**HLA-G gene polymorphisms as Predictors of Survival in Colorectal Cancer: A Unified Machine Learning Approach**

***Supplementary Data***

1. **Machine Learning Algorithms**

This section provides detailed description on each Machine Learning algorithm that have been used for the purpose of our study.

* 1. **Decision Trees (DT)**

A decision tree is a fundamental machine learning model known for its simplicity and interpretability. Structurally, it resembles an inverted tree where each internal node represents a decision rule based on a feature, and each branch divides data into subsets based on those rules. The process continues down the "branches" of the tree until reaching the "leaves," which represent the model's final classifications or predictions.

Decision trees are highly intuitive: each step in the tree is a decision that separates data, making them easy to interpret and explain. However, their flexibility also makes them prone to overfitting, especially when they grow too deep, as they may capture noise or overly specific details from the training data that don’t generalize well to unseen data.

***Tuned Parameters***

*Criterion* (Gini or entropy): Specifies the metric used to evaluate each split in the data, controlling how the model decides to branch. "Gini" aims to minimize impurity in the nodes, while "entropy" focuses on reducing information gain at each split.

*Max depth*: Restricts the depth of the tree, preventing it from growing too complex and overfitting by controlling how many decision levels are allowed.

*Min samples split*: Sets a minimum threshold for the number of samples required to create a split, limiting further branching when data becomes sparse to avoid unnecessary complexity.

* 1. **Random Forest (RF)**

Random Forest is a powerful ensemble learning method that combines the predictions of multiple decision trees to improve accuracy and reduce overfitting. Unlike a single decision tree, which can be prone to capturing noise and overfitting, a Random Forest builds each tree on a randomly selected subset of features and data samples. This randomness adds diversity among the trees, ensuring that individual trees don’t rely on the same patterns, which in turn, makes the overall model more robust and accurate.

Imagine Random Forest as a team of experts, each analyzing the data from slightly different perspectives and sources. By averaging their predictions, the model gains both stability and accuracy, as the group’s consensus is generally more reliable than any single tree's opinion. This approach is particularly effective for handling large, complex datasets and is resistant to the overfitting issues that can impact individual decision trees.

***Tuned Parameters***

*Number of estimators*: Controls the number of trees in the forest. More trees generally improve accuracy but increase computational requirements.

*max depth*, which restricts how deep each tree can grow, and *min samples split*, which limits branching to maintain generalization.

* 1. **AdaBoost (Adaptive Boosting)**

AdaBoost is an ensemble learning algorithm that improves the accuracy of predictions by combining multiple "weak learners" (simpler models), often in the form of decision stumps. These stumps are very basic decision trees with only one split point, consisting of a single root node and two leaf nodes, making them highly interpretable but limited in complexity.

In AdaBoost, each new decision stump in the sequence focuses on correcting the errors made by the previous ones. During the training process, the algorithm assigns more weight to the data points that were previously misclassified, encouraging each subsequent stump to learn from the mistakes of its predecessors. As a result, the model places more emphasis on challenging cases, which makes it effective for datasets with varying distributions or imbalanced classes.

***Tuned Parameters***

*Number of estimators*: Controls how many weak learners (stumps) are combined to build the final model.

* 1. **Gradient Boosting (GB)**

Gradient Boosting is an ensemble technique that, like AdaBoost, builds a powerful model by combining multiple weaker models sequentially, most commonly decision trees. However, while AdaBoost focuses on correcting misclassified examples, Gradient Boosting optimizes by minimizing the overall residual error—essentially, the difference between the model’s predictions and the actual values—using a gradient descent approach.

In practice, Gradient Boosting starts by creating an initial, simple model, often a single decision tree. Then, each new tree is trained to predict the residuals (errors) of the previous trees, which allows to refine the model iteratively. By continually correcting errors from prior iterations, each stage improves the model's accuracy, gradually reducing the residual error. This makes Gradient Boosting particularly effective for complex datasets where accuracy is crucial.

***Tuned Parameters***

Number of estimators: Specifies the total number of trees in the ensemble, balancing model accuracy with computation time.

Max depth: Limits how complex each tree can become, preventing overfitting by controlling the depth of splits. Shallower trees capture more general patterns, while deeper trees capture finer details.

* 1. **Stochastic Gradient Boosting (SGB)**

Stochastic Gradient Boosting (SGB) is an extension of the traditional Gradient Boosting method, introducing an additional layer of randomness that helps reduce overfitting and improve generalization to new data. In SGB, each new tree is trained on a random subset of the data at each iteration, rather than on the entire dataset as in standard Gradient Boosting. By using different subsets of data, SGB injects diversity into the model, which often leads to better performance on unseen data and a more robust model overall.

This stochastic element makes SGB especially effective in situations where datasets are large or have noisy patterns. The injected randomness acts as a regularizer, balancing the model's accuracy and its ability to generalize.

***Tuned Parameters***

*Number of estimators*: Defines the total number of trees in the model, balancing training time and accuracy.

*Max depth*: Limits the depth of each tree, preventing excessive complexity and reducing overfitting.

* 1. **Bagging (Bootstrap Aggregating)**

Bagging, short for Bootstrap Aggregating, is an ensemble technique designed to improve model stability and reduce variance by training multiple models on different subsets of the data. Similar to Random Forest, Bagging combines the predictions of several decision trees, but with a key difference: each tree in Bagging generally uses *all available features*, while Random Forest typically limits the features considered at each split. This allows each tree in Bagging to make decisions based on the full feature set, which can be beneficial when there’s limited feature redundancy.

Each tree is trained on a unique "bootstrap" sample of the dataset—these samples are generated by drawing data points with replacement, so some points may appear multiple times within a sample while others may be left out entirely. This randomized sampling approach introduces variability across the trees, reducing the likelihood of overfitting and enabling the model to generalize better to new data.

***Tuned Parameters***

*Number of estimators*: Specifies the number of trees to include in the ensemble. Increasing this parameter generally improves stability and accuracy.

*Max samples*: Controls the size of the bootstrap sample for each tree, representing the percentage or absolute number of data points drawn from the original dataset.

*Max features*: Determines the maximum number of features each tree has access to, allowing for further control over complexity and variance in the model.

* 1. **Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a robust machine learning model designed for classification and regression tasks. Its primary objective is to find the optimal "hyperplane" that separates data points belonging to different classes in a way that maximizes the margin—the distance between the closest points of the different classes, known as support vectors. This focus on maximizing the margin helps ensure that the model generalizes well to unseen data.

SVM is particularly effective for high-dimensional datasets, making it a popular choice in fields such as bioinformatics and text classification. However, its computational intensity can make it slower when handling large datasets, as the complexity grows with the number of data points.

Additionally, SVM can use various kernel functions to handle non-linear relationships between classes. By transforming the original feature space into a higher-dimensional space, SVM can find complex boundaries that separate classes effectively.

***Tuned Parameters***

*Kernel type*: Specifies the kernel function to be used, such as linear, polynomial, or Radial Basis Function (RBF). The choice of kernel greatly influences the model's ability to capture complex relationships within the data.

*C (regularization parameter)*: Controls the trade-off between maximizing the margin and minimizing classification errors. A larger C value places greater emphasis on minimizing errors, which may lead to overfitting, while a smaller C encourages a wider margin but can result in misclassifications.

*Gamma (for RBF kernel)*: Defines the influence of a single training example on the decision boundary. A low gamma value creates a smoother boundary, while a high gamma value leads to a more complex and potentially overfitted boundary, as it makes the model sensitive to the training data.

1. **Usage of GridSearchCV to optimize parameters**

GridSearchCV is a powerful tool in the machine learning toolkit, designed to systematically explore a specified range of hyperparameters for a given model. By evaluating all possible combinations of prespecified parameters, it helps identify the optimal settings that lead to the best performance on the dataset in question. This exhaustive approach ensures that no potential configuration is overlooked and maximizes the model's predictive accuracy.

The primary goal of GridSearchCV is to refine and enhance the performance of machine learning models by performing a comprehensive search over parameter combinations. Each combination is assessed based on a specified scoring metric that allows researchers to pinpoint the configuration that yields the highest predictive power.

To illustrate its utility, consider the example of tuning three parameters for a Random Forest model: the number of estimators (the total number of trees in the forest), the maximum depth of each tree, and the minimum samples required for a split. Given sample sets for each parameter, GridSearchCV generates all possible combinations.

For each configuration, it employs cross-validation techniques, such as 5-fold or 10-fold cross-validation, to obtain reliable performance metrics. This approach mitigates the risk of overfitting by ensuring that the evaluation is based on multiple subsets of the data.

The final step involves selecting a scoring metric to evaluate the model's performance. Common choices include accuracy, F1-score, precision, recall, or any custom metric relevant to the specific problem. The parameter set that achieves the best score according to this metric is then identified as the optimal configuration for the model.

### *Application in the Paper*

In this study, the selection of hyperparameters for the machine learning models was guided by the necessity to achieve a balance between model complexity and generalization capability. We carefully chose the range of values for each parameter to explore the full spectrum of possible configurations and enhance model performance.

The parameters considered included splitting criterion, maximum depth, maximum leaf nodes, minimum samples per leaf, and minimum samples split, all aimed at influencing the model's ability to capture patterns in the data while mitigating the risk of overfitting.

For example, the splitting criterion allows the model to prioritize the purity of the nodes, with both Gini and entropy providing distinct yet valuable perspectives on classification quality. The ranges for maximum depth (1 to 10) and maximum leaf nodes (1 to 20) were set to explore varying levels of tree complexity, ensuring a thorough examination of both shallow and deeper trees. Similarly, the parameters for minimum samples per leaf and minimum samples split were extended from 1 to 20, allowing the model to remain flexible in its decision-making process while maintaining robust training dynamics. This comprehensive tuning approach ultimately facilitates the discovery of the most effective model settings.

Table 1 presents the range of values for each parameter setting explored in this study. At the outcome, the best combination of each parameter variant, along with the best cross-validation score, was used to construct the tree models. For training the data, we randomly selected 70% of the total dataset to accomplish the task effectively.

**Table 1.** Proposed ranges for hyperparameters optimization using the GridSearchCV algorithm

|  |  |
| --- | --- |
| **Hyperparameter** | **Ranges of evaluated values**  |
| "splitting criterion": | ["gini","entropy"] |
| "max\_depth": | [1,2,3,4,5,6,7,8,9,10] |
| "max\_leaf\_nodes" | [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20] |
| "min\_samples\_leaf" | [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20] |
| "min\_samples\_split" | [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20] |