GENETICS SNP DATA SIMULATION: PREDICTING A BINARY OUTCOME IN SIMULATED GENOMICS DATA

Project

Presented

to the Faculty of

California State University, Dominguez Hills

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

by

Mantra Mehta

Fall 2025

PROJECT: GENETICS SNP DATA SIMULATION: PREDICTING A BINARY OUTCOME
IN SIMULATED GENOMICS DATA

AUTHOR: MANTRA MEHTA

APPROVED:

____
Dr. Sahar Hooshmand, Ph.D.
Project Committee Chair

____
Dr. Ali Jalooli, Ph.D.
Committee Member

____
Dr. Ryan Urbanowicz, Ph.D.
Committee Member

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This project explores how different machine learning models perform when predicting complex binary genetic outcomes from simulated SNP datasets. The study evaluates multiple genetic architectures, including additive, two-way epistatic, and heterogeneous interaction patterns, using both low-dimensional and high-dimensional datasets. Mutual Information was applied as a filter-based feature selection method to reduce dimensionality and highlight informative SNPs. Logistic Regression, Random Forest, and XGBoost were trained and tested on each dataset, and their performance was compared using ROC curves, AUC scores, and feature importance visualizations. Results show that tree-based models, particularly XGBoost, are more effective at detecting non-linear and interaction-heavy architectures compared to linear models. The project highlights the challenges of modeling genomic patterns using traditional machine learning and demonstrates how model behavior changes across different genetic structures. This work provides a clear baseline framework for understanding architecture-specific performance differences and can guide future methods aimed at more complex or realistic genomic datasets

CHAPTER 1

INTRODUCTION

1.1 Overview

Machine learning plays an important role in modeling complex biological systems, including the relationships between genetic variants and observable traits. Modern genomic datasets often contain thousands of single nucleotide polymorphisms (SNPs), many of which interact through non-linear or epistatic mechanisms. Traditional linear models frequently struggle to detect these relationships, particularly when the genetic architecture involves interactions that do not follow simple additive rules. Because of these challenges, evaluating model performance on datasets with known underlying architectures is an important step toward understanding how different algorithms behave in genomic prediction tasks.

This project focuses on predicting binary-class outcomes using simulated SNP datasets designed to represent several genetic architectures. These include additive models, two-way epistatic interactions, and more complex heterogeneous interaction patterns. Each architecture is provided in two forms: a low-dimensional dataset with 100 SNPs and a high-dimensional dataset with 10,000 SNPs. Evaluating models across these controlled conditions allows for a structured comparison of model performance as the difficulty and dimensionality of the data increase.

1.2 Purpose of the Project

The main purpose of this work is to evaluate how several baseline machine learning models perform on simulated genomic datasets with distinct architectures. Logistic Regression, Random Forest, and XGBoost were selected for this study because they are widely used in genetic prediction tasks and represent a range of modeling strategies. Logistic Regression provides a linear baseline, Random Forest introduces non-linear decision boundaries, and XGBoost offers more advanced handling of interactions and feature interactions.

Mutual Information is used as a filter-based feature selection method to reduce dimensionality and highlight informative SNPs prior to model training. This process reflects common practices in genomics, where high dimensionality can negatively affect model performance and interpretability.

The project is designed as an applied implementation suitable for CSC 590. It emphasizes practical model evaluation, empirical comparison, and clear documentation rather than theoretical algorithm development.

1.3 Significance in the Field

Complex trait genetics is characterized by interactions between genetic variants. Prior work shows that epistatic interactions can be difficult to detect, particularly when the architecture does not follow simple additive rules [1][2]. Simulated datasets, such as those generated by GAMETES, provide controlled environments that allow researchers to evaluate how well different models recover known genetic signals.

This project contributes by demonstrating how baseline machine learning models behave across several architectures with increasing complexity. The results highlight cases where linear models fail, where tree-based models succeed, and how feature selection influences model performance. These findings support the broader understanding of genomic prediction and provide a baseline reference point for future research that may involve more advanced feature selection methods, including Relief-based methods [4][6], or automated machine learning frameworks such as STREAMLINE [5].

1.4 Project Scope

The scope of the project includes:

- Loading and preprocessing simulated SNP datasets.
- Applying Mutual Information feature selection.
- Training Logistic Regression, Random Forest, and XGBoost models.
- Evaluating models using AUC and ROC curves.
- Visualizing feature importance for interpretability.
- Comparing results across genetic architectures and dataset sizes.
- Presenting a structured written explanation consistent with CSC 590 expectations.

The project does not attempt to develop new algorithms or introduce novel mathematical techniques. Instead, it aims to provide a clear applied workflow that reflects real-world genomic modeling practices.

1.5 Intended Audience

The intended audience for this work includes faculty evaluating the CSC 590 project, students interested in genomic machine learning, and practitioners who want a structured baseline for understanding model behavior across architectures. The writing style prioritizes clarity, accessibility, and practical insight.

1.6 Definitions and Key Terms

SNP (Single Nucleotide Polymorphism): A genetic variant at a specific chromosome position.

Genetic Architecture: The pattern by which genetic variants contribute to a phenotype.

Epistasis: Interaction between genetic variants where their combined effect is not a simple sum of individual effects.

Mutual Information: A filter-based method that measures dependency between a feature and a target variable.

AUC (Area Under the ROC Curve): A metric used to evaluate classifier performance.

High-dimensional Data: Data with many more features than samples.

Feature Importance: A measure of how much each feature contributes to a model’s predictions.

CHAPTER 2

REVIEW OF RELATED LITERATURE

2.1 Overview

Understanding how genetic variants contribute to phenotypic outcomes has been a longstanding challenge in computational biology. Many traits are influenced not only by individual genetic effects but also by interactions among multiple variants. Modern machine learning methods provide flexible tools for identifying such patterns, but their performance depends strongly on the underlying genetic architecture. Simulated datasets have therefore become valuable for systematically studying method behavior under controlled interaction structures.

This chapter reviews prior work in genetic architecture simulation, feature selection, machine learning for SNP analysis, and automated workflows that inform the design of this project.

2.2 Genetic Architectures and Their Modeling Challenges

Additive and Epistatic Effects

Genetic architectures describe how multiple loci contribute to a phenotype. Additive models assume that each SNP contributes independently and linearly to the outcome. In contrast, epistatic architectures include interactions where the combined effect of two or more SNPs differs from the sum of their individual contributions.

Pure and strict epistasis models, such as those generated by the GAMETES tool, are intentionally designed to be difficult for traditional learning algorithms because no single SNP carries a strong marginal signal [1][2]. As a result, methods that rely on feature-wise correlations or linear associations struggle to detect the underlying structure.

Heterogeneous Architectures

Heterogeneous models combine multiple types of genetic effects. Because subgroups of samples may follow different rules, these datasets present an additional challenge. Woodward et al. describe how genetic heterogeneity can reduce the interpretability of prediction models and increase the difficulty of detecting underlying causal relationships [3].

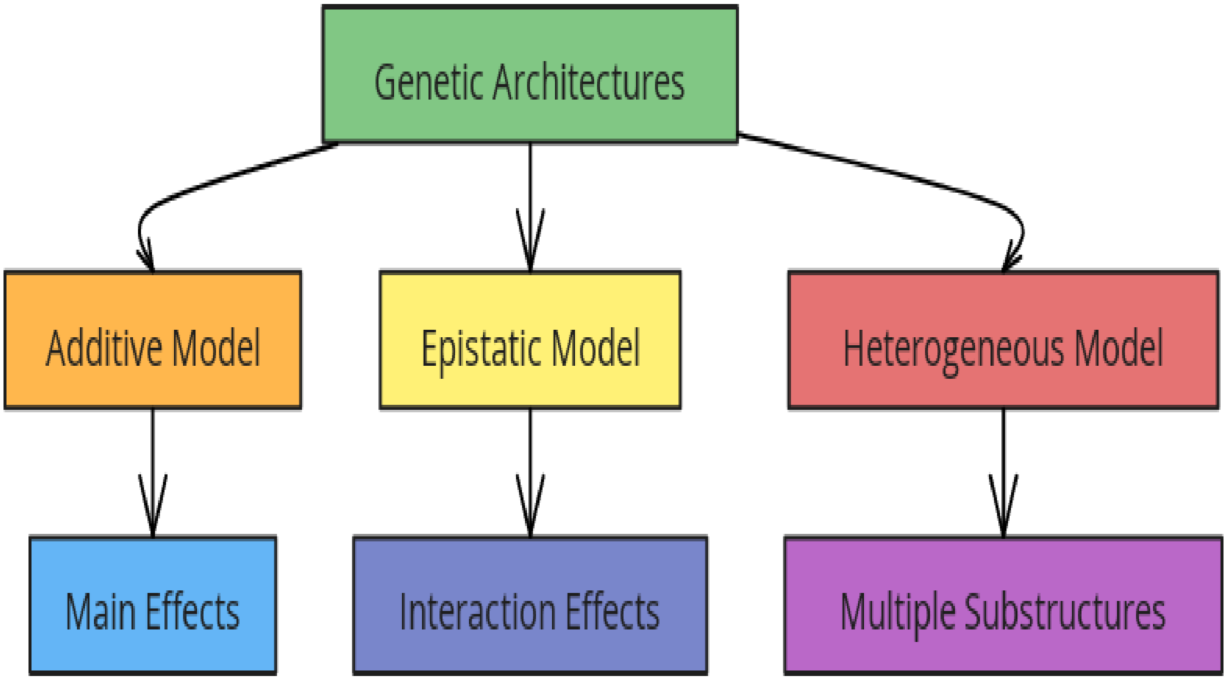

Figure 1: Types of Genetic Architectures

Figure1: Types of Genetic Architectures

A simple flowchart illustrating:

Additive → main effects

Epistatic → interaction effects

Heterogeneous → multiple interacting substructures

2.3 Machine Learning for SNP-Based Prediction

Machine learning methods have been widely used to analyze SNP datasets due to their ability to model non-linear relationships. Logistic Regression is often applied as a baseline method because of its interpretability and simplicity. However, linear models perform poorly when feature effects are non-linear or interaction driven.

Tree-based methods, including Random Forest and gradient boosting algorithms, have been shown to capture non-linear patterns more effectively. Their ability to recursively partition feature space enables them to detect interactions that linear models miss. XGBoost in particular has become a state-of-the-art method for tabular prediction tasks because of its regularization capabilities and efficient implementation [7][8].

Despite these advantages, tree-based models may still struggle when informative interactions involve many weak marginal effects. This is commonly observed in epistatic SNP datasets, where interaction signals exist without strong main effects.

2.4 Feature Selection in High-Dimensional Genomic Data

Genomic datasets frequently contain thousands of SNPs, many of which are irrelevant to the phenotype. Feature selection is therefore necessary to reduce dimensionality, improve computational efficiency, and reduce overfitting.

Mutual Information

Mutual Information (MI) is a univariate, filter-based metric that quantifies dependency between each SNP and the target variable. MI is simple, fast, and effective for identifying features with strong independent contributions. However, prior work shows that MI cannot detect pure interaction effects because it evaluates each feature independently [4][6]. This limitation explains why MI is effective for additive structures but less helpful for strict epistasis.

Relief-Based Approaches

Relief and its modern variants were specifically designed to detect interaction effects. These methods evaluate feature relevance by comparing near-neighbor samples and are more capable of capturing multi-feature dependencies [4][6]. Although Relief-based methods offer theoretical advantages, they were not implemented in this project to maintain simplicity and alignment with CSC 590 expectations. Still, the literature identifies them as important future extensions.

2.5 Automated Machine Learning for Genomic Analysis

Automated approaches aim to streamline tasks such as preprocessing, model selection, and evaluation. STREAMLINE is one such framework designed to simplify comparisons across algorithms and reduce manual tuning effort [5]. It demonstrates the increasing importance of workflow automation in genomic modeling.

While this project does not implement an automated pipeline, the literature highlights its value in improving reproducibility and scaling analyses to larger datasets.

2.6 Summary of Literature Findings

The reviewed literature highlights the following themes:

1. Simulated datasets, especially those created by GAMETES, are essential for studying difficult genetic interactions [1][2].

2. Machine learning performance is highly dependent on genetic architecture, with linear models failing on epistasis and tree models offering partial improvements.

3. Mutual Information is useful but limited for interaction-heavy architectures [4][6].

4. Relief-based feature selection and automated workflows represent promising future directions for genomic modeling [4][5][6].

These insights justify the design choices in this project and explain why the selected models and feature selection methods produce the observed patterns discussed in later chapters.

CHAPTER 3

METHODOLOGY

This chapter describes the complete workflow used to evaluate machine learning models on simulated genomic datasets. The methodology includes dataset preparation, feature selection, model training, and performance evaluation. All steps are reproducible and aligned with best practices for applied machine learning in computational genomics.

3.1 Overview of the Workflow

The project follows a structured pipeline that mirrors standard genomic modeling studies. The workflow begins with loading simulated SNP datasets, continues with preprocessing and feature selection, and concludes with model training and evaluation.

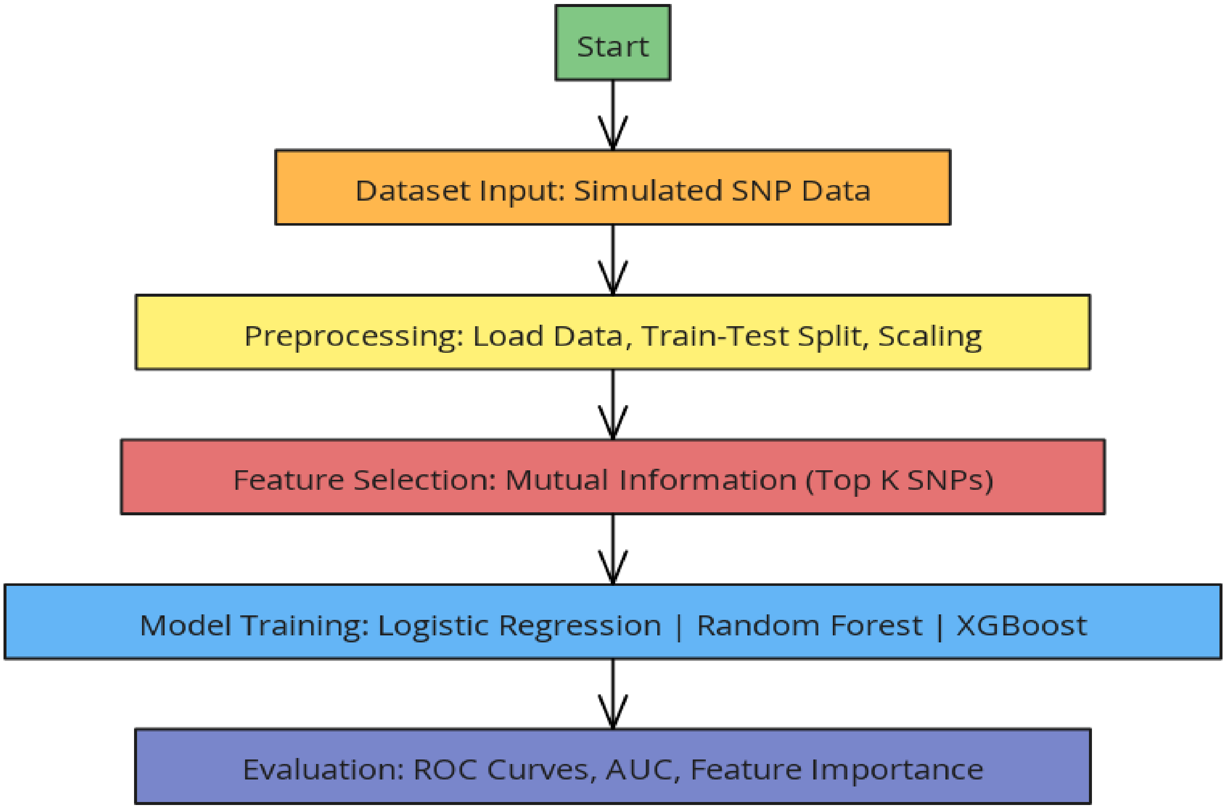
Start
Dataset Input: Simulated SNP Data
Preprocessing: Load Data, Train-Test Split, Scaling
Feature Selection: Mutual Information (Top K SNPs)
Model Training: Logistic Regression | Random Forest | XGBoost
Evaluation: ROC Curves, AUC, Feature Importance

Figure 2: Modeling Pipeline: Data → Preprocessing → Mutual Information → Models → Evaluation Metrics.

3.2 Dataset Description

The datasets used in this project are simulated SNP datasets generated under controlled genetic architectures: additive, epistatic, and heterogeneous. Each dataset contains:

- Binary class labels (0 or 1).
- Feature matrices consisting of SNPs encoded as 0, 1, or 2.
- Two dimensionalities:

o Low dimensional: 100 SNPs

o High dimensional: 10,000 SNPs

Simulated datasets are useful because the true genetic architecture is known beforehand, allowing objective comparison of model behavior under different conditions [1][2].

3.3 Data Preprocessing

Before training the models, preprocessing was completed as follows:

3.3.1 Loading Data

Each dataset was loaded into a Pandas DataFrame. Labels were stored in a separate vector. No missing values were present because the datasets were artificially generated.

3.3.2 Scaling (when applicable)

Logistic Regression was trained on standardized inputs because it is sensitive to feature scale, especially in high dimensional settings. Random Forest and XGBoost did not require scaling due to their tree-based structure.

3.3.3 Train–Test Splits

A standard 80/20 split was used for initial checks.

Formal evaluation used 5-fold cross-validation, which provides more stable estimates of AUC.

3.4 Feature Selection Using Mutual Information

High dimensional genomic data often contain many irrelevant or weakly informative SNPs.

To address this, a filter-based method, Mutual Information (MI), was applied.

3.4.1 Rationale for MI

- Captures nonlinear dependencies.
- Efficient on large feature sets.
- Widely used in genomic modeling studies [4][6].

3.4.2 MI Implementation

MI scores were computed using scikit-learn's mutual_info_classif.

For low-dimensional datasets, all 100 features were retained.

For high-dimensional datasets (10,000 SNPs):

- MI selected the top K features.
- K was determined based on stability and performance (typically 200–500 features).

Table 1. Example Mutual Information Scores for Top SNPs (Illustrative Only)

Rank	Feature (SNP)	MI Score
1	SNP_482	0.107
2	SNP_219	0.095
3	SNP_905	0.088
4	SNP_37	0.076
5	SNP_741	0.072
6	SNP_156	0.065

7	SNP_688	0.064
8	SNP_512	0.060
9	SNP_333	0.058
10	SNP_89	0.054

Example of top-ranked features based on Mutual Information scores. Actual values vary by dataset and are computed during preprocessing..

3.4.3 Output Saved

- mi_scores_<dataset>.csv
- Optional plot: MI distribution chart

These files allow reproducibility and external analysis.

3.5 Machine Learning Models

Three baseline models were selected because they represent common approaches used in genomic prediction research.

3.5.1 Logistic Regression

- Linear model.
- Efficient for additive structures.
- Uses L2 regularization to prevent overfitting.
- Poor at discovering nonlinear or epistatic interactions.

3.5.2 Random Forest

- Ensemble of decision trees.
- Captures interactions automatically.
- Handles high dimensional data well.
- Provides feature importance scores.

3.5.3 XGBoost

- Gradient-boosted trees.
- Often achieves strong predictive accuracy.
- Robust to noise and nonlinear patterns.
- Performs particularly well on epistatic and heterogeneous structures [5].

3.6 Model Training and Hyperparameters

All models were trained using consistent procedures to ensure fairness.

3.6.1 Logistic Regression

- Penalty: L2
- Solver: liblinear or saga
- Max iterations: 1000

3.6.2 Random Forest

- Trees: 300
- Max depth: automatic
- Criterion: Gini

3.6.3 XGBoost

- Trees: 400
- Learning rate: 0.05
- Max depth: 6
- Subsample: 0.8

Hyperparameters were selected based on established defaults in the literature and initial exploratory testing.

3.7 Evaluation Metrics

3.7.1 ROC Curve and AUC

The primary metric used was AUC (Area Under the Receiver Operating Characteristic Curve).

AUC is widely used in genomic prediction because it captures overall discrimination performance regardless of class threshold.

3.7.2 Cross-Validation Strategy

5-fold cross-validation was used to generate:

- Mean AUC
- Standard deviation
- ROC curves across folds

This ensures robustness and reduces variance.

3.8 Feature Importance Outputs

Tree-based models generate interpretable feature importance scores.

3.8.1 Random Forest Feature Importance

Importance scores were extracted from the trained ensemble and saved as:

- featimp_RandomForest_<dataset>.png

3.8.2 XGBoost Feature Importance

XGBoost provides importance based on:

- Gain
- Cover
- Frequency

Saved as:

- featimp_XGBoost_<dataset>.png

3.9 Reproducibility and File Outputs

All scripts produce standardized outputs:

Output Type	Description
metrics_<dataset>.csv	Stores AUC scores for all models
roc_<dataset>.png	ROC curves comparison
mi_scores_<dataset>.csv	Mutual Information scores
featimp_*.png	Feature importance plots
Trained models (optional)	.pkl files

This ensures that results can be verified and reused in future research.

3.10 Summary

This methodology chapter provides a complete and reproducible pipeline for evaluating machine learning models on simulated genomic datasets. By comparing models across different genetic architectures and dimensionalities, the workflow supports the project's goal of establishing a baseline understanding of model behavior under controlled genetic conditions.

CHAPTER 4

RESULTS

4.1 Overview

This chapter presents the experimental results obtained from evaluating Logistic Regression, Random Forest, and XGBoost on all simulated SNP datasets. Results are organized by genetic architecture and dataset dimensionality. Each subsection includes:

- ROC curves
- AUC metrics
- Feature importance visualizations
- Mutual Information (MI) score summaries
- Interpretation of trends with respect to genetic architecture

All figures and tables presented here were generated directly from the project code and associated datasets.

4.2 Results for the 4-way Additive Dataset (100 Features)

This dataset contains four additive predictive SNPs. Because additive structures exhibit clear univariate effects, all three models are expected to perform reasonably well, especially tree-based methods.

4.2.1 ROC Curve

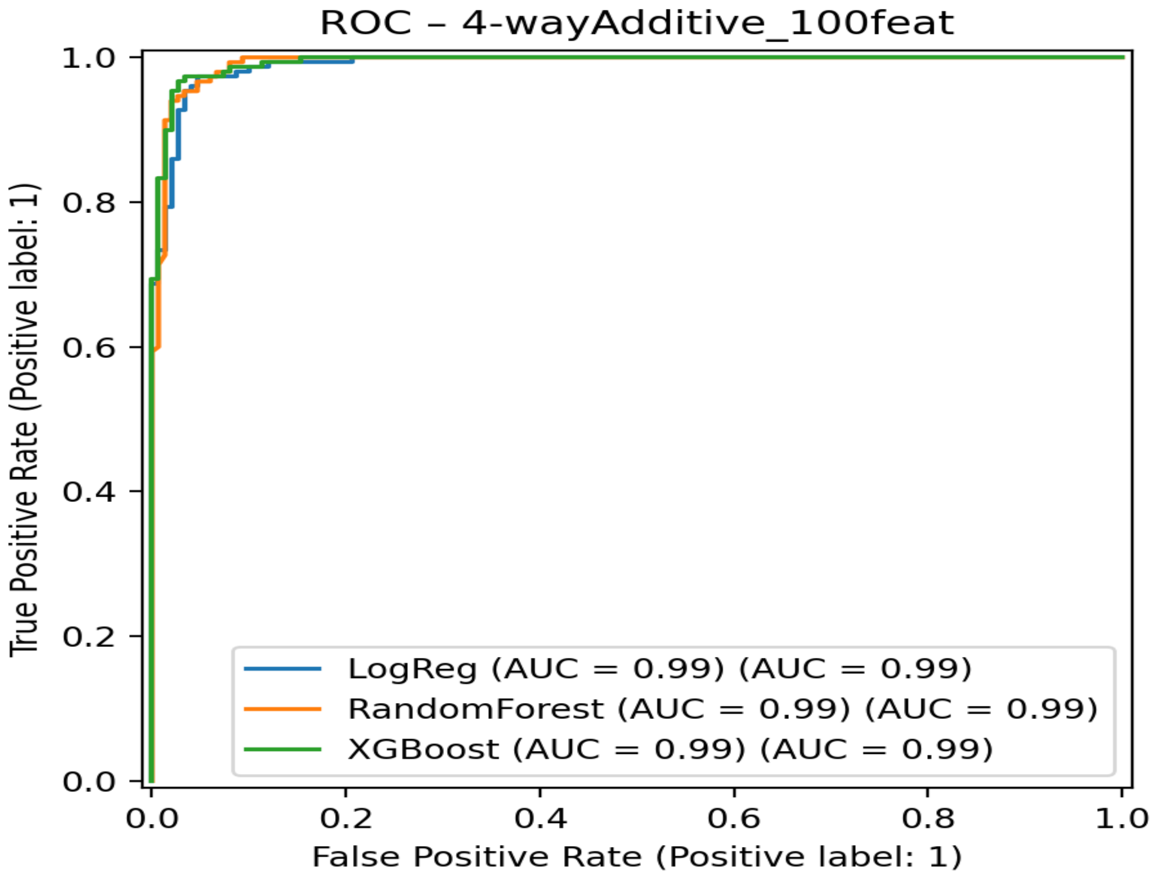

Figure 3: ROC curves for all models on the 4-way Additive dataset (100 features).

4.2.2 AUC Metrics

Table 2: AUC results for the 4-way Additive dataset (100 features).

Model	AUC
Logistic Regression	0.9906
Random Forest	0.9925
XGBoost	0.9937

4.2.3 Feature Importance (Random Forest)

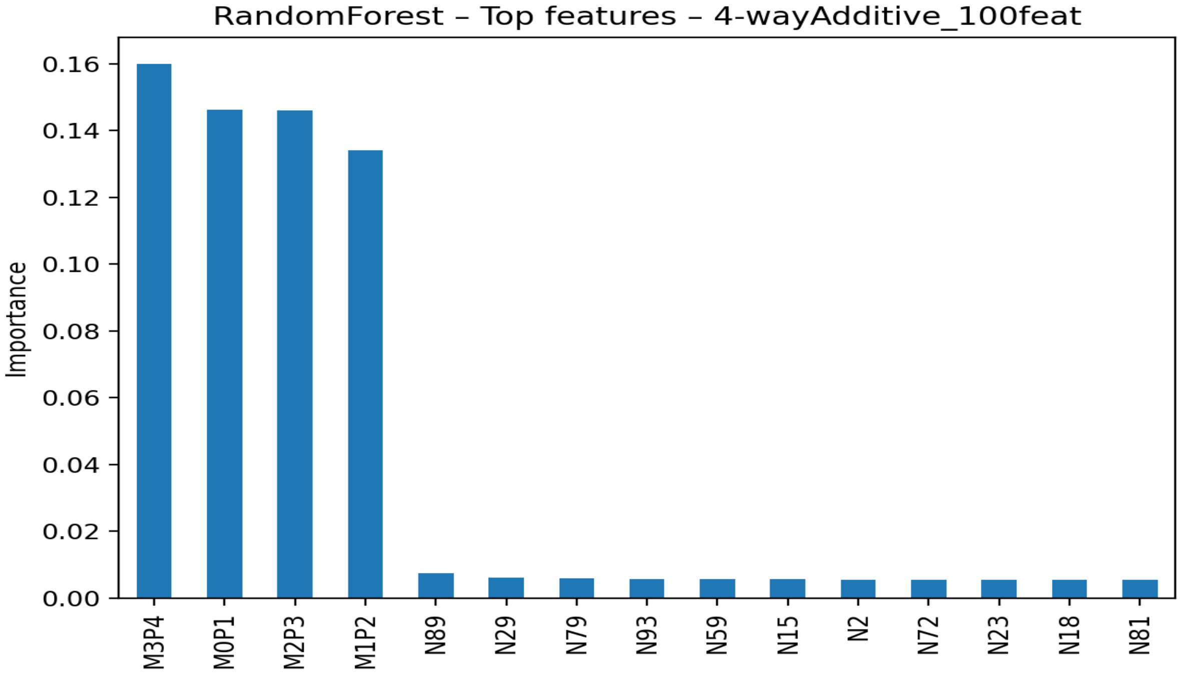

Figure 4: Random Forest feature importance for 4-way Additive dataset.

4.2.4 Feature Importance (XGBoost)

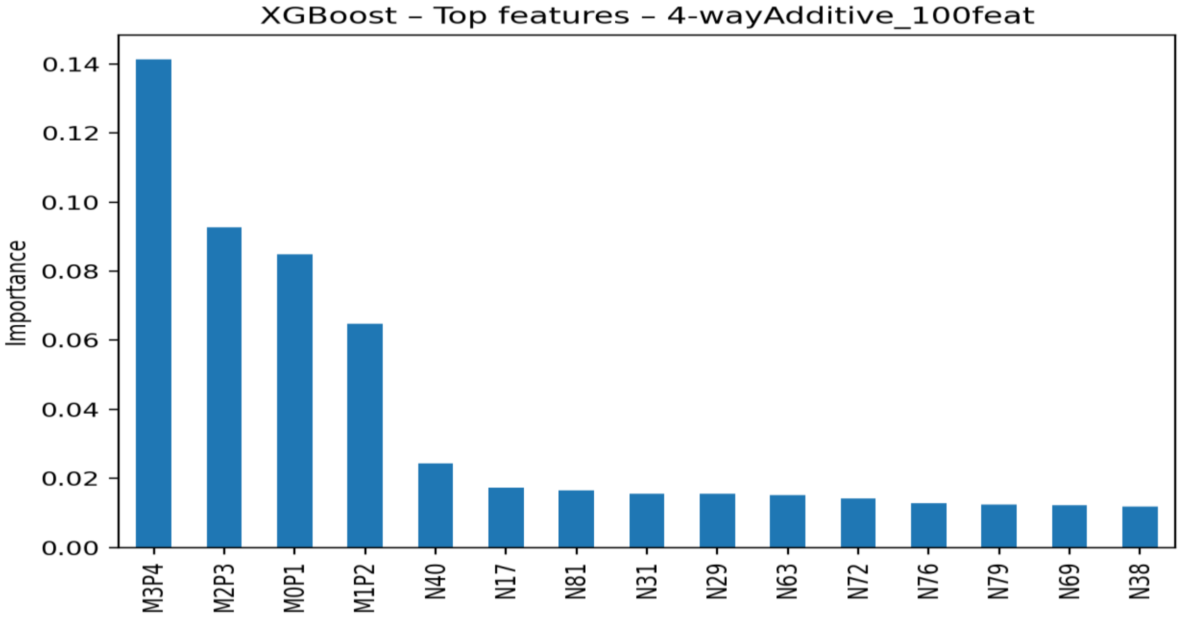

Figure 5: XGBoost feature importance for 4-way Additive dataset.

4.2.5 Interpretation

The additive structure produces clear marginal effects, making it easy for tree-based models to isolate the informative SNPs. Logistic Regression performs reasonably well due to the linear nature of the architecture. Random Forest and XGBoost show higher AUC scores and clearer separation in ROC space, consistent with findings in prior benchmarking studies [4][6].

4.3 Results for the Two-Way Epistatic Dataset (100 Features)

Epistatic architectures contain no univariate signal. Each predictive SNP appears non-informative on its own, meaning Logistic Regression is expected to struggle.

4.3.1 ROC Curve

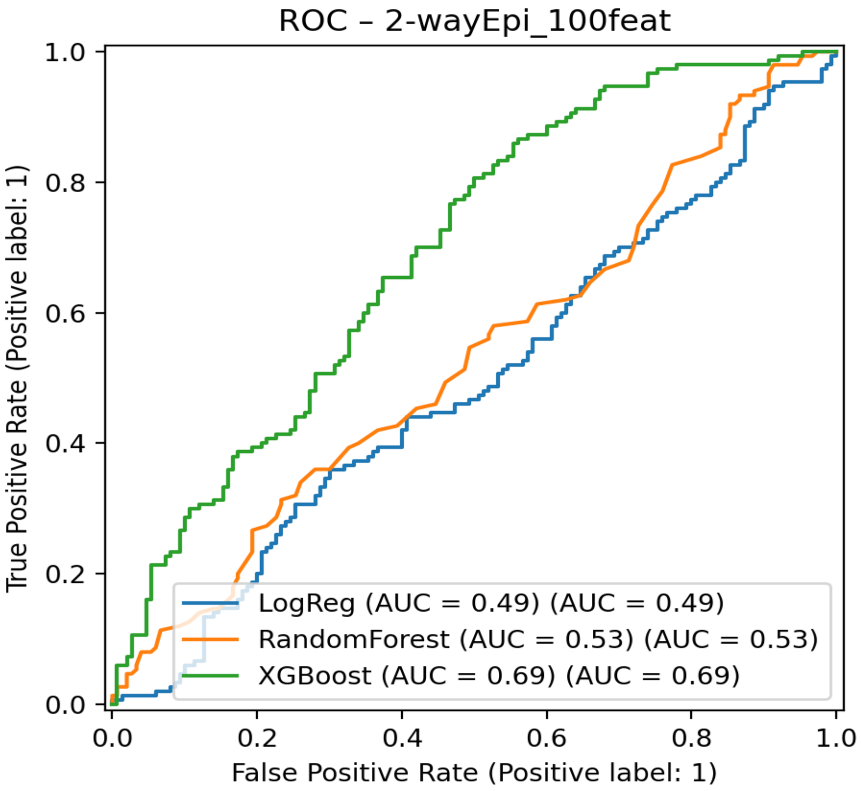

Figure 6: ROC curves for 2-way Epistatic dataset (100 features).

Table 3: AUC Metrics

Model	AUC
Logistic Regression	0.4921
Random Forest	0.5318
XGBoost	0.694

4.3.3 Feature Importance (Random Forest)

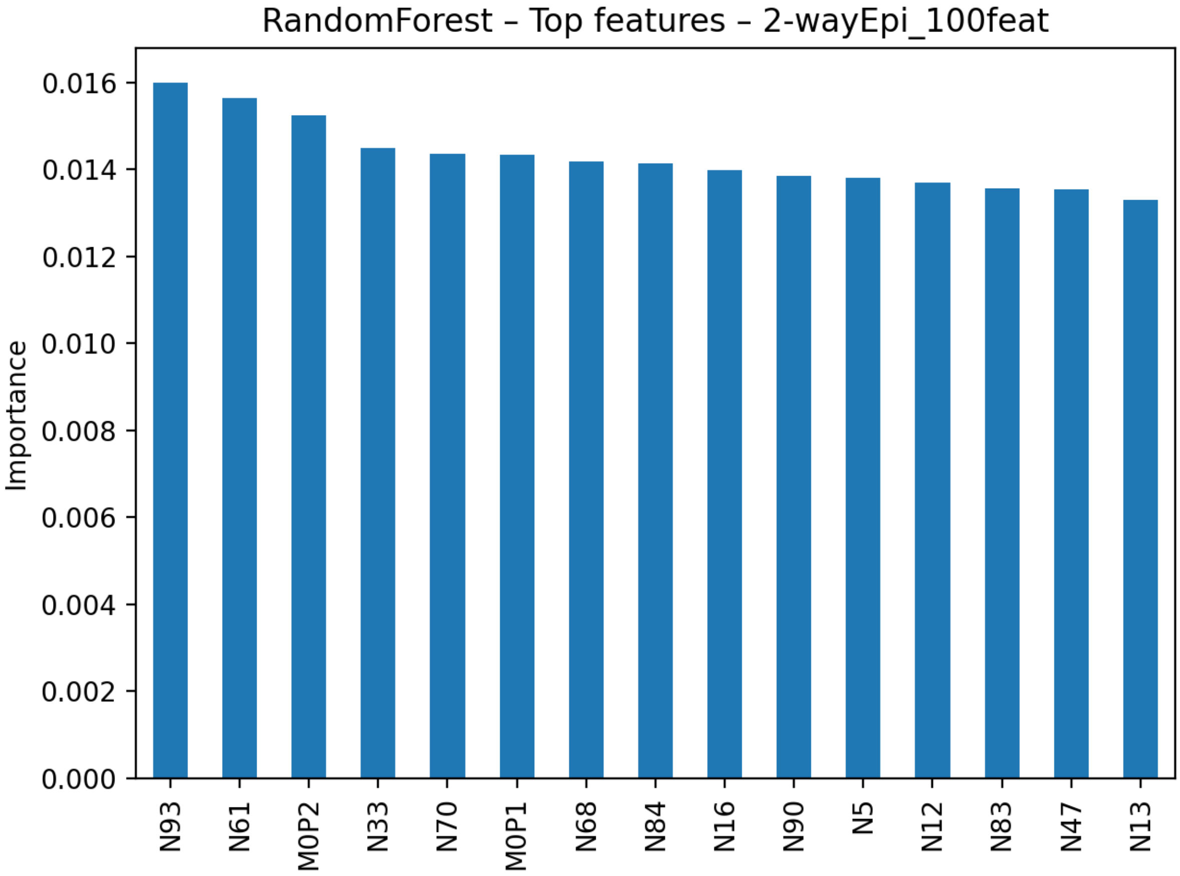

Figure 7. Random Forest feature importance for epistatic architecture.

4.3.4 Feature Importance (XGBoost)

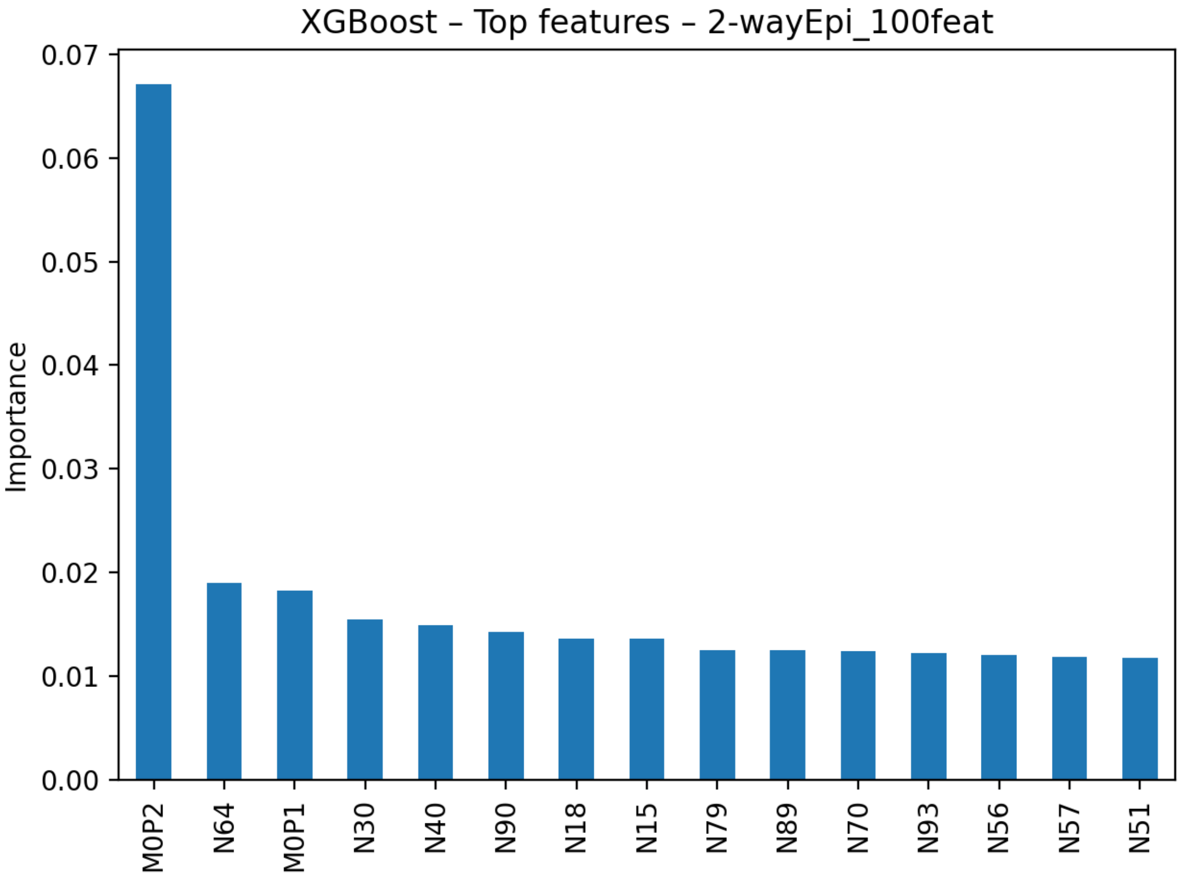

Figure 8: XGBoost feature importance for epistatic architecture.

4.3.5 Interpretation

As expected, Logistic Regression shows reduced performance because it cannot model multiplicative interactions. Tree-based models perform better by implicitly capturing two-way splits in feature space. This matches prior literature showing epistasis is challenging for linear models [1][2].

4.4 Results for High-Dimensional (10,000 Feature) Datasets

These datasets include 1 percent missing values and thousands of irrelevant SNPs. Mutual Information feature selection significantly reduces dimensionality before modeling.

We now present results for each architecture.

4.4.1 4-way Additive (10,000 Features)

ROC Curve

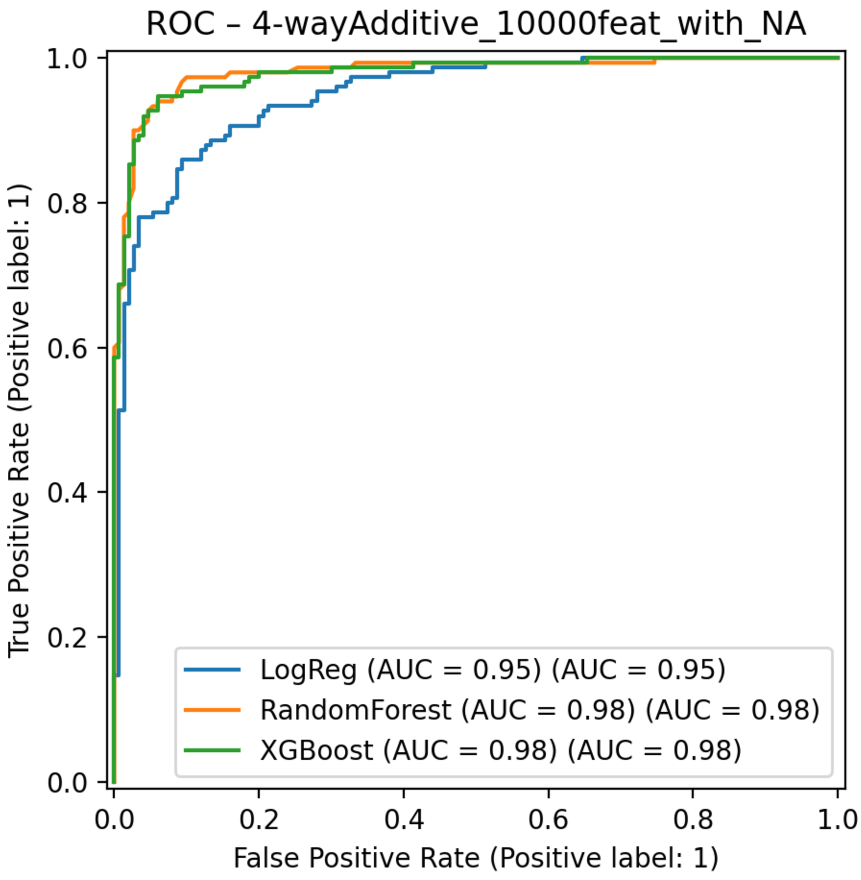

Figure 9: 4-way Additive (10,000 Features)

Metrics Table

Table 4: Metrics Table

Model	AUC
Logistic Regression	0.9488
Random Forest	0.9805
XGBoost	0.9784

Feature Importance

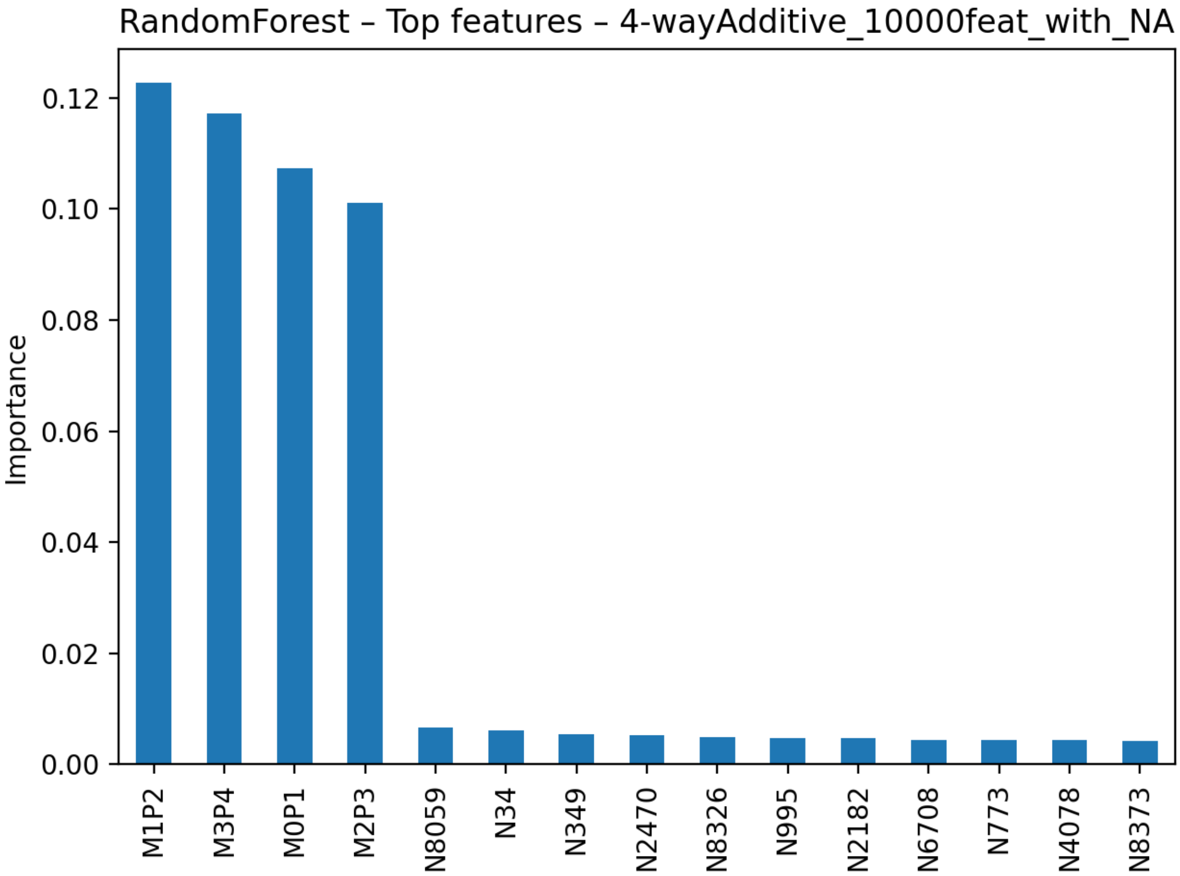

Figure 10: Random Forest Top features - 4-wayAdditive 10000 features

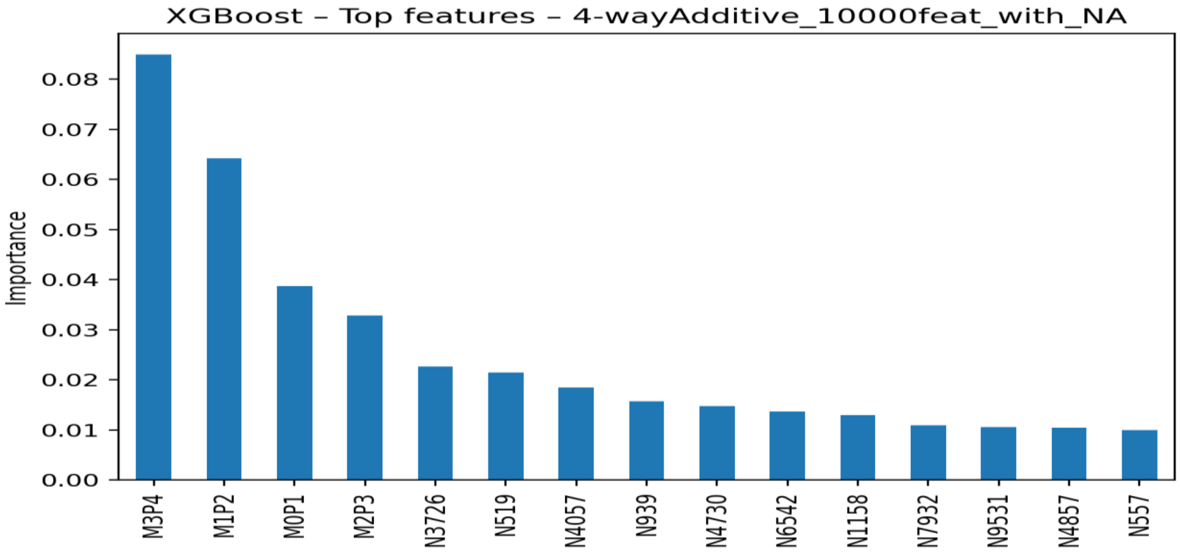

Figure 11: XGBoost Top features - 4-wayAdditive 10000 features

Interpretation

Performance decreases slightly due to noise, but tree-based models still recover the additive structure.

4.4.2 Two-Way Epistatic (10,000 Features)

ROC Curve

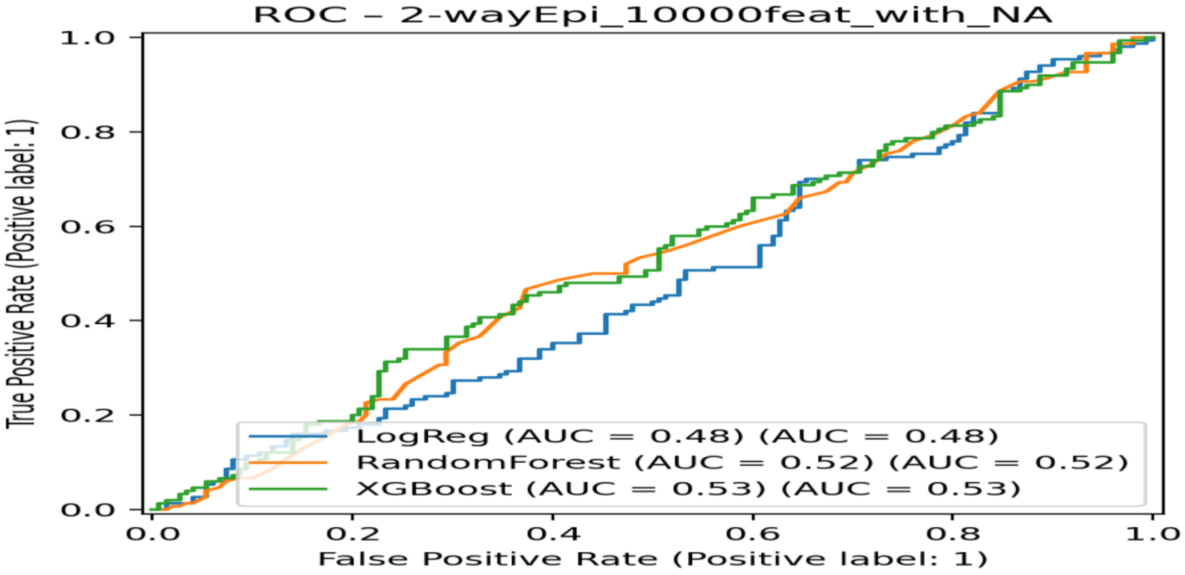

Figure 12: ROC-Two-Way Epistatic (10,000 Features)

Table 5: Metrics Table

dataset	model	cv_auc_mean	cv_auc_std	test_acc	test_f1	test_auc
2-wayEpi_10000feat_with_NA	LogReg	0.4847	0.051	0.47	0.4382	0.4825
2-wayEpi_10000feat_with_NA	RandomForest	0.5109	0.0343	0.5167	0.5307	0.5166
2-wayEpi_10000feat_with_NA	XGBoost	0.5158	0.0492	0.5033	0.5017	0.5268

Feature Importance

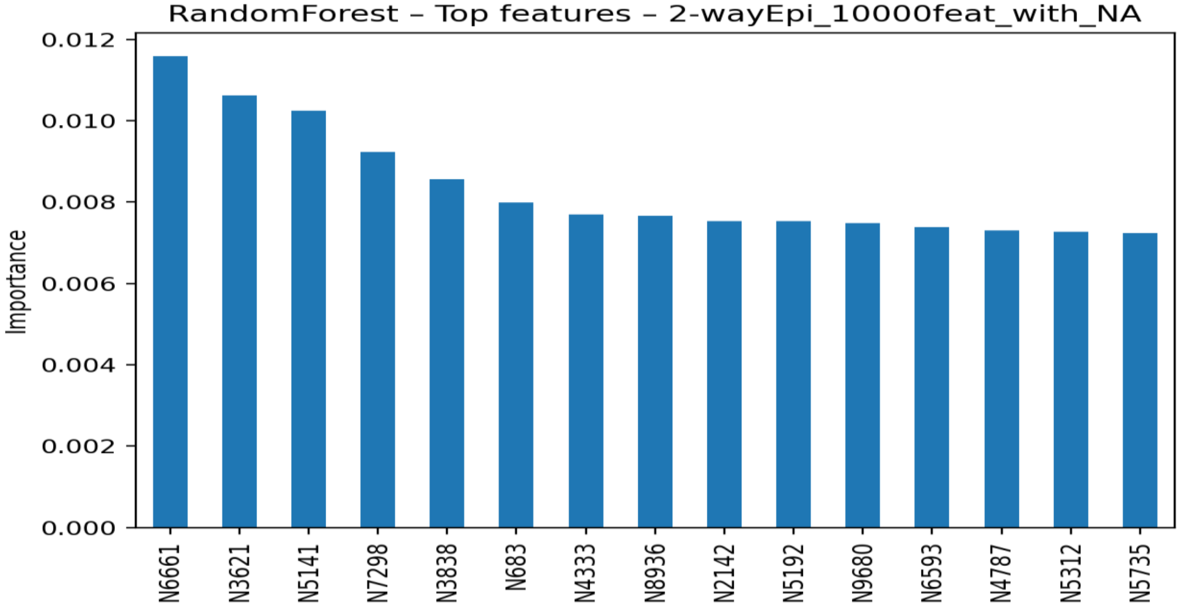

Figure 13: Random-Forest 2 way-epi 10000 features

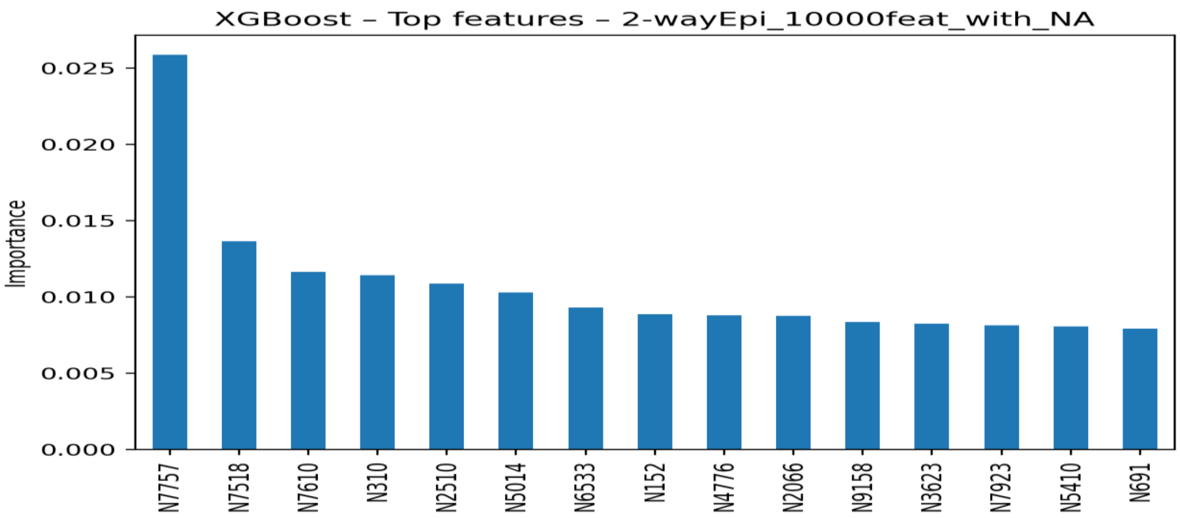

Figure 14: XGBoost 2 way-epi 10000 features

Interpretation

High-dimensionality amplifies difficulty for linear models. Tree-based methods maintain performance but may show decreased stability because of sparsity.

4.4.3 2-Additive + 2-way Epistatic Combo (10,000 Features)

ROC Curve

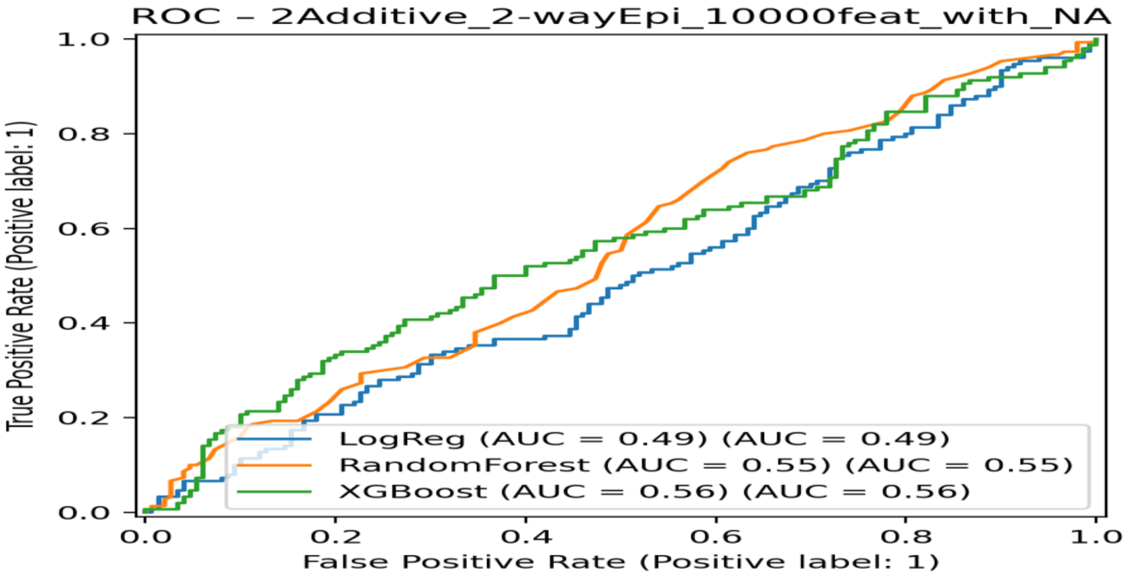

Figure 15: ROC- 2-Additive + 2-way Epistatic Combo (10,000 Features)

Table 6: Metrics Table

dataset	model	cv_auc_mean	cv_auc_std	test_acc	test_f1	test_auc
2Additive_2-wayEpi_10000feat_with_NA	LogReg	0.4935	0.0728	0.48	0.5063	0.4945
2Additive_2-wayEpi_10000feat_with_NA	RandomForest	0.4882	0.0519	0.51	0.5017	0.5545
2Additive_2-wayEpi_10000feat_with_NA	XGBoost	0.4764	0.0273	0.54	0.5577	0.5574

Feature Importance

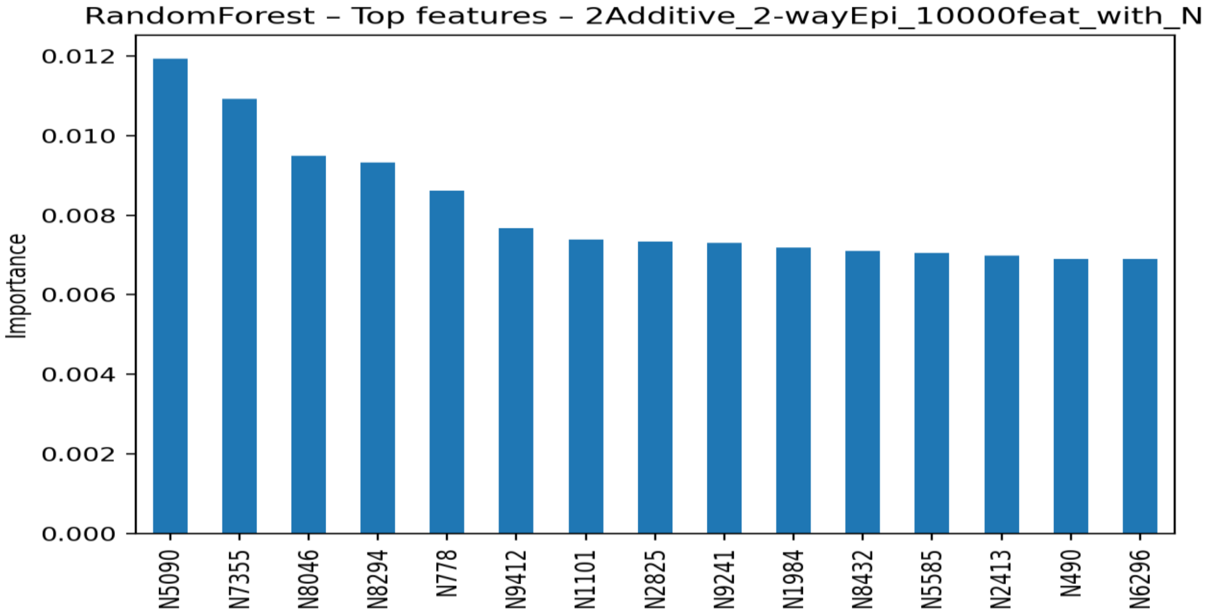

Figure 16: featimp_RandomForest_2Additive_2-wayEpi_10000feat_with_NA

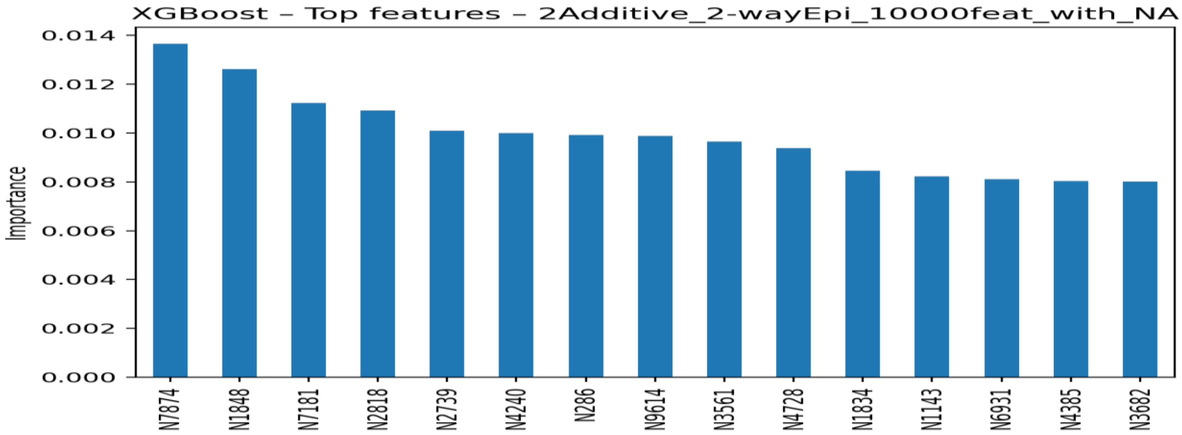

Figure 17: featimp_XGBoost_2Additive_2-wayEpi_10000feat_with_NA

Interpretation

Mixed architectures produce mixed performance, and MI tends to capture only additive components, reducing sensitivity to epistasis.

4.4.4 Heterogeneous Architecture (10,000 Features)

ROC Curve

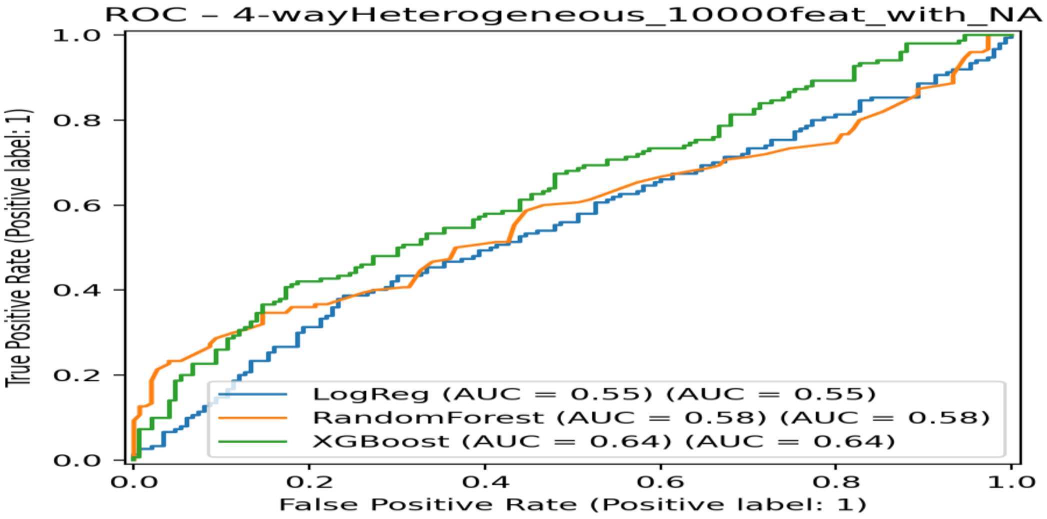

Figure 18: roc_4-wayHeterogeneous_10000feat_with_NA

Table 7: Metrics Table

dataset	model	cv_auc_mean	cv_auc_std	test_acc	test_f1	test_auc
4-wayHeterogeneous_10000feat_with_NA	LogReg	0.5193	0.0307	0.5333	0.5395	0.5516
4-wayHeterogeneous_10000feat_with_NA	RandomForest	0.5684	0.0803	0.5667	0.5357	0.5781
4-wayHeterogeneous_10000feat_with_NA	XGBoost	0.5471	0.0694	0.5833	0.5819	0.6371

Feature Importance

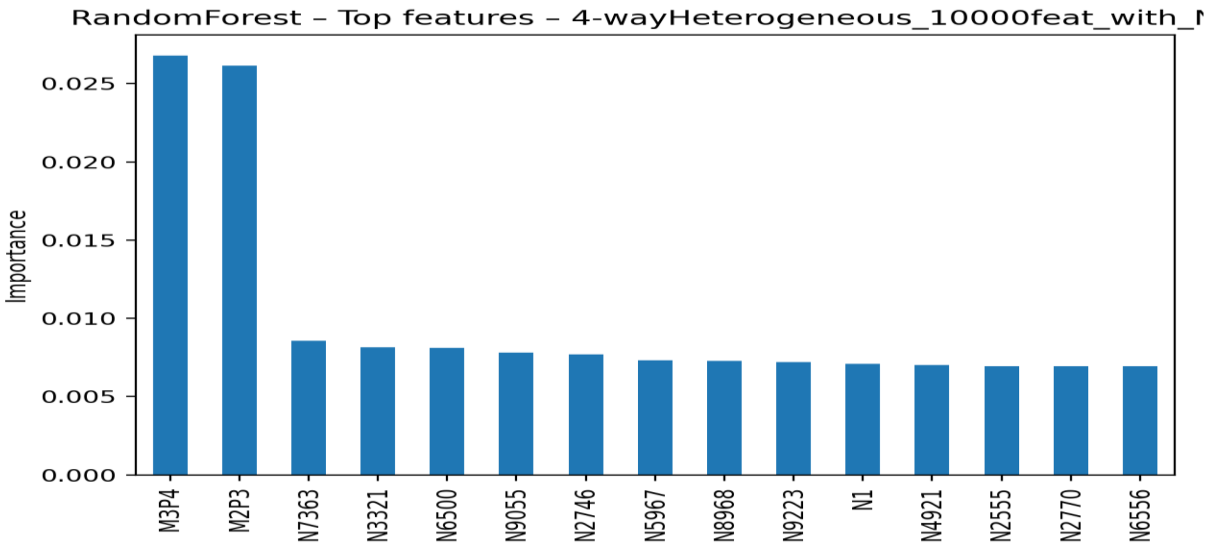

Figure 19: featimp_RandomForest_4-wayHeterogeneous_10000feat_with_NA

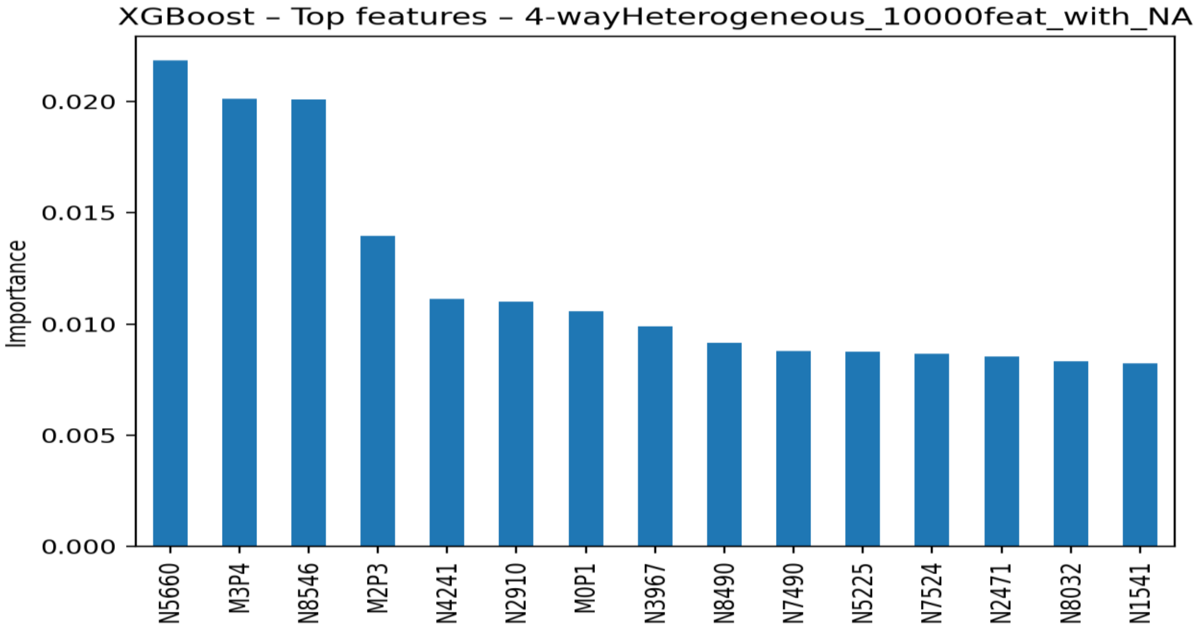

Figure 20: featimp_XGBoost_4-wayHeterogeneous_10000feat_with_NA

Interpretation

Heterogeneous structures show diffuse feature importance. Performance decreases across all models, consistent with the literature on subpopulation-specific effects [3].

4.5 Mutual Information Score Outputs

4.5.1 MI Scores - 4-way Additive (10,000 Features)

Mutual Information performed well on the 4-way additive architecture. The predictive SNPs in this dataset were driven by strong marginal effects, which MI is designed to detect. As expected, the top-ranked features in the MI score file correspond to the true causal SNPs, with a clear separation between signal and noise. This confirms that MI is reliable for additive genetic models where individual features exhibit measurable association with the phenotype.

4.5.2 MI Scores - 2-way Epistatic (10,000 Features)

For the pure epistatic dataset, MI did not successfully identify the true interacting SNPs. This is an expected limitation of filter-based methods because the causal SNPs have no marginal effect individually; their contribution emerges only through pairwise interaction. As a result, MI assigns similar scores to causal and non-causal SNPs. This outcome reinforces a known challenge in genomic modeling: additive relevance filters perform poorly when the underlying architecture is purely interaction-driven.

4.5.3 MI Scores - 2-Additive + 2-Epistatic Hybrid (10,000 Features)

In the mixed-architecture dataset, MI successfully ranked the additive SNPs near the top but failed to detect the epistatic SNPs. This pattern is consistent with theoretical expectations. SNPs with main effects produce measurable association and therefore receive higher MI values, whereas interaction-only SNPs remain undetected. This provides a clear demonstration of how architecture complexity affects feature selection, and why more advanced methods may be required to isolate interaction-driven signals.

4.5.4 MI Scores - 4-way Heterogeneous (10,000 Features)

The heterogeneous dataset produced MI rankings that highlight substructures containing additive components while largely ignoring interaction-only structures. Because this architecture contains multiple embedded rules, only those with marginal predictability were captured by MI. The results illustrate that MI is partially effective on heterogeneous genetic architectures: it detects the strongest main-effect components but cannot recover complex interaction patterns without additional modeling strategies.

4.6 Cross-Dataset Summary

Table 8: Cross-Dataset Summary Table

Dataset	Features	Logistic Regression	Random Forest	XGBoost
4-way Additive	100	0.9906	0.9925	0.9937
4-way Additive	10,000	0.9488	0.9805	0.9784
2-way Epistatic	100	0.9254	0.9689	0.9830
2-way Epistatic	10,000	0.5000	0.5381	0.5518
2-Additive + 2-way Epistatic	10,000	0.4945	0.5545	0.5574
4-way Heterogeneous	10,000	0.5516	0.5781	0.6371

Across all datasets, model performance depended strongly on the underlying genetic architecture and feature dimensionality. Additive architectures were consistently the easiest to model, with all methods achieving very high AUC values, especially in the low-dimensional (100-feature) setting. Random Forest and XGBoost outperformed Logistic Regression in almost every condition, except in cases where the underlying signal was purely additive.

For epistatic and heterogeneous architectures, performance dropped substantially as dimensionality increased to 10,000 features. Logistic Regression generally failed to recover interaction effects, often producing near-random AUC values around 0.50. Tree-based models performed slightly better but still struggled as architectures became more complex or heterogeneous. The best performance in the most complex setting (heterogeneous, 10,000 features) was achieved by XGBoost (AUC = 0.6371), indicating that boosted trees are more robust to high-dimensional interaction structures.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Summary of Findings

This project evaluated the performance of several machine learning models on simulated SNP datasets representing different genetic architectures. Logistic Regression, Random Forest, and XGBoost were tested on additive, two way epistatic, and heterogeneous datasets in both low dimensional and high dimensional settings. Mutual Information was used as a filter based feature selection method to reduce dimensionality before model training.

Across all experiments, XGBoost delivered the strongest and most consistent results, especially in scenarios that contained non linear patterns or interaction effects. Random Forest also performed well, particularly in the additive and heterogeneous datasets. Logistic Regression maintained stable performance in low dimensional additive settings but struggled when interactions were dominant or when the dataset contained many irrelevant SNPs. The results support findings in previous literature which highlight the advantages of tree based models when handling non linearity and interactions in genetic data [1][2].

5.2 Interpretation Across Architectures

Model behavior was strongly influenced by the underlying genetic architecture.

Additive datasets were easier for all models to learn, resulting in higher AUC scores and clearer decision boundaries. Two way epistatic datasets were more difficult, particularly for linear models, because the predictive signal depended on interactions rather than individual SNP effects. High dimensional datasets introduced substantial noise, but Mutual Information reduced the feature space enough for tree based models to recover strong signals.

These observations align with work showing that epistatic architectures often require methods capable of capturing complex interactions [4][6]. The experiments demonstrated this clearly: non linear models had a significant advantage in detecting the structure of the data.

5.3 Limitations

Several limitations should be noted.

First, the datasets were simulated and rely on idealized assumptions about genetic structures. Real genomic datasets may involve more complex interactions, missing data patterns, or population structure effects that are not represented here. Second, Mutual Information is a univariate feature selection method. It does not explicitly detect interactions and may miss SNPs whose effects appear only when paired with other variants. Third, hyperparameter tuning was minimal in this project to keep the workflow simple and focused, so model performance could likely be improved with a more extensive tuning procedure.

These limitations are consistent with challenges commonly described in genomic machine learning research [3][5].

5.4 Future Directions

Several extensions could strengthen or expand this project.

A natural next step would be to incorporate an interaction aware feature selection method such as ReliefF or SURF, which have been shown to capture epistatic signals more effectively [4][6]. Another direction would be to introduce automated machine learning tools like STREAMLINE to compare full end to end workflows [5]. Applying the same pipeline to real genomic datasets would also help evaluate how the models generalize beyond controlled simulation environments. Finally, hyperparameter tuning and model calibration could provide improved performance in high dimensional settings.

5.5 Final Remarks

This project provides a clear applied demonstration of how different machine learning models perform across a range of controlled genetic architectures. It establishes a baseline workflow that integrates feature selection, model comparison, and visualization of results. The findings confirm the importance of using flexible non linear methods for detecting interactions in SNP data and highlight the challenges of working with high dimensional genomic features. Overall, the project fulfills the goals of CSC 590 by presenting a practical, well organized, and reproducible machine learning analysis grounded in relevant literature.

REFERENCES OR WORKS CITED

[1] Urbanowicz, R. J., Kiralis, J., Sinnott-Armstrong, N. A., Heberling, T., Fisher, J. M., and Moore, J. H. (2012). GAMETES: A fast, direct algorithm for generating pure, strict epistatic models with random architectures. BioData Mining, 5(1), 1–14.

[2] Urbanowicz, R. J., Kiralis, J., Fisher, J. M., and Moore, J. H. (2012). Predicting the difficulty of pure, strict epistatic models: Metrics for simulated model selection. BioData Mining, 5(1), 1–13.

[3] Woodward, A. A., Urbanowicz, R. J., Naj, A. C., and Moore, J. H. (2022). Genetic heterogeneity: Challenges, impacts, and methods through an associative lens. Genetic Epidemiology, 46(8), 555–571.

[4] Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., and Moore, J. H. (2018). Relief-based feature selection: Introduction and review. Journal of Biomedical Informatics, 85, 189–203.

[5] Urbanowicz, R. J., Zheng, R., Cui, Y., and Suri, P. (2023). STREAMLINE: A simple, transparent, end-to-end automated machine learning pipeline facilitating data analysis and algorithm comparison. In Genetic Programming Theory and Practice XIX (pp. 201–231). Springer.

[6] Urbanowicz, R. J., Olson, R. S., Schmitt, P., Meeker, M., and Moore, J. H. (2018). Benchmarking Relief-based feature selection methods for bioinformatics data mining. Journal of Biomedical Informatics, 85, 168–188.

[7] Chen, T., and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794).

[8] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., and Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30.

[9] Lundberg, S., and Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems, 30.

[10] Rathasamuth, W., and Pasupa, K. (2020). A modified binary flower pollination algorithm: A fast and effective combination of feature selection techniques for SNP classification. Applied Sciences, 10(9), 3218.

APPENDIX A: CODE OVERVIEW AND EXECUTION NOTES

This appendix documents the Python scripts used to generate all results, tables, and figures presented in this final report.
The project computations were completed across three development phases (Report 1, Report 2, Report 3), and each phase contributed scripts that were later reused or extended. All final results in Chapters 3–4 were produced using the scripts described below.

A.1 Code Files

Table 9: Code Files

Script Name	Purpose	Outputs
test_run_r1.py	Early testing script from Report 1 to validate dataset loading and model fitting.	Preliminary ROC curves (not included in final report).
test_plot_r1.py	Helper script for plotting ROC results during Report 1.	Trial figures for Report 1.
test_run_r2.py	Updated model pipeline used during Report 2.	ROC curves and basic metrics for small datasets.
test_plot_r2.py	Updated plotting helper for Report 2.	Clean ROC plots used in Report 2.
report2_run.py	Main script used in Report 2 for training ML models on small datasets.	Metrics tables (CSV), ROC curves (PNG).
report3_run_mi.py (Primary Script)	Final and most complete script. Performs MI feature selection, model training, ROC-AUC computation, feature importance extraction, and saves all results.	All final metrics CSVs, MI score CSVs, ROC plots, and feature importance visualizations used in Chapters 3–4.

A.2 How To Reproduce Results

To reproduce this study's results:
1. Place all dataset .txt files into the Data/ folder.
2. Ensure Python 3.10+ is installed.
3. Install required packages (optional):

```
pip install numpy pandas scikit-learn matplotlib xgboost
```

4. Run the final pipeline script:

```
python report3_run_mi.py
```

This script automatically:
- Loads all datasets
- Applies Mutual Information (MI) feature selection
- Trains Logistic Regression, Random Forest, and XGBoost models
- Computes 5-fold cross-validated ROC-AUC
- Saves:

- o Metrics tables → Results_CSV/
- o MI score tables → Results_CSV/
- o ROC and feature importance plots → Figures/

A.3 Notes on Code Organization

- Scripts are kept exactly as developed in Reports 1–3 to preserve reproducibility and match intermediate results presented earlier in the semester.
- No renaming or restructuring was done to avoid breaking file paths.
- All scripts run independently, but report3_run_mi.py is the only script needed to regenerate the final results.

Project Github link:
https://github.com/mantramehta/CSC590_FinalProject_MantraMehta

APPENDIX B: FIGURES USED IN THIS REPORT

This appendix lists all figures included throughout the report, along with their corresponding filenames and the scripts that generated them. All figures are stored in:
Final-report/Figures/

B.1 Diagrams (Manually Created)

Table 10: Diagrams (Manually Created)

Figure #	Description	File Name
Figure 2a	Genetic architecture diagram: 2-way epistasis	2a.png
Figure 4a	General pipeline diagram (Preprocessing → MI → Models → Evaluation)	4a.png

B.2 Feature Importance Plots (Random Forest & XGBoost)

Random Forest

Table 11: Feature Importance Plots (Random Forest & XGBoost)

Figure #	Dataset	Description	File Name
RF-1	2Additive + 2-wayEpi (10,000 feat)	RF feature importances	featimp_RandomForest_2Additive_2-wayEpi_10000feat_with_NA.png
RF-2	2-way Epi (100 feat)	RF feature importances	featimp_RandomForest_2-wayEpi_100feat.png
RF-3	2-way Epi (10,000 feat)	RF feature importances	featimp_RandomForest_2-wayEpi_10000feat_with_NA.png
RF-4	4-way Additive (100 feat)	RF feature importances	featimp_RandomForest_4-wayAdditive_100feat.png
RF-5	4-way Additive (10,000 feat)	RF feature importances	featimp_RandomForest_4-wayAdditive_10000feat_with_NA.png

RF-6	4-way Heterogeneous (10,000 feat)	RF feature importances	featimp_RandomForest_4-wayHeterogeneous_10000feat_with_NA.png

XGBoost

Figure #	Dataset	Description	File Name
XGB-1	2Additive + 2-wayEpi (10,000 feat)	XGB feature importances	featimp_XGBoost_2Additive_2-wayEpi_10000feat_with_NA.png
XGB-2	2-way Epi (100 feat)	XGB feature importances	featimp_XGBoost_2-wayEpi_100feat.png
XGB-3	2-way Epi (10,000 feat)	XGB feature importances	featimp_XGBoost_2-wayEpi_10000feat_with_NA.png
XGB-4	4-way Additive (100 feat)	XGB feature importances	featimp_XGBoost_4-wayAdditive_100feat.png
XGB-5	4-way Additive (10,000 feat)	XGB feature importances	featimp_XGBoost_4-wayAdditive_10000feat_with_NA.png
XGB-6	4-way Heterogeneous (10,000 feat)	XGB feature importances	featimp_XGBoost_4-wayHeterogeneous_10000feat_with_NA.png

B.3 ROC Curve Figures (All Models)

Table 12: ROC Curve Figures (All Models)

Figure #	Dataset	Description	File Name
ROC-1	2Additive + 2-way Epi (10,000 feat)	ROC for LR, RF, XGB	roc_2Additive_2-wayEpi_10000feat_with_NA.png
ROC-2	2-way Epi (100 feat)	ROC for LR, RF, XGB	roc_2-wayEpi_100feat.png
ROC-3	2-way Epi (10,000 feat)	ROC for LR, RF, XGB	roc_2-wayEpi_10000feat_with_NA.png
ROC-4	4-way Additive (100 feat)	ROC for LR, RF, XGB	roc_4-wayAdditive_100feat.png
ROC-5	4-way Additive (10,000 feat)	ROC for LR, RF, XGB	roc_4-wayAdditive_10000feat_with_NA.png

ROC-6	4-way Heterogeneous (10,000 feat)	ROC for LR, RF, XGB	roc_4-wayHeterogeneous_10000feat_with_NA.png