# **Multi-class Logistic Regression**

Multi-class Logistic Regression aims to assign labels to instances by using binary logistic regression (Denis, add link to the Logistic Regression Page), where the labels are drawn from a finite set of several elements.

The implemented approach for doing so is to reduce the single multi-class problem into multiple binary classification problems via one-versus-all.

The one-versus-all approach is the process of building binary classifiers which distinguish between one of the labels and the rest.

## **Model**

The model keeps the pairs <ClassLabel, LogisticRegressionModel> and it enables a prediction to be made for a given vector of features, in the following way:

LogRegressionMultiClassModel

mdl = …

**double** prediction = mdl.withRawLabels(true).withThreshold(0.5).apply(**observation**)

Presently Ignite supports several parameters for LogisticRegressionModel :

* isKeepingRawLabels - controls the output label format: 0 and 1 for false value and raw distances from the separating hyperplane (default value: false)
* threshold - a threshold to assign label ‘1’ to the observation if the raw value is more than this threshold (default value: 0.5)

LogisticRegressionSGDTrainer

mdl = …

**double** prediction = mdl.withRawLabels(true).withThreshold(0.5).apply(**observation**)

## **Trainer**

Trainer of the multi-class logistic regression model runs a number of binary logistic regression trainers under the hood.

Presently, Ignite supports the following parameters for LogRegressionMultiClassTrainer

* **updatesStgy** - update strategy
* **maxIterations - max amount of iterations before convergence**
* **batchSize - The size of learning batch**
* **locIterations - the amount of local iterations of SGD algorithm**
* **seed - seed value for internal random purposes to reproduce training results**

// Set up the trainer

LogRegressionMultiClassTrainer<?> trainer = new LogRegressionMultiClassTrainer<>()  
 .withUpdatesStgy(UPDATES\_STRATEGY)  
 .withAmountOfIterations(MAX\_ITERATIONS)  
 .withAmountOfLocIterations(BATCH\_SIZE)  
 .withBatchSize(LOC\_ITERATIONS)  
 .withSeed(SEED);  
  
 // Build the model  
 LogisticRegressionModel mdl = trainer.fit(  
 ignite,  
 dataCache,  
 featureExtractor,  
 labelExtractor

);  
  
All properties will be propagated for each pair one-versus-all LogRegressionMultiClassTrainer.

# **Example**

To see how LogRegressionMultiClassModel can be used in practice, try this [example](https://github.com/apache/ignite/blob/master/examples/src/main/java/org/apache/ignite/examples/ml/regression/logistic/multiclass/LogRegressionMultiClassClassificationExample.java) that is available on GitHub and delivered with every Apache Ignite distribution. The preprocessed Glass dataset is from the [UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/datasets/Glass+Identification).

There are 3 classes with labels: 1 (building\_windows\_float\_processed), 3 (vehicle\_windows\_float\_processed), 7 (headlamps) and feature names: 'Na-Sodium', 'Mg-Magnesium', 'Al-Aluminum', 'Ba-Barium', 'Fe-Iron'.