Specific parameters for model construction

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | C | class\_weight | decision\_function\_shape | degree | gamma | max\_iter | kernel | - |
| **SVM** | 0.1 | balanced | ovr | 3 | scale | -1 | linear | - |
| ***C***: Regularization parameter; class\_weight: Automatically adjusts the weights of each class when handling class imbalance; ***decision\_function\_shape***: The shape of the decision function, where “ovr” indicates a one-vs-rest strategy, suitable for multi-class problems; ***degree***: The degree of the polynomial kernel function; ***gamma***: The coefficient for the kernel function, which controls the influence radius of the support vectors; ***kernel***: The type of kernel function used, where “linear” indicates a linear kernel; ***max\_iter***: The maximum number of iterations, with -1 indicating no limit on the number of iterations. | | | | | | | | |
|  | C | class\_weight | penalty | solver | dual | max\_iter | - | |
| **Logit** | 0.242 | balanced | L1 | saga | False | 100 |
| ***C***: Inverse regularization strength, primarily used to control the complexity of the model; ***class\_weight***: Used to handle class imbalance; setting it to “balanced” automatically applies weights based on the frequency of the data; ***penalty***: Specifies the type of penalty; ***solver***: Specifies the optimization algorithm used; saga is suitable for optimizing large datasets and supports L1 regularization; ***dual***: Determines whether to solve the problem using the dual or primal form; ***max\_iter***: Specifies the maximum number of iterations to ensure the algorithm converges. | | | | | | | | |
|  | min\_samples\_leaf | min\_samples\_split | max\_depth | n\_estimators | max\_features | max\_samples | max\_leaf\_nodes | - |
| **RF** | 2 | 20 | 20 | 500 | sqrt | 0.5 | 60 | - |
| ***min\_samples\_leaf***: The minimum number of samples required in a leaf node; ***min\_samples\_split***: The minimum number of samples required to split an internal node; ***max\_depth***: The maximum depth of the tree; ***n\_estimators***: The number of trees in the random forest; ***max\_features***: The maximum number of features to consider for each split; ***max\_samples***: The maximum proportion of samples to use; ***max\_leaf\_nodes***: The maximum number of leaf nodes. | | | | | | | | |
|  | learning\_rate | subsample | max\_depth | n\_estimators | colsample\_bytree | reg\_alpha | reg\_lambda | min\_child\_weight |
| **XGBoost** | 0.01 | 0.5 | 15 | 200 | 0.4 | 0.1 | 0.4 | 4 |
| ***learning\_rate***: Controls the contribution of each tree to the final prediction; a smaller learning rate generally improves model performance; ***subsample***: The proportion of subsamples; this randomly selects a portion of the training samples to build trees, thus helping to prevent overfitting; ***max\_depth***: The depth of the tree, affecting model complexity; ***n\_estimators***: The total number of estimators, specifying how many trees to build; more trees generally enhance model fitting capacity; ***colsample\_bytree***: The proportion of columns sampled when creating trees, reducing the risk of overfitting; ***reg\_alpha***: L1 regularization coefficient; ***reg\_lambda***: L2 regularization coefficient; ***min\_child\_weight***: The minimum sum of sample weights in a leaf node, controlling the minimum number of samples in child nodes, which helps prevent overfitting. | | | | | | | | |
|  | learning\_rate | num\_leaves | max\_depth | n\_estimators | reg\_alpha | reg\_lambda | - | |
| **LightGBM** | 0.01 | 30 | 15 | 200 | 0.4 | 0.1 |
| ***learning\_rate***: Controls the contribution of each tree to the final result; a smaller learning rate requires more trees to achieve the same model effect; ***num\_leaves***: Controls the maximum number of leaves in each tree, often used in conjunction with max\_depth to prevent overfitting; ***max\_depth***: The maximum depth of the tree; ***n\_estimators***: The number of trees; ***reg\_alpha***: L1 regularization; ***reg\_lambda***: L2 regularization. | | | | | | | | |
|  | learning\_rate | n\_estimators | algorithm | base\_estimator | max\_depth | - | | |
| **AdaBoost** | 0.1 | 300 | SAMME | DecisionTreeClassifier | 5 |
| ***learning\_rate***: Controls the contribution of each weak learner; a smaller learning rate combined with a greater number of iterations (n\_estimators) can enhance model accuracy; ***n\_estimators***: The number of weak learners; ***algorithm***: Specifies the type of Boosting algorithm; SAMME is an unweighted method; ***base\_estimator***: Specifies the weak learner, typically a decision tree; ***max\_depth***: Limits the depth of the tree, controlling model complexity and helping to avoid overfitting. | | | | | | | | |
|  | hidden\_layer\_sizes | activation | solver | alpha | learning\_rate | max\_iter | - | |
| **MLP** | (50, 50) | tanh | sgd | 0.0001 | adaptive | 500 |
| ***hidden\_layer\_sizes***: The structure and size of the hidden layers, which can be set to one or multiple layers, each containing a certain number of neurons, influencing the model's expressiveness; ***activation***: The choice of activation function, affecting the output of each neuron; selectable activation functions include “tanh” and “relu”, influencing the model's non-linear capability and convergence speed; ***solver***: The choice of algorithm for weight optimization; available optimization algorithms include “adam” and “sgd” (stochastic gradient descent); ***alpha***: L2 regularization coefficient; ***learning\_rate***: Learning rate strategy, can choose “constant” (fixed learning rate) or “adaptive” (adaptive learning rate); ***max\_iter***: The maximum number of iterations, limiting the number of training cycles to prevent prolonged training from affecting performance. | | | | | | | | |
|  | learning\_rate | depth | iterations | L2\_leaf\_reg | subsample | random\_strength | auto\_class\_weights | - |
| **Catboost** | 0.01 | 8 | 200 | 0.8 | 0.7 | 0.6 | Balanced |
| ***learning\_rate***: Controls the weight update level for each tree; ***depth***: The maximum depth of the tree; ***iterations***: The number of trees (iterations); ***L2\_leaf\_reg***: L2 regularization term used to control leaf node weights to prevent overfitting; ***subsample***: Subsample rate, indicating the proportion of data randomly selected for training each tree; an appropriate subsample rate can improve generalization capability; ***random\_strength***: Introduces additional random noise when selecting split features, enhancing model robustness and randomness; by increasing randomness, the model becomes more resilient to outliers and noise in the data, aiding in improving generalization capability; ***auto\_class\_weights***: Adjusts the class weights of the classifier to balance the influence of each class. | | | | | | | | |