Technical Report

TASK FOR RESEARCH ASSISTANT ROLE

Tabular-Data Classification

Author:
Ashish Kumar Pokharel

Submitted to:
NAAMII Selection Team

May 31, 2025

1 Data Preprocessing & Feature Engineering

1.1 Data Loading and Cleaning

- Loaded training, test, and blinded test datasets from CSV files
- Handled missing values: Replaced infinities with NaNs and filled with feature medians
- Capped extreme values at 5th and 95th percentiles
- Ensured consistent ID column handling across all datasets

1.2 Feature Engineering

- Created custom `FeatureChange` transformer
- Engineered new features:
  - Feature interaction terms (product of first two features)
  - Logarithmic transformations for positive features
- Aligned features across datasets: Added zero-filled columns for missing features in test/blinded sets

1.3 Processing Pipeline

1. **Feature Engineering**: Custom transformations for feature creation
2. **Scaling**: StandardScaler normalization (fit on training, transform on all sets)
3. **Feature Selection**: Recursive Feature Elimination (RFE) with Logistic Regression
   - Selected top 50 features using 5-fold cross-validation
   - Applied same feature selection to all datasets
4. **Class Imbalance**: Integrated SMOTE with model training pipelines

2 Model Architectures & Key Hyperparameters

2.1 Implemented Algorithms

- **Logistic Regression**:
  - Class weight: Balanced
  - Hyperparameters: Regularization strength (`C`), solver
- **Random Forest**:
  - Class weight: Balanced subsample
  - Hyperparameters: `n_estimators` (100/200), `max_depth` (5/10/None)

- **XGBoost**:
  - Evaluation metric: Log loss
  - Hyperparameters: `learning_rate` (0.01/0.1), `max_depth` (3/6), `n_estimators` (100/200)
- **LightGBM**:
  - Hyperparameters: `learning_rate` (0.01/0.1), `max_depth` (3/6), `n_estimators` (100/200)

2.2 Stacking Ensemble

- Combined predictions from all base models
- Meta-classifier: Logistic Regression with class weighting
- Stacking method: Predict probabilities

2.3 Hyperparameter Tuning

- RandomizedSearchCV with 10 iterations per model
- 5-fold stratified cross-validation
- Optimization metric: AUROC

3 Cross-Validation Scheme

3.1 Validation Strategy

1. **Initial Split**: 80% training / 20% validation (stratified by class)
2. **Hyperparameter Tuning**: 5-fold stratified CV during RandomizedSearchCV
3. **Probability Calibration**: 5-fold CV with isotonic regression
4. **Final Evaluation**: Validation set for model selection, test set for unbiased evaluation

4 Results

4.1 Validation Set Performance

Model	Accuracy	AUROC	Sensitivity	Specificity	F1-score
LogisticRegression	0.832	0.901	0.812	0.851	0.821
RandomForest	0.845	0.917	0.831	0.859	0.839
XGBoost	0.851	0.923	0.843	0.859	0.847
LightGBM	0.853	0.926	0.845	0.861	0.849
Stacking Ensemble	0.862	0.938	0.857	0.867	0.859

4.2 Test Set Performance

Model	Accuracy	AUROC	Sensitivity	Specificity	F1-score
Best Model	0.847	0.928	0.839	0.855	0.842

5 Discussion

5.1 Strengths

- **Class Imbalance Handling**: Combined SMOTE and class weighting
- **Model Diversity**: Linear, tree-based, and ensemble methods

5.2 Limitations

- **Computational Cost**: Stacking ensemble increases training time
- **Timing Issue**: Due to time limitation it was hard for me to properly focus on the training. Given more time, with more research I could do better using more processing techniques.

5.3 Future Improvements

- **Advanced Feature Selection**: Model-specific importance analysis
- **Neural Networks**: Implement simple MLP for comparison
- **Dimensionality Reduction**: PCA/t-SNE for high-dimensional cases
- **Ensemble Pruning**: Remove redundant base models