“Estimator Configuration” Module Transcript

Chapter 1

Intro, Topics Covered, & Learning Outcomes

Hi. I'm Hayley from the One AI team here at One Model. In previous modules, we've explored common algorithms used in classification and regression tasks. Understanding these algorithms is helpful because they are the building blocks that estimators use to learn patterns from data. In this module, we will continue the advanced configuration module series by exploring estimators and their configuration in One AI. Estimators are models that employ these algorithms to make predictions or decisions based on input data.

We will cover an overview of estimators, the estimators available in One AI, the default estimator configuration settings in One AI, and how to configure estimators for One AI models.

After watching, you will understand the role and types of estimators in machine learning tasks, enabling you to select the most suitable estimators for each model; gain the ability to configure estimators in One AI, including how to enable and adjust default and optional estimators for specific models; and you will adjust estimator parameters and hyperparameters to fine-tune and optimize model performance, understanding the impact of these adjustments on the accuracy and reliability of predictions.

Chapter 2

Overview of Estimators

Section 2 - Overview of Estimators

An estimator is a model or tool that learns from data to make predictions or transform data. Think of it as a mathematical formula or set of rules that can take input data, such as your data in One Model that makes up the model dataset and produce an output, such as predictions or results from your model.

While sometimes - debatably incorrectly - used interchangeably with the term "algorithm", "estimator" typically refers to an instance of a machine learning algorithm that has been trained on a dataset. In simpler terms, an estimator is like a predictive model that learns from data to make predictions, while an algorithm is the set of rules or

steps that the estimator follows to learn from the data. It's the method used to train the estimator.

The term estimator in machine learning comes from the field of statistics where an estimator is a rule or method for estimating an unknown parameter or quantity based on observed data. In machine learning, the concept is extended to include models and algorithms that "estimate" the relationship between input features and the target variable.

Estimators can be used for various tasks such as classification, regression, clustering, and data preprocessing.

Chapter 3

Estimators Available in One AI

Section 3 - Estimators Available in One AI.

One Model offers a variety of machine learning estimators for making predictions in One AI. I will provide brief descriptions of each estimator and explain when they are most appropriate to use.

Chapter 4

Estimators Available in One AI: Classification Estimators

Let's start with classification estimators.

The AdaBoostClassifier, which is short for adaptive boosting, is an ensemble method that combines multiple weak classifiers, such as decision trees, to form a strong classifier. It adjusts the weights of incorrectly classified instances so that subsequent classifiers focus more on difficult cases. This estimator is useful when you need to boost the performance of simpler models, especially on moderate-sized datasets. It's effective in reducing bias and variance in models.

Next we have the DecisionTreeClassifier, which works by splitting the data into branches at each node based on feature values resulting in a tree like structure of decisions. Each internal node represents a test on an attribute. For example, whether a coin flip comes up heads or tails. Each branch represents the outcome of the test, and each leaf node represents a class label, which is the decision taken after computing all attributes. It's useful when you need a model that is easy to understand and interpret. It

can handle both numerical and categorical data, but may overfit on small datasets without proper tuning.

The KNeighborsClassifier classifies new cases based on a similarity measure based on the classes of their nearest neighbors in the feature space. It's most suitable for small to medium sized datasets where simplicity and intuitive results are desired.

Next, we have the LightGBMClassifier, which is short for Light Gradient Boosting Machine. This builds an ensemble of decision trees sequentially where each new tree corrects the errors of the previous trees. It uses a technique called gradient boosting, which optimizes the model by minimizing a loss function. It's ideal for large datasets and situations where model performance and training speed are important. It works well with complex data and can handle a large number of features. Personally, this is my favorite classifier to use for One AI models most of the time.

Next, we have the LogisticRegression, which is in fact a classifier despite its name. It's a linear model that estimates the probability of a binary outcome using a logistic function. It's most appropriate for simple binary classification problems where interpretability and computational efficiency are important. It performs well with linearly separable data and provides probabilistic outputs.

The RandomForestClassifier is an ensemble learning method that constructs multiple decision trees during training and merges their results. Each tree is built from a random subset of the training data and a random subset of features at each split, ensuring diversity among the trees. The final prediction is made by majority voting among all the trees' predictions. It's effective for a variety of tasks and works well with large datasets, providing robust performance even with missing data and noisy features. It's particularly useful when you need high accuracy and can tolerate less interpretability.

And finally, we have SVC, which is short for Support Vector Classifier, and this works by finding the hyperplane, which is the decision boundary that separates the data into classes. It's advantageous for complex datasets with clear margin separations and when high accuracy is required. It can handle both linear and nonlinear relationships through the use of kernel functions.

Chapter 5

Estimators Available in One AI: Regression Estimators

Now we will move into regression estimators.

The AdaBoost, DecisionTree, LightGBM, RandomForestRegressor, and SVR all function essentially the same as their classifier counterparts; so, I won't get too into those. Basically, the key difference is that the regressor is used for regression tasks and outputs continuous values, whereas the classifier is used for classification tasks and outputs categorical outcomes.

Onto the new regressors.

Lasso is a type of linear regression that includes L1 regularization, which adds a penalty equal to the absolute value of the magnitude of the coefficients. It shrinks some coefficients to exactly zero, effectively selecting a simpler model that only includes the most important predictors. It also helps prevent overfitting by constraining the coefficients leading to a more generalized model. It's suitable when you have a large number of features and want to perform feature selection, especially when some of the features are irrelevant or redundant.

Ridge is another type of linear regression that includes L2 regularization, which adds a penalty equal to the square of the magnitude of the coefficients. It shrinks the coefficients, but unlike Lasso, it does not set any coefficients exactly to zero. This means it keeps all the predictors in the model. It's also particularly effective in handling multicollinearity as it distributes the coefficients among correlated variables.

ElasticNet is a regularized regression method that combines the properties of both Lasso and Ridge regression techniques to enhance the prediction accuracy and interpretability of the statistical model. It works by adding both L1 and L2 penalties to the loss function used in linear regression. It's useful when there are multiple correlated features, and it's beneficial for datasets where some predictors are strongly correlated and where both variable selection and regularization are needed to enhance model performance and generalizability.

Next, we have the GaussianProcessRegressor. These model the data as a distribution of possible functions and uses observed data to update this distribution. The model then makes predictions based on this updated distribution, giving both an expected value and a confidence interval for each prediction. It uses kernel functions to determine how data points influence each other. Use this when you need to predict not only the expected values but also understand the uncertainty of these predictions.

The HuberRegressor uses a special loss function that behaves like least squares regression for small errors and like absolute value regression for large large errors. This means it gives less weight to outliers, reducing their impact on the model. This is ideal when your data has outliers and you want a model that balances being accurate for most data points while not being skewed by extreme values.

Next, we have LinearRegression, which is a simple and commonly used type of predictive analysis that models the relationship between a dependent variable and one or more independent variables by fitting a linear equation to observe data. Use this when you believe there is a linear relationship between the variables and when interpretability is important. It works best for small- to medium-sized datasets with less noise.

And finally, we have the SGD regressor, which is short for stochastic gradient descent regressor. This uses the entire dataset to calculate the gradients and updates the model parameters using one training example at a time. This makes the algorithm faster and more scalable. It minimizes a loss function by iteratively adjusting the parameters in the direction that reduces the error. This is most suitable for large datasets and when you need an efficient and scalable method for linear regression.

Chapter 6

Default Settings in One AI

Section 4 - Default Settings in One AI

If you choose not to perform estimator configuration, One AI will automatically try an intelligent subset of estimators, parameters, and hyperparameters based on the type of model you are working with. The combination that results in the best model performance and fit will be selected and used unless manual configuration is performed.

If you are working with a classification model, by default, One AI tries the following classification estimators: AdaBoost, LightGBM, LogisticRegression, and RandomForest.

Optional classifiers include: DecisionTreeClassifier, KNeighborsClassifier, and SVC.

If you are working with a regression model, by default, One AI will try the following regression estimators, ElasticNet, Lasso, LightGBM, LinearRegression, and RandomForest.

Optional regressors include AdaBoostRegression, DecisionTree, GaussianProcess, Huber, Ridge, SGD, and SVR.

For the sake of time, I won't go through each estimator's default parameter and hyperparameter configuration in this module, but I have linked the "One AI Machine Learning Algorithms and Settings" help article in the module description for easy access.

This is that help article here. As you can see, each parameter and hyperparameter that can be edited in One AI is listed with the default value. You can also click the link link next to the classifier for more information about the estimator and each parameter and hyperparameter.

In the next section, we will head over to the One Model site to talk about configuration.

Chapter 7

Estimator Configuration in One AI

Section 5 - Estimator Configuration in One AI

To perform estimator configuration in One AI, click the 'Edit' button for the model of interest and scroll down to "Estimator Configuration" under "One AI Configuration".

Click the caret to expand and reveal your options. You will only see estimators that match the type of machine learning problem you are working with for the selected model. For example, this model is a classification, so it will only reveal classifiers.

You have the option to configure which estimators One AI will try, the estimators' parameters and hyperparameters, or both.

Slide the toggle to 'On' for each estimator you want One AI to try. If you have made any manual configurations, you must slide the toggle to 'On' for each estimator you want One AI to try, including the defaults. This allows users to turn off default estimators. For example, if you only want One AI to try the RandomForest and AdaBoostClassifier, slide these two toggles to 'On' like so. This will turn off the other default estimators.

To configure parameters and hyperparameters, click the caret next to the estimator to expand the options. Turn 'On' the override toggle for each parameter and hyperparameter that you wish to configure.

Within each estimator type, there will be a combination of drop down menus and designated fields for free text. Hover over the black information icon next to each parameter to see what types of values are accepted.

In many cases, multiple values can be accepted, even though only a single parameter is used for the instance of the estimator, allowing you to configure several parameters at once within a single One AI run to see which combination performs the best. For example, if you use the following configuration with two values for learning rate, two models will be evaluated. One with an AdaBoostClassifier estimator that has a learning

rate of 0.5 and another with a learning rate of 1.0. The scores would be compared with One AI selecting the best performing model.

Remember, you can refer back to the "One AI Machine Learning Algorithms and Settings" help article to see the default parameters and hyperparameters for each estimator to make making these configurations a little easier.

Once you are happy with your estimator configuration, scroll to the bottom and click 'Save', and then rerun your model.

Remember, you can always view the final selected estimator configuration in the Estimation Details section of the Result Summary for any completed model run.

Chapter 8

Conclusion & Thanks

We've explored the importance of estimators in machine learning and how they are configured in One AI. By understanding the various types of estimators and their specific uses, you can make informed decisions to enhance your model's performance. Remember, proper configuration is key to building accurate and reliable models. Happy modeling!