Detailed Report on GST Hackathon Model

Team ID: GSTN_697

This report details the development, evaluation, and performance of an XGBoost model for classification within the GST Hackathon. The primary goal was to achieve high prediction accuracy and recall on the provided dataset.

1. Data Overview

The dataset contains **785,133 rows and 22 columns**, consisting of both numerical and categorical data. Specifically, **13 columns contain floating-point values** (float64), and **9 columns have integer values** (int64). The dataset also contains missing values in several key columns, which require careful data cleaning and preprocessing for any modeling tasks.

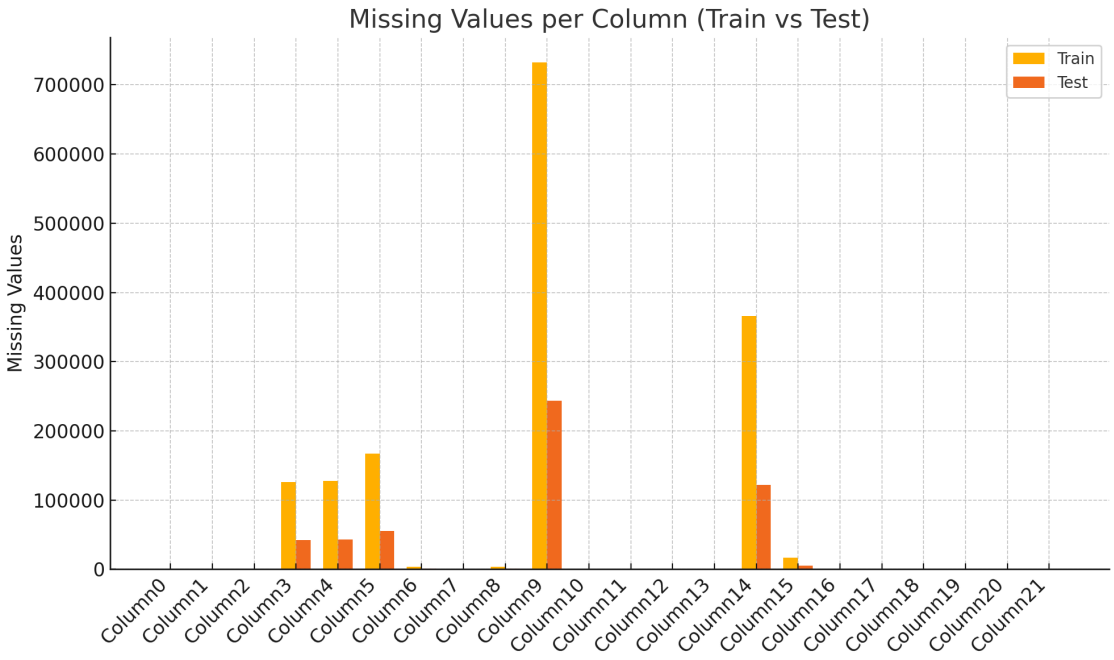
2. Data Cleaning and Preprocessing:

Missing Values Handling

The dataset contained missing values in 5 columns (Column3, Column4, Column5, Column9, and Column14). Missing values were imputed using the median of each respective column to mitigate the impact of potential outliers.

- **Column9**: 732,137 missing entries (93.25%) – highest.
- **Column14**: 365,703 missing entries (46.57%).- second highest

Median imputation was performed separately for training and testing datasets to avoid data leakage.



Feature Scaling

Min-Max scaling was applied to normalize the features to a range between 0 and 1. This ensures that features with larger values don't disproportionately influence the model's learning process.

3. Exploratory Data Analysis (EDA):

Descriptive Statistics

To understand the data distribution, we computed key statistics such as **mean, median, standard deviation,** and **quartiles**. This helps identify potential outliers and provides a clearer picture of the dataset's overall structure.

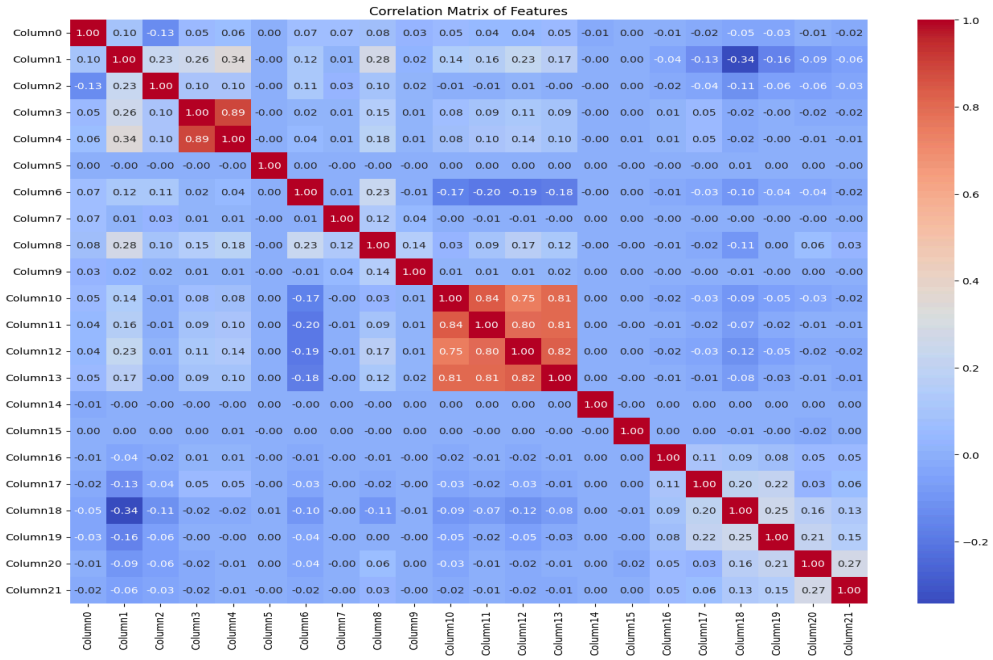
- **Column9**: High proportion of missing values.
- **Correlation Analysis**: Column3 shows a strong positive correlation (0.89) with the target, followed by Column4 (0.81).

Correlation Matrix

- The correlation matrix displays the relationships between different features, represented by both color intensity and numerical values. Key takeaways:
    1. High positive correlations:

- Column3 and Column4 have a high correlation (0.89) with each other, as well as individually with other features, suggesting potential multicollinearity.
- Column10 and Column11 show strong correlations with each other (0.84), indicating they may capture related information.
- Column13 correlates well with Column12 (0.82) and Column11 (0.81).

2. Negative correlations:
   - Column1 stands out with a significant negative correlation with Column18 (-0.34), and smaller negative correlations with Columns3, 4, and 11.
3. Low correlations:
   - Many features, especially Columns0, 9, 14, and 15, show minimal correlations with other columns, indicating they may be independent or weakly related to the rest of the dataset.



4. Model Development and Evaluation:

Model Selection

We evaluated logistic regression, svm, and random forest models to determine the best-performing algorithm for our classification task:

- **Random Forest Classifier**: Initial baseline model, achieving high accuracy and recall.

- **XGBoost Classifier**: Chosen as the final model due to its superior performance and ability to handle complex datasets.

Hyperparameter Tuning with XGBoost:

In this process, we aimed to optimize the performance of the XGBoost classifier through hyperparameter tuning. The key steps included:

- **Parameter Grid Definition**: A range of hyperparameters was defined for tuning, including:
    - n_estimators (200, 300, 350)
    - learning_rate (0.01, 0.05, 0.1)
    - max_depth (4, 5, 7)
- **Model Initialization**: XGBoost was initialized and combined with GridSearchCV, using 5-fold cross-validation and the F1 score as the evaluation metric.
- **GridSearchCV**: Various hyperparameter combinations were tested to find the best set based on cross-validated F1 scores.
- **Best Model Selection**: The optimal hyperparameters and model were selected after grid search completion.
- **Evaluation**: The best model was tested on unseen data, and its performance was measured using accuracy and F1 scores.

Model Evaluation

- **Random Forest Classifier**:
    - Accuracy: **97.59%**
    - Precision: **84.04%**
    - Recall: **91.96%**
    - F1 Score: **87.82%**
    - ROC-AUC: **0.99**
- **XGBoost Model**: After hyperparameter tuning, the XGBoost model achieved:
    - Accuracy: **97.83%**
    - F1-Score: **0.8910**
    - Precision: **84.65%**
    - Recall: **94.04%**
    - ROC-AUC: **0.9948**

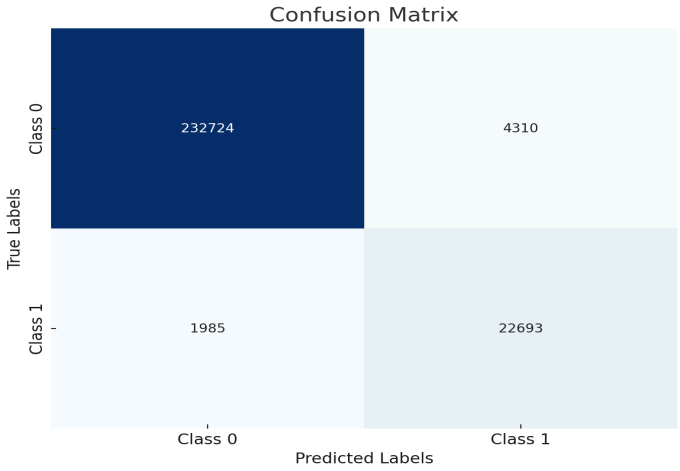
These metrics indicate the model performs exceptionally well, with strong recall and precision, making it highly suitable for imbalanced datasets.

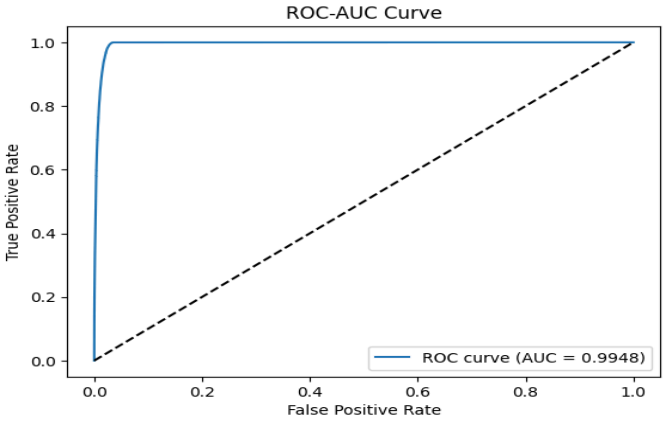
Confusion Matrix

The confusion matrix for the XGBoost model is as follows:

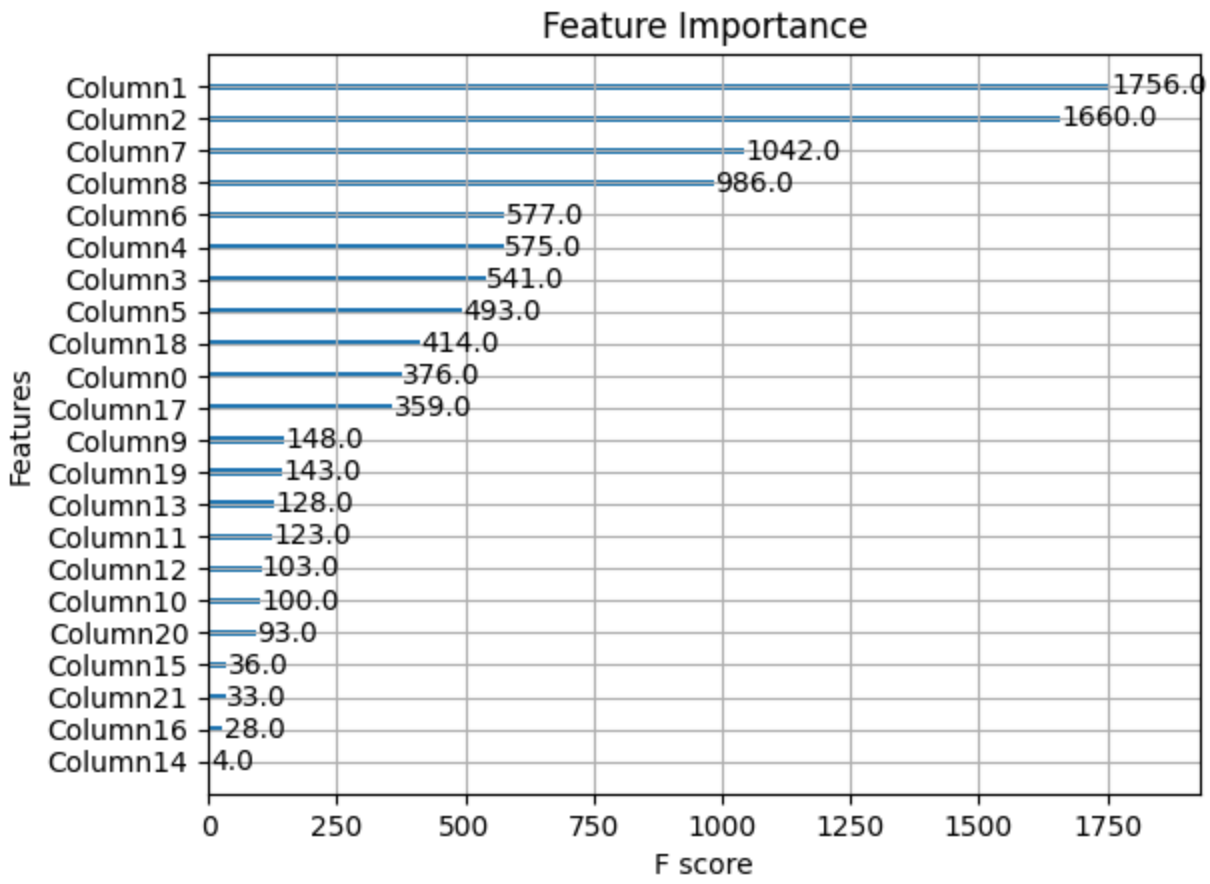
- **True Positives**: 22,693
- **True Negatives**: 232,724
- **False Positives**: 4,310
- **False Negatives**: 1,985

This breakdown demonstrates the model's excellent discriminative capabilities, as indicated by the **high ROC-AUC score** of **0.9948**.





5.Feature Importance Analysis:



The feature importance graph shows that **Column1** and **Column2** are the most influential features, followed by **Column7** and **Column8**. The top four features significantly impact the model, while many others have minimal influence. This suggests focusing on key features for model optimization.

Key Observations:

- **Column 1**: Most important feature, frequently influencing decisions.
- **Columns 2 and 7**: Highly influential, providing valuable model insights.
- **Columns 9 and 10**: Least important, with minimal impact on performance.

6. Conclusion:

- The final XGBoost model, after optimization, demonstrated excellent performance across all evaluation metrics. It handles missing data effectively and provides robust predictions, making it suitable for large-scale classification tasks.
- The final model was saved for future use, and predictions were exported to a CSV file.

7. Appendices:

The XGBoost model, trained with optimized hyperparameters, demonstrated excellent performance on the test set, achieving high accuracy, F1-score, and ROC-AUC. The model effectively handles missing data and demonstrates strong predictive capabilities.

9. Citation Report:

- **Libraries:**In this project, the following libraries were used for data processing, model development, and evaluation:
    - Scikit-learn – A robust machine learning library providing simple and efficient tools for data modeling, classification, and regression. Scikit-learn Documentation
    - XGBoost – A highly efficient, scalable machine learning library for gradient boosting, offering strong performance for structured data. XGBoost Documentation
    - Pandas – A powerful library for data manipulation, offering flexible data structures for easy data cleaning, analysis, and transformation. Pandas Documentation
    - NumPy – The foundational library for numerical computing, providing support for large multi-dimensional arrays and matrices. NumPy Documentation
    - Matplotlib – A versatile plotting library for creating a wide range of static, animated, and interactive visualizations. Matplotlib Documentation
    - Seaborn – A visualization library built on top of Matplotlib, designed for creating informative and aesthetically pleasing statistical graphics. Seaborn Documentation
    - Joblib – A library for efficient serialization of Python objects, commonly used to save models and pipelines. Joblib Documentation

  These libraries form the backbone of the project's data processing, model development, and evaluation workflows.

9.Plagiarism Declaration: We declare that the submitted code and report are our original work. Any external code snippets or ideas have been properly cited.

This structured report provides a comprehensive overview of the model development process, results, and key findings. The inclusion of visualizations and clear explanations strengthens the presentation and allows for a deeper understanding of the model's performance.