**Model Building**

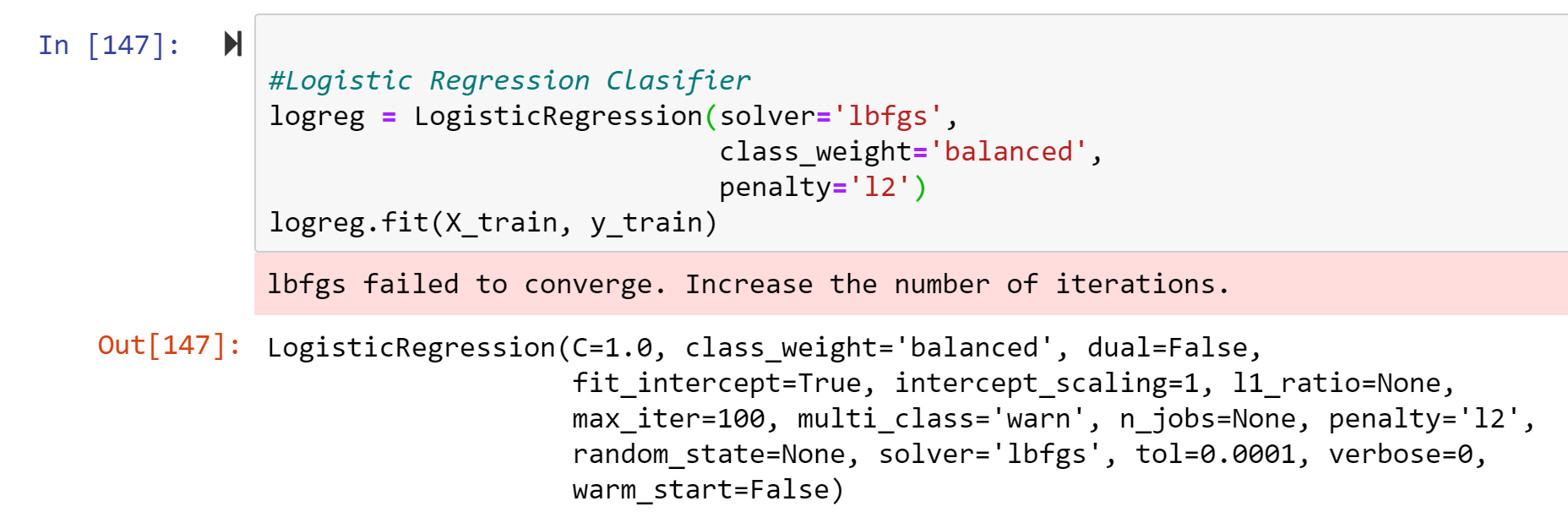
**Logistic Regression:**

**Used parameters:**

**solver**: lbfgs, Algorithm to use in the optimization problem.

**class\_weight**: balanced, mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data.

**penalty**: l2, used to specify the norm used in the penalization.



**‘**

[**https://scikit-learn.org/stable/modules/generated/sklearn.linear\_model.LogisticRegression.html**](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)

**Random Forest:**

**Important parameters:**

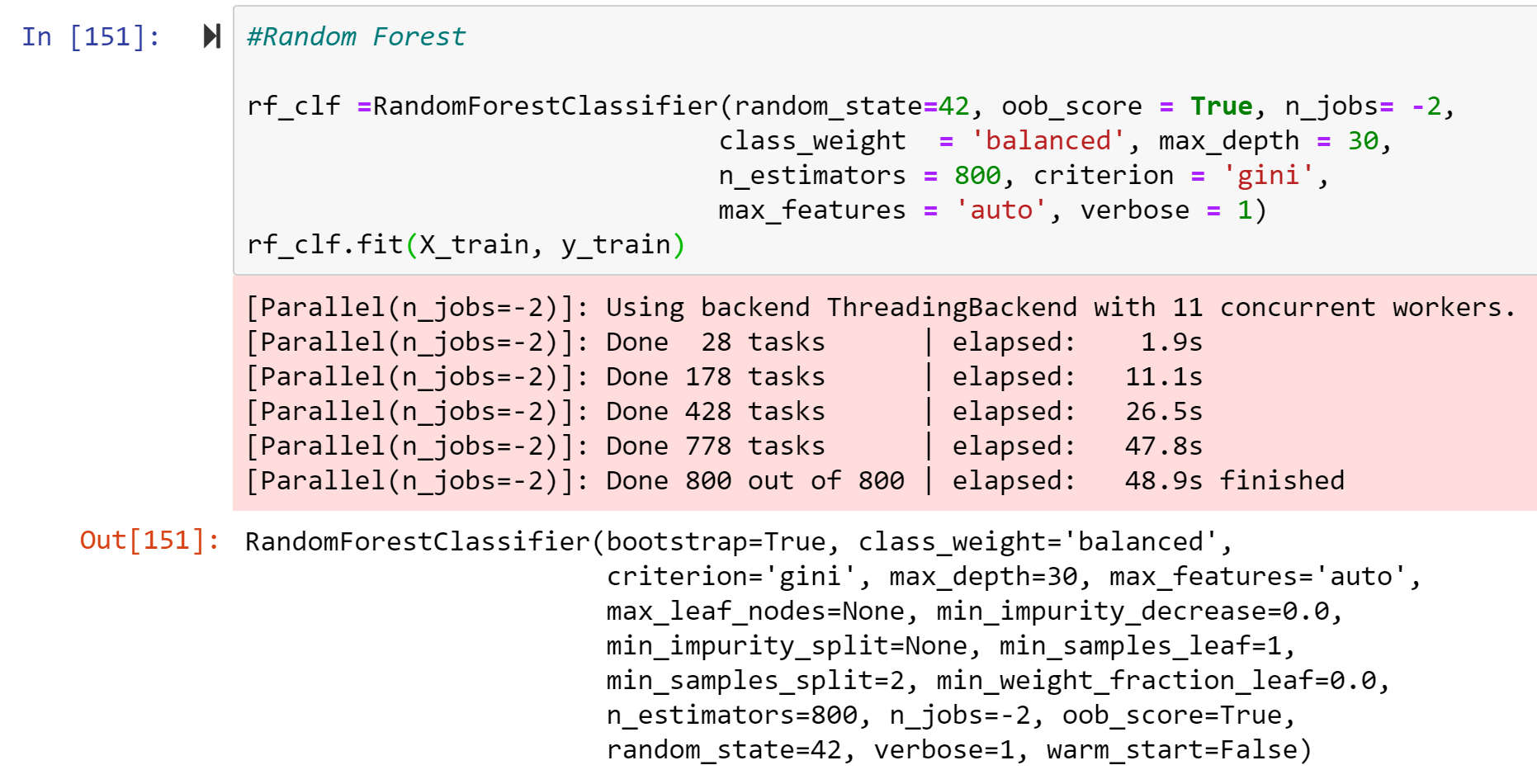
**max\_depth**: The maximum depth of the tree.

**n\_estimators**: The number of trees in the forest.

**class\_weight**: balanced, mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data.

**criterion**: gini, The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain.

**max\_features**: The number of features to consider when looking for the best split.



<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

LightGBM:

**Important parameters:**

**boosting\_type**: dart, Dropouts meet Multiple Additive Regression Trees.

**dart\_subsample**: Subsample ratio of the training instance

**num\_leaves:** Maximum tree leaves for base learners.

**max\_dept:** Maximum tree depth for base learners,

**learning\_rate**: Boosting learning rate.

**n\_estimators**: Number of boosted trees to fit.

**is\_unbalance**: False, balanced, mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data.

**reg\_alpha:** L1 regularization term on weights

**reg\_lambda:** L2 regularization term on weights

<https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html>