Hyper Parameter Optimization

In machine learning, models are trained to predict unknown labels for new data based on correlations between known labels and features found in the training data. Depending on the algorithm used, you may need to specify **hyperparameters** to configure how the model is trained.

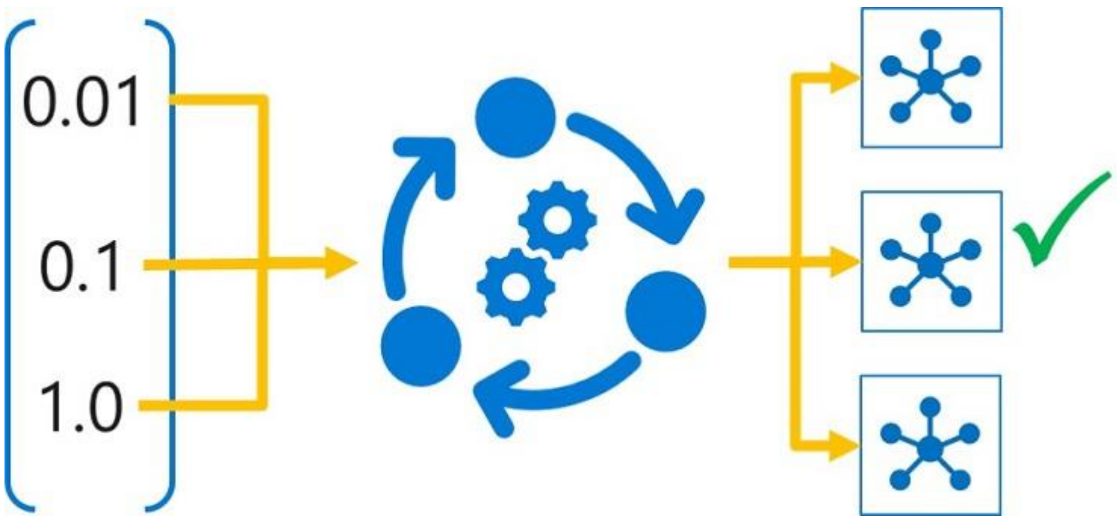
For example, the *logistic regression* algorithm uses a *regularization rate* hyperparameter to counteract overfitting; and deep learning techniques for convolutional neural networks (CNNs) use hyperparameters like *learning rate* to control how weights are adjusted during training, and *batch size* to determine how many data items are included in each training batch.

Note

Machine Learning is an academic field with its own particular terminology. Data scientists refer to the values determined from the training features as *parameters*, so a different term is required for values that are used to configure training behavior but which are ***not*** derived from the training data - hence the term *hyperparameter*.

The choice of hyperparameter values can significantly affect the resulting model, making it important to select the best possible values for your particular data and predictive performance goals.

Tuning hyperparameters



Hyperparameter tuning is accomplished by training the multiple models, using the same algorithm and training data but different hyperparameter values. The resulting model from each training run is then evaluated to determine the performance metric for which you want to optimize (for example, *accuracy*), and the best-performing model is selected.

SEARCH SPACE

The set of hyperparameter values tried during hyperparameter tuning is known as the **search space**. The definition of the range of possible values that can be chosen depends on the type of hyperparameter.

Discrete hyperparameters

Some hyperparameters require *discrete* values - in other words, you must select the value from a particular *finite* set of possibilities. You can define a search space for a discrete parameter using a **Choice** from a list of explicit values, which you can define as a Python **list** (Choice(values=[10,20,30])), a **range** (Choice(values=range(1,10))), or an arbitrary set of comma-separated values (Choice(values=(30,50,100)))

You can also select discrete values from any of the following discrete distributions:

- QUniform(min_value, max_value, q): Returns a value like round(Uniform(min_value, max_value) / q) * q
- QLogUniform(min_value, max_value, q): Returns a value like round(exp(Uniform(min_value, max_value)) / q) * q
- QNormal(mu, sigma, q): Returns a value like round(Normal(mu, sigma) / q) * q

- QLogNormal(mu, sigma, q): Returns a value like round(exp(Normal(mu, sigma)) / q) * q

Continuous hyperparameters

Some hyperparameters are *continuous* - in other words you can use any value along a scale, resulting in an *infinite* number of possibilities. To define a search space for these kinds of value, you can use any of the following distribution types:

- Uniform(min_value, max_value): Returns a value uniformly distributed between min_value and max_value
- LogUniform(min_value, max_value): Returns a value drawn according to exp(Uniform(min_value, max_value)) so that the logarithm of the return value is uniformly distributed
- Normal(mu, sigma): Returns a real value that's normally distributed with mean mu and standard deviation sigma
- LogNormal(mu, sigma): Returns a value drawn according to exp(Normal(mu, sigma)) so that the logarithm of the return value is normally distributed

SAMPLING TECHNIQUES

The specific values used in a hyperparameter tuning run, or **sweep job**, depend on the type of **sampling** used.

There are three main sampling methods available in Azure Machine Learning:

- **Grid sampling**: Tries every possible combination.
- **Random sampling**: Randomly chooses values from the search space.

- **Bayesian sampling**: Chooses new values based on previous results.

Grid sampling

Grid sampling can only be applied when all hyperparameters are discrete, and is used to try every possible combination of parameters in the search space.

For example, grid sampling is used to try every possible combination of discrete *batch_size* and *learning_rate* value:

Random sampling

Random sampling is used to randomly select a value for each hyperparameter, which can be a mix of discrete and continuous values.

Bayesian sampling

Bayesian sampling chooses hyperparameter values based on the Bayesian optimization algorithm, which tries to select parameter combinations that will result in improved performance from the previous selection.

You can only use Bayesian sampling with **choice**, **uniform**, and **quniform** parameter expressions.

Model Hyperparameters

- The model hyperparameters are external to the machine learning model.
- The values of the model hyperparameters must be set using hyperparameter tuning or determined manually.
- The values of the model hyperparameters must be set before starting the training of the machine learning model.
- The model hyperparameters are in-turn used to determine values of the model parameters.

Examples

- Learning rate.
- Number of clusters.
- Number of hidden layers in a neural network.
- Choice of activation function, penalty function, loss function, etc.
- Value of regularization variable.
- Choice of train test split.

HYPERPARAMETER SEARCH SPACE

- The search space includes a set of values associated with a hyperparameter variable that should be tried during the training of the multiple models.

- There are two main types of hyperparameters namely, Discrete and Continuous.

Discrete

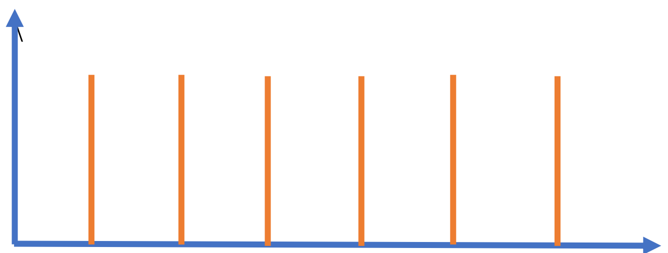
The hyper parameter values are discrete just as,

a. [10, 12, 18, 24, 36, 45]

b. [Yes, No]

c. [True, False]

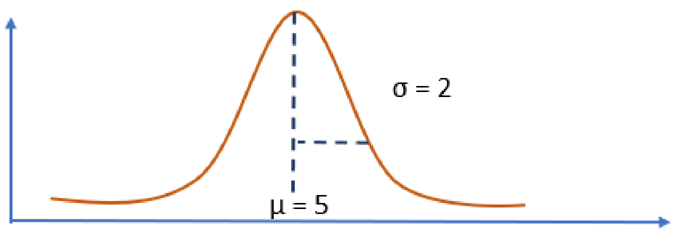
d. qUniform (5, 10)



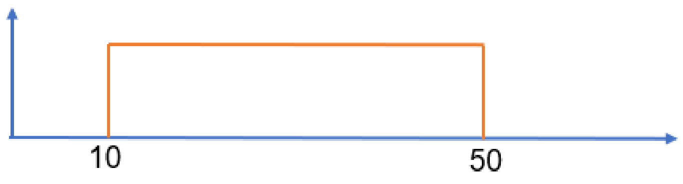
Continuous

Continuous search space has hyper parameter values from continuous data distribution.

a. normal(5, 2)



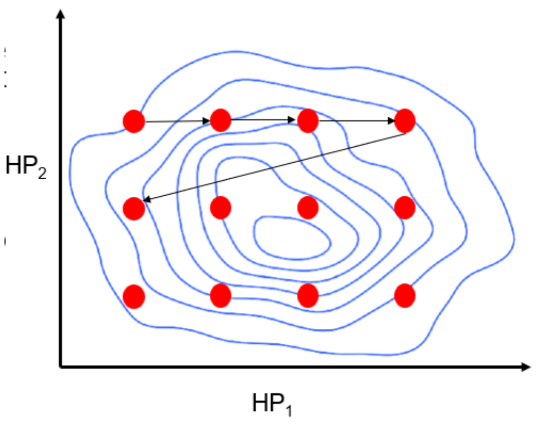
b. uniform(10, 50)



SAMPLING TECHNIQUES

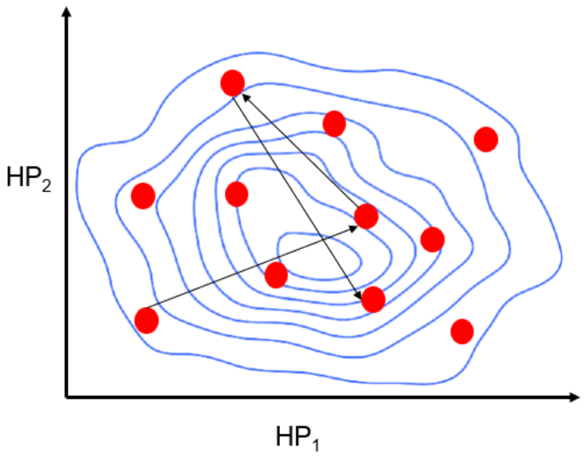
Grid Sampling

- All the possible combinations of the hyperparameters are tried and the best combination is returned.
- Example:-
    - If HP1 = [2, 5], and HP2 = [60, 85, 90].
    - Then the 6 models trained using the two hyperparameters specified comprises
    - [2,60],
    - [2,85],
    - [2,90],
    - [5,60],
    - [5,85],
    - [5,90]



Random Sampling

- The hyperparameter values are randomly selected from the specified values or range and models are trained.
- The best of values is provided based on the performance metric.
- Example:- If HP1 = [2, 5, 8, 10], and HP2 = uniform(150, 156).
- Then the models trained for the two hyperparameters specified may comprise {[8,155.6], [10,151.94], [8,154.82]}



Bayesian Sampling

- The value of the hyperparameters to be tried is decided using the Bayesian optimization algorithm.
- A probabilistic model is built for the function mapping between the hyperparameters and the performance metric.
- Example:- If HP$_1$ = [2, 5, 10], and HP$_2$ = normal(1, 7), then at iteration ‘i’ if the values [2, 1.9] resulted in a performance metric value of ‘X’.

- For the iteration ‘i+1’ the algorithm provided the hyperparameter values (say [2, 1.1]) such that the model performance is better than ‘X’.