CHAPTER 3: METHODOLOGY

3.1 Overview

This chapter presents a comprehensive methodology for predicting Antimicrobial Resistance (AMR) using machine learning techniques applied to resistance gene profiles. The proposed approach demonstrates that resistance gene-based prediction can achieve performance comparable to, or exceeding, whole-genome sequence (WGS)-based methods while maintaining computational efficiency and biological interpretability. The methodology encompasses six key stages: (1) Dataset Collection and Preprocessing, (2) Feature Engineering, (3) Feature Selection, (4) Class Imbalance Handling, (5) Model Training and Ensemble Construction, and (6) Model Interpretability and Explainability.

The pipeline was designed to address the specific challenges inherent in AMR genomic data, including high dimensionality, extreme sparsity of the binary gene presence/absence matrices, and class imbalance between resistant and susceptible isolates. Figure 3.1 illustrates the complete workflow of the proposed methodology.

3.2 Dataset Collection and Preprocessing

3.2.1 Data Source

The study utilized antimicrobial susceptibility testing (AST) data and associated genomic information from the National Center for Biotechnology Information (NCBI) Pathogen Detection Database. This database provides curated isolate metadata, including phenotypic resistance profiles and computationally predicted antimicrobial resistance genes identified through the AMRFinderPlus pipeline.

Twelve datasets were compiled, representing different antibiotic-pathogen combinations across multiple bacterial species:

- **Klebsiella pneumoniae (KN):** Doripenem, Ertapenem, Imipenem, Meropenem
- **Escherichia coli and Shigella (ECS):** Doripenem, Ertapenem, Imipenem, Meropenem
- **Pseudomonas aeruginosa (PA):** Doripenem
- **Salmonella enterica (SE):** Streptomycin, Kanamycin
- **Campylobacter jejuni (CJ):** Clindamycin

3.2.2 Binary Gene Matrix Construction

For each antibiotic-pathogen combination, the raw genomic data was transformed into a binary gene presence/absence matrix. Let $G = \{g_1, g_2, ..., g_n\}$ represent the set of all unique

antimicrobial resistance genes detected across all isolates in a given dataset. For each isolate i, a feature vector $x_i \in \{0, 1\}^n$ was constructed where:

$$x_{ij} = 1 \text{ if gene } g_j \text{ is present in isolate } i, \text{ and 0 otherwise}$$

Gene annotations included information about detection completeness (COMPLETE, PARTIAL, PARTIAL_END_OF_CONTIG), which was preserved as separate binary features to capture potential information about gene integrity and expression likelihood. The target variable $y \in \{0, 1\}$ encoded the phenotypic resistance status, where y = 1 indicates resistance and y = 0 indicates susceptibility to the specified antibiotic.

3.2.3 Dataset Characteristics

The compiled datasets exhibited considerable variation in sample size (ranging from 26 to 1,042 isolates), feature dimensionality (11 to 326 resistance genes), and class distribution. Table 3.1 summarizes the characteristics of each dataset, including the number of resistant and susceptible isolates and the computed class imbalance ratio.

3.3 Feature Engineering

3.3.1 Resistance Gene Load Score (R-Score)

A novel engineered feature, termed the Resistance Gene Load Score (R-Score), was introduced to capture the cumulative resistance gene burden of each isolate. The biological rationale underlying this feature is that isolates harboring a larger number of resistance genes may exhibit higher likelihood of phenotypic resistance, as resistance mechanisms often act synergistically or provide redundant protection against antimicrobial action.

The R-Score for isolate i is computed as the sum of all binary gene presence indicators:

$$R\text{-}Score_i = \Sigma_j x_{ij} \quad (\text{where } j = 1 \text{ to } n)$$

To ensure comparability across datasets with different total gene counts and to prevent scale-related issues during model training, the R-Score was normalized using Min-Max scaling:

$$R\text{-}Score'_i = (R\text{-}Score_i - min(R\text{-}Score)) / (max(R\text{-}Score) - min(R\text{-}Score))$$

The normalized R-Score was appended to the feature matrix as an additional continuous feature, positioned immediately before the target variable. This placement ensures the R-Score is included in downstream feature selection procedures and can contribute to predictive models alongside individual gene presence indicators.

3.3.2 Justification and Impact Analysis

Preliminary experiments demonstrated that inclusion of the R-Score consistently improved model performance across multiple datasets. Histograms of R-Score distributions stratified by resistance status revealed clear separation between resistant and susceptible isolates in most datasets, validating the biological hypothesis that gene accumulation correlates with resistance phenotype.

3.4 Feature Selection

AMR gene matrices are characterized by high dimensionality and extreme sparsity, with many genes present in only a small fraction of isolates. Effective feature selection is critical to remove noise, reduce overfitting risk, and improve computational efficiency. This study implements a novel hybrid feature selection strategy that combines filter-based and embedded methods to capture features relevant under both linear and non-linear modeling assumptions.

3.4.1 Stage 1: Statistical Filter Selection (ANOVA F-test)

The first stage employs Analysis of Variance (ANOVA) F-test as a univariate statistical filter to identify features with significant linear association with the target variable. For each feature g_j, the F-statistic measures the ratio of between-class variance to within-class variance:

$$F(g_j) = \text{(Between-group variability)} / \text{(Within-group variability)}$$

Features with ANOVA p-value ≤ 0.30 were retained in the ANOVA-selected feature set. This relatively relaxed threshold was chosen based on empirical evaluation across multiple datasets, balancing the retention of potentially informative features against noise reduction. The R-Score feature was automatically retained regardless of its p-value to ensure its inclusion in downstream modeling.

3.4.2 Stage 2: Model-Based Importance Selection (XGBoost)

The second stage utilizes XGBoost (eXtreme Gradient Boosting) as an embedded feature selector capable of capturing non-linear feature relationships and complex interactions. An XGBoost classifier was trained on the full feature set with the following hyperparameters:

- Number of estimators: 400
- Maximum depth: 6
- Subsample ratio: 0.8
- Column subsample ratio: 0.8

Feature importances were extracted based on the gain criterion, which measures the average improvement in loss function contributed by each feature across all trees. Features were ranked by importance, and a cumulative importance threshold of 85% was applied. Specifically, features were selected in descending order of importance until their cumulative contribution reached 85% of the total importance sum:

$$S\_XGB = \{g_j : \Sigma_i \, Importance(g_i) \leq 0.85 \times \Sigma_k \, Importance(g_k)\}$$

3.4.3 Union-Based Feature Integration

The final feature set was constructed as the union of ANOVA-selected and XGBoost-selected features:

$$S\_final = S\_ANOVA \cup S\_XGB$$

This union-based approach ensures that features important under either linear (ANOVA) or non-linear (XGBoost) assumptions are retained, providing comprehensive coverage of potentially predictive genes. The union strategy outperformed both intersection-based selection and individual method selection in preliminary experiments, likely because it captures complementary information identified by each method's distinct modeling perspective.

3.4.4 Hyperparameter Selection Rationale

Grid search experiments were conducted to optimize the feature selection hyperparameters. The following parameter ranges were evaluated:

- ANOVA p-value threshold: {0.05, 0.10, 0.30}
- Selector model: {Random Forest, XGBoost}
- Cumulative importance cutoff: {85%, 90%, 95%}

The combination of ANOVA $p \leq 0.30$, XGBoost selector, and 85% cumulative importance cutoff yielded the highest average F1-score (0.955) and AUROC (0.965) across all datasets, and was therefore selected as the optimal configuration.

3.5 Class Imbalance Handling

3.5.1 Challenge of Imbalanced Data

Class imbalance is a prevalent challenge in AMR prediction, where the distribution of resistant and susceptible isolates varies substantially across different antibiotic-pathogen combinations. Imbalance ratios in this study ranged from nearly balanced (1.0) to highly skewed (3.2:1), which can bias classifiers toward the majority class and result in poor sensitivity for resistance detection.

3.5.2 Evaluation of Resampling Strategies

Multiple resampling strategies were evaluated to address class imbalance:

1. **Random Oversampling:** Duplicates random minority class samples
2. **Random Undersampling:** Removes random majority class samples
3. **SMOTE (Synthetic Minority Over-sampling Technique):** Generates synthetic minority samples via k-nearest neighbor interpolation
4. **ADASYN (Adaptive Synthetic Sampling):** Adaptively generates more synthetic samples for harder-to-learn minority instances
5. **Borderline-SMOTE:** Focuses synthetic sample generation on borderline minority instances
6. **SMOTE-ENN:** Combines SMOTE with Edited Nearest Neighbors cleaning
7. **SMOTE-Tomek:** Combines SMOTE with Tomek links removal

3.5.3 Selection of SMOTETomek

SMOTETomek was selected as the optimal resampling strategy based on comparative experiments. This hybrid method combines SMOTE's synthetic minority sample generation with Tomek links cleaning, which removes potentially noisy or ambiguous samples near the decision boundary.

The selection of SMOTETomek is particularly appropriate for AMR gene matrices due to the following characteristics:

- **High Dimensionality and Sparsity:** Simple SMOTE can generate unrealistic synthetic samples in sparse, high-dimensional spaces by interpolating between distant points. The Tomek links cleaning step removes such borderline samples that may not represent biologically plausible gene combinations.
- **Cleaner Decision Boundaries:** By removing Tomek links—pairs of nearest neighbors belonging to different classes—the method creates clearer class separation, improving classifier performance.
- **Reduced Overfitting Risk:** The cleaning step mitigates the risk of overfitting to synthetic samples that may not generalize to real-world isolates.

Resampling was applied exclusively to the training set after the train-test split to prevent data leakage and ensure unbiased evaluation on the held-out test set.

3.6 Model Training and Ensemble Construction

3.6.1 Individual Base Models

Multiple machine learning algorithms were evaluated as individual predictors to assess their suitability for AMR prediction from resistance gene profiles:

1. **Logistic Regression (LogR):** A linear classifier that models the log-odds of resistance as a linear combination of features. Configured with maximum iterations of 1000 to ensure convergence.

2. **Support Vector Machine (SVM):** A kernel-based classifier using RBF kernel to capture non-linear decision boundaries. Probability estimates enabled for ensemble integration.
3. **Decision Tree (DT):** A tree-based classifier using entropy criterion for split selection. Provides interpretable decision rules.
4. **Random Forest (RF):** An ensemble of decision trees with 10 estimators and entropy criterion.
5. **XGBoost (XGB):** A gradient boosting implementation optimized for performance and regularization, using logloss evaluation metric.

3.6.2 Ensemble Model: R-Blend

A custom ensemble model, designated R-Blend (Resistance Gene Blend), was constructed to leverage the complementary strengths of multiple base classifiers. R-Blend employs soft voting, where the final prediction is based on the weighted average of class probability estimates from constituent models.

Ensemble Composition:

- Decision Tree (DT) — Weight: 1.0
- Logistic Regression (LogR) — Weight: 1.5
- XGBoost (XGB) — Weight: 1.0

The weighted probability aggregation for class c is computed as:

$$P(y = c \mid x) = \Sigma_i \, w_i \times P_i(y = c \mid x) / \Sigma_i \, w_i$$

where w_i denotes the weight assigned to model i and $P_i(y = c \mid x)$ represents the probability estimate for class c from model i.

3.6.3 Weight Selection Rationale

The weight assignment of [1, 1.5, 1] for [DT, LogR, XGB] was determined through grid search optimization. The elevated weight for Logistic Regression reflects several empirical and theoretical considerations:

1. **Linear Separability Hypothesis:** Analysis of the feature space suggested that resistance classes exhibit substantial linear separability, particularly after feature selection. The R-Score feature, which captures cumulative gene burden, contributes significantly to this linear separation.
2. **Calibration Quality:** Logistic Regression produces well-calibrated probability estimates, which benefits soft voting aggregation. Tree-based models (DT, XGB) may produce less calibrated probabilities, especially near decision boundaries.
3. **Regularization Effect:** The increased LogR weight provides implicit regularization against overfitting tendencies of tree-based models, particularly beneficial for smaller datasets.
4. **Empirical Validation:** Cross-validation experiments demonstrated that this weight configuration yielded superior generalization performance compared to equal weighting or alternative configurations.

3.6.4 Training Protocol

The following training protocol was implemented:

- **Data Splitting:** Stratified 80-20 train-test split to preserve class distribution
- **Resampling:** SMOTETomek applied to training set only
- **Model Fitting:** R-Blend ensemble trained on resampled training data
- **Evaluation:** Performance assessed on held-out test set
- **Cross-Validation:** 5-fold stratified cross-validation on training data for hyperparameter selection and stability assessment
- **Random State:** Fixed seed (42) for reproducibility across all stochastic components

3.7 Evaluation Metrics

Model performance was assessed using a comprehensive suite of metrics to capture different aspects of predictive quality:

- **Accuracy:** Proportion of correctly classified isolates: (TP + TN) / (TP + TN + FP + FN)
- **Precision:** Proportion of predicted resistant isolates that are truly resistant: TP / (TP + FP)
- **Recall (Sensitivity):** Proportion of truly resistant isolates correctly identified: TP / (TP + FN)
- **F1-Score:** Harmonic mean of precision and recall: 2 × (Precision × Recall) / (Precision + Recall)
- **AUROC (Area Under ROC Curve):** Measures discriminative ability across all classification thresholds
- **AUPRC (Area Under Precision-Recall Curve):** Particularly informative for imbalanced datasets, emphasizing positive class prediction quality

F1-Score and AUROC were prioritized as primary evaluation metrics given their robustness to class imbalance and their relevance to the clinical application of AMR prediction.

3.8 Model Interpretability and Explainability

Interpretability is crucial for clinical acceptance of AMR prediction models, as clinicians and microbiologists require understanding of which genetic factors drive predictions. This study incorporates multiple complementary interpretability approaches.

3.8.1 Permutation Feature Importance

Permutation importance quantifies the contribution of each feature by measuring the decrease in model performance when feature values are randomly shuffled. For each feature g_j:

$$Importance(g_j) = Performance\_original - Performance\_permuted(g_j)$$

This model-agnostic approach can be applied to any trained classifier and captures the feature's unique contribution to predictive accuracy.

3.8.2 SHAP (SHapley Additive exPlanations)

SHAP values provide theoretically grounded, consistent feature attributions based on cooperative game theory. For each prediction, SHAP decomposes the model output into additive contributions from each feature:

$$f(x) = \varphi_0 + \Sigma_j \varphi_j(x)$$

where φ_0 is the base value (expected model output) and $\varphi_j(x)$ represents the contribution of feature g_j for instance x.

Model-specific SHAP explainers were employed:

- **TreeExplainer:** Efficient exact SHAP computation for Decision Tree and XGBoost models
- **LinearExplainer:** Analytical SHAP computation for Logistic Regression

3.8.3 Ensemble SHAP Aggregation

For the R-Blend ensemble, SHAP values from individual base models were aggregated using the same weights as the soft voting mechanism:

$$\varphi_{j\_ensemble}(x) = \Sigma_i w_i \times \varphi_{ji}(x) / \Sigma_i w_i$$

This weighted aggregation provides ensemble-level feature attributions that reflect the contribution weighting of constituent models.

3.8.4 Global and Local Explanations

Both global and local explanations were generated:

- **Global Importance:** Mean absolute SHAP values across all test instances identify features most influential for the model's overall predictions
- **Local Explanations:** Instance-level SHAP values explain individual predictions, identifying the top contributing genes for each classified isolate
- **Misclassification Analysis:** SHAP-based analysis of misclassified samples identifies potential sources of prediction errors and highlights cases where gene profiles may conflict with phenotypic resistance status

3.9 Experimental Design and Validation

3.9.1 Ablation Studies

Systematic ablation experiments were conducted to quantify the contribution of each pipeline component:

- Baseline without R-Score vs. with R-Score feature
- No resampling vs. various resampling strategies

- Individual models vs. R-Blend ensemble
- Different feature selection configurations

3.9.2 Cross-Dataset Generalization

The methodology was validated across all 12 antibiotic-pathogen datasets to assess generalization capability. Performance metrics were aggregated (mean ± standard deviation) to evaluate consistency across diverse resistance mechanisms and bacterial species.

3.9.3 Comparative Analysis

The proposed resistance gene-based methodology was compared against published approaches:

- **Resistance Gene-Based Baseline:** Machine learning framework by Her & Wu (2021) using resistance genes for AMR prediction
- **Whole-Genome Sequence Approach:** Deep learning methods by Yasir et al. (2023) utilizing complete genomic sequences

This comparison demonstrates the competitiveness of the resistance gene-based approach relative to computationally intensive whole-genome methods.

3.10 Implementation Details

The complete pipeline was implemented in Python 3.x using the following key libraries:

- **scikit-learn:** Model training, evaluation, feature selection, and preprocessing
- **XGBoost:** Gradient boosting classifier and feature importance extraction
- **imbalanced-learn:** Resampling strategies including SMOTETomek
- **SHAP:** Model-agnostic and model-specific explainability
- **pandas/NumPy:** Data manipulation and numerical computation
- **matplotlib/seaborn:** Visualization

All experiments were conducted on Google Colab with GPU acceleration where applicable. Source code and processed datasets are available in the supplementary materials.

3.11 Chapter Summary

This chapter presented a comprehensive methodology for AMR prediction from resistance gene profiles. The key methodological contributions include: (1) the R-Score engineered feature capturing cumulative gene burden, (2) a hybrid feature selection strategy combining ANOVA and XGBoost importance, (3) SMOTETomek resampling tailored for sparse binary gene matrices, (4) the R-Blend weighted soft voting ensemble, and (5) integrated SHAP-based explainability. The following chapter presents experimental results demonstrating the effectiveness of this methodology across diverse antibiotic-pathogen combinations.