make sure all rows were included, and the values predicted were within the expected range.

Summary

We begin our exploration of the dataset to understand the structure, the distributions, and any issues with respect to missing values. After addressing the missing data strategies like dropping columns that are highly missing and cases with median imputation, we have standardised the data in order to improve the performance of the model.

Baseline models with Logistic Regression and Decision Trees were run first, and the more advanced models, XGBoost, CatBoost, and LGBM were used because they performed well with structured data and also had mechanisms in place for handling missing values. We also implemented a custom voting classifier that accurately combined predictions from these models.

Further, we ran the hyperparameter tuning using GridSearchCV and further optimized this with Optuna for optimization which is really more efficient. To ensure the model performs a balance, I used multiple evaluation metrics: Mean Squared Error (MSE), accuracy, precision, recall, F1 score, ROC AUC, balanced accuracy, log loss, confusion matrix, and classification report.

With the optimal hyperparameters derived, we retrained the final model on the entire training dataset to maximize learning, and we generated predictions on the test set. For submission, we formatted our output, collapsed predictions against their original ID column for traceability, and ensured the accuracy and completeness of the submission file. The approach was hopefully thorough to deliver a robust predictive model ready for evaluation.

Evaluation of the model using the provided metrics

Model Evaluation Results

The model was evaluated on three different datasets: training, validation, and testing, yielding robust performance across all metrics.

Training Results

- **Accuracy:** The model achieved an impressive accuracy of approximately **98.19%**, indicating that it correctly classified the majority of training instances.
- **Precision:** The precision was around **86.92%**, demonstrating the model's ability to accurately identify true positive cases (class 1) among its positive predictions.
- **Recall:** With a recall score of approximately 95.16%, the model effectively captured a high percentage of actual positive cases, showcasing its strength in detecting class 1 instances.
- **F1 Score:** The F1 score was approximately 90.85%, reflecting a solid balance between precision and recall, indicating the model's overall effectiveness.
- **ROC AUC Score:** The ROC AUC score of about 96.83% suggests an excellent capability of the model to distinguish between classes across different thresholds.
- **Balanced Accuracy:** The balanced accuracy matched the ROC AUC score at 96.83%, confirming that the model performed well even in the presence of class imbalance.
- **Log Loss:** A low log loss of approximately 0.042 indicates that the predicted probabilities were quite accurate.

Classification Report: The classification report showed:

- Class 0: Precision: 0.99, Recall: 0.99, F1 Score: 0.99 (Support: 568,880)
- Class 1: Precision: 0.87, Recall: 0.95, F1 Score: 0.91 (Support: 59,226)

Overall, the model demonstrated strong performance with high accuracy and well-balanced precision and recall.

Validation Results

- **Accuracy:** The validation accuracy was approximately 97.87%, confirming the model's strong performance on unseen data.
- **Precision:** Precision for the validation set was around 85.46%, indicating a solid ability to identify true positives.
- **Recall:** The recall score of approximately 93.25% showed the model's effectiveness in capturing positive instances.
- **F1 Score:** An F1 score of around 89.19% reflected a good balance between precision and recall.

- **ROC AUC Score:** The ROC AUC score was approximately 95.80%, reinforcing the model's capability to differentiate between classes.
- **Balanced Accuracy:** The balanced accuracy also stood at 95.80%, ensuring performance consistency across classes.
- **Log Loss:** The log loss value of about 0.049 indicates a low prediction error.

Classification Report: The classification report for validation indicated:

- Class 0: Precision: 0.99, Recall: 0.98, F1 Score: 0.99 (Support: 142,220)
- Class 1: Precision: 0.85, Recall: 0.93, F1 Score: 0.89 (Support: 14,807)

Testing Results

The model's performance was evaluated using various metrics, which provided insights into its predictive accuracy and reliability.

- **Accuracy:** The model achieved an accuracy of approximately 97.86%, indicating that it correctly classified the vast majority of instances in the test set.
- **Precision:** With a precision of about 85.04%, the model demonstrated a strong ability to identify true positive cases (class 1) among the predicted positives, though some false positives were present.
- **Recall:** The recall score of approximately 93.77% signifies that the model successfully captured a high percentage of actual positive cases, highlighting its effectiveness in detecting class 1 instances.
- **F1 Score:** The F1 score, which balances precision and recall, was calculated at around 89.19%. This score reflects the model's overall effectiveness in classifying the positive class while considering both false positives and false negatives.
- **ROC AUC Score:** The ROC AUC score of approximately 96.03% indicates an excellent ability of the model to distinguish between the two classes across various thresholds, reinforcing its classification capabilities.
- **Balanced Accuracy:** The balanced accuracy was also reported at 96.03%, ensuring that the model performed well even in the presence of class imbalance.
- **Log Loss:** The log loss value of about 0.049 demonstrates a low level of prediction error, suggesting that the model's predicted probabilities were quite accurate.

Confusion Matrix:
The confusion matrix showed the following results:

```
[232962   4072
 1537    23141]
```

This matrix indicates that the model correctly classified 232,962 true negatives and 23,141 true positives. However, it also misclassified 4,072 negatives as positives and 1,537 positives as negatives, highlighting areas for potential improvement.

Classification Report:
The classification report provided a detailed breakdown of precision, recall, and F1 scores for each class:

- For class 0, the precision was 0.99, recall was 0.98, and F1 score was 0.99, based on a support of 237,034 instances.
- For class 1, the precision was 0.85, recall was 0.94, and F1 score was 0.89, with a support of 24,678 instances.

Here's a table summarizing the model evaluation results across training, validation, and testing datasets:

Metric	Training Results	Validation Results	Testing Results
Accuracy	98.19%	97.87%	97.86%
Precision	86.92%	85.46%	85.04%
Recall	95.16%	93.25%	93.77%
F1 Score	90.85%	89.19%	89.19%
ROC AUC Score	96.83%	95.80%	96.03%
Balanced Accuracy	96.83%	95.80%	96.03%
Log Loss	0.042	0.049	0.049

Confusion Matrices

Dataset	True Negatives	False Positives	False Negatives	True Positives
Training	560,397	8,483	2,867	56,359
Validation	139,871	2,349	999	13,808
Testing	232,962	4,072	1,537	23,141

Classification Reports

Class	Precision	Recall	F1 Score	Support
Class 0				
Training	0.99	0.99	0.99	568,880
Validation	0.99	0.98	0.99	142,220

Testing	0.99	0.98	0.99	237,034
Class 1				
Training	0.87	0.95	0.91	59,226
Validation	0.85	0.93	0.89	14,807
Testing	0.85	0.94	0.89	24,678

This table effectively summarizes the performance metrics, confusion matrices, and classification reports for the model across different datasets.

Here is the plot representing above data in bar form.