machine learning models: summary

Flexible models are judged by how well they predict data they have not seen, because a model that is too complex fits the noise in its training data. A **random forest** builds many decision trees, each from a random subset of the data and trying a random subset of the predictors at each split, then averages their predictions (regression) or takes a majority vote (classification). The kind of response decides the `model_type`. `buildModel()` tries every combination of settings in the `optimization_parameters` grid and reports each one's cross-validated metric: RMSE for regression, where lower is better, and accuracy for classification, where higher is better. `predictWithModel()` then applies the best model to new samples.

Function or setting	What it is for	What we set or vary
`model_type = "random_forest_regression"`	a forest that predicts a number, such as Valine abundance	used with a numeric `response`
`model_type = "random_forest_classification"`	a forest that predicts a category, such as `patient_status`	used with a categorical `response`
`buildModel()`	building and tuning the forest	`data`, `model_type`, `predictors`, `response`, `optimization_parameters`
`optimization_parameters`	the grid of settings to try, as a `list()` of `seq()` ranges	the three settings below
`n_vars_tried_at_split`	how many predictors are candidates at each split	smaller values make the forest more random
`n_trees`	how many trees the forest grows	more trees give steadier predictions, with diminishing returns
`min_leaf_size`	the fewest samples a node needs before it is split again	larger values make simpler, less overfitted trees
`$metrics`	one row per combination, with `mean` holding its cross-validated `.metric`	for a classifier, sort with `arrange(desc(mean))` (higher accuracy is better), or map it across the grid with `geom_tile()`
`predictWithModel()`	predictions for samples the model was not built on, returned as the same table with `predicted_<response>` (regression) or `probability_` and `predicted_class` columns (classification) added	`data`, and `model` (the whole `buildModel()` output)