GSTIN Classification Project Report

Team members:
Akhilesh T S
Karthik Sriram V

Executive Summary
This project implements a machine learning solution for GSTIN (Goods and Services Tax Identification Number) classification using LightGBM with hyperparameter optimization. The model achieves robust performance with comprehensive evaluation metrics and visualizations.

1. Project Overview

1.1 Objectives

- Develop a classification model for GSTIN data
- Optimize model performance through hyperparameter tuning
- Evaluate model performance using multiple metrics
- Provide comprehensive visualization of results

1.2 Technologies Used

- Python 3.x
- LightGBM
- Scikit-learn
- Hyperopt
- Pandas/NumPy
- Matplotlib/Seaborn

2. Data Processing Pipeline

2.1 Data Loading

- Training and testing datasets loaded from CSV files
- Features and labels combined into unified dataframes

2.2 Preprocessing Steps

1. Data Cleaning

- Removal of unnecessary columns (ID, Column9)
- Handling of missing values using specialized imputation strategies:
    - Mean imputation for Column0
    - Median imputation for Column3, Column4, Column6, Column8, Column15
    - Iterative imputation for Column5, Column14

2. Feature Engineering

- Feature reduction: Removed 9 columns identified as less important
- Outlier removal using z-score method (threshold: 3)
- Feature scaling using StandardScaler

3. Model Development

3.1 Model Selection

- Algorithm: LightGBM Classifier
- Rationale: Efficient gradient boosting framework suitable for classification tasks

3.2 Hyperparameter Optimization

- Optimization Method: Bayesian optimization using Hyperopt
- Search Space:
    - n_estimators: 100-1000
    - learning_rate: 0.01-0.3 (log-uniform)
    - max_depth: 3-10
    - num_leaves: 20-100
    - min_child_samples: 10-100
    - subsample: 0.5-1.0

    - colsample_bytree: 0.5-1.0
    - reg_alpha: 1e-8-1.0 (log-uniform)
    - reg_lambda: 1e-8-1.0 (log-uniform)

4. Model Performance

4.1 Key Metrics

- Accuracy: Shows overall prediction success
- Precision: Indicates true positive accuracy
- Recall: Measures ability to find all positive cases
- F1 Score: Harmonic mean of precision and recall
- AUC-ROC: Evaluates model's discriminative ability
- Balanced Accuracy: Accounts for class imbalance
- Log Loss: Measures probability prediction quality

4.2 Visualization Suite

1. Confusion Matrix
    - Visual representation of prediction errors and successes
    - Helps identify specific classification strengths/weaknesses

2. ROC Curve
    - Plots true positive rate against false positive rate
    - AUC value indicates model's discriminative ability

3. Precision-Recall Curve
    - Shows trade-off between precision and recall
    - Particularly useful for imbalanced datasets

4. Train vs Test Accuracy Comparison
    - Visualizes model's generalization capability
    - Helps detect potential overfitting

5. Results

Following are the best hyperparameters obtained from the model

Best Hyperparameters: {'colsample_bytree': 0.9579278157315392, 'learning_rate': 0.06883860730453947, 'max_depth': 8.0, 'min_child_samples': 52.0, 'n_estimators': 116.0, 'num_leaves': 52.0, 'reg_alpha': 1.3013899454475667e-06, 'reg_lambda': 0.6208410278564448, 'subsample': 0.791360581118324}

The following results are achieved using the model:

1. Accuracy: 97.81%
2. Precision: 0.85
3. Recall: 0.94
4. F1 Score: 0.89
5. AUC-ROC: 0.99
6. Balanced Accuracy: 0.96
7. Log Loss: 0.05

6. Strengths and Limitations

6.1 Strengths

1. Comprehensive preprocessing pipeline
2. Advanced hyperparameter optimization
3. Robust evaluation methodology
4. Multiple visualization techniques

6.2 Limitations

1. Dependency on specific feature columns
2. Potential data loss from outlier removal
3. Computational intensity of hyperparameter optimization

7. Future Improvements

1. Feature importance analysis
2. Cross-validation implementation
3. Ensemble modeling approaches
4. Advanced feature engineering
5. Model interpretability analysis

8. Conclusion

The project demonstrates a well-structured machine learning pipeline for GSTIN classification. The combination of careful preprocessing, hyperparameter optimization, and comprehensive evaluation provides a robust foundation for classification tasks. The multiple evaluation metrics and visualizations offer clear insights into model performance and potential areas for improvement.