scikit-learn LogisticRegression Hyperparameters Reference

Class Definition

```
class sklearn.linear_model.LogisticRegression(
    penalty='l2', *, dual=False, tol=0.0001, C=1.0,
    fit_intercept=True, intercept_scaling=1,
    class_weight=None, random_state=None,
    solver='lbfgs', max_iter=100, verbose=0,
    warm_start=False, n_jobs=None, l1_ratio=None
)
```

1 Key Hyperparameters

1.1 Regularization

- **penalty** {'l1', 'l2', 'elasticnet', None}, default='l2'
  Norm of the penalty. Note: Some penalties may not work with some solvers.
- **C** float, default=1.0
  Inverse of regularization strength (must be > 0). Smaller values = stronger regularization.
- **l1_ratio** float, default=None
  Elastic-Net mixing parameter ($0 \leq$ l1_ratio ≤ 1). Only used when penalty='elasticnet'.

1.2 Solver Configuration

- **solver** {'lbfgs', 'liblinear', 'newton-cg', 'newton-cholesky', 'sag', 'saga'}, default='lbfgs'
  Optimization algorithm. See compatibility table below.
- **max_iter** int, default=100
  Maximum iterations for solvers to converge.
- **tol** float, default=1e-4
  Tolerance for stopping criteria.

1.3 Other Parameters

- **fit_intercept** bool, default=True
  Whether to add intercept/bias to decision function.
- **class_weight** dict or 'balanced', default=None
  Weights associated with classes. 'balanced' adjusts weights inversely proportional to class frequencies.
- **random_state** int, RandomState, default=None
  Seed for shuffling data (used when solver='sag', 'saga' or 'liblinear').
- **n_jobs** int, default=None
  CPU cores used when parallelizing over classes (for multi_class='ovr').

Solver Compatibility Matrix

Solver	Supported Penalties	Multinomial
'lbfgs'	'l2', None	Yes
'liblinear'	'l1', 'l2'	No
'newton-cg'	'l2', None	Yes
'newton-cholesky'	'l2', None	No
'sag'	'l2', None	Yes
'saga'	'elasticnet', 'l1', 'l2', None	Yes

Key Attributes

- `coef_`: Feature coefficients (shape: [n_classes, n_features])
- `intercept_`: Intercept terms (shape: [n_classes,])
- `classes_`: Array of class labels
- `n_iter_`: Actual iterations for each class

Example Usage

```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression

X, y = load_iris(return_X_y=True)
clf = LogisticRegression(
    penalty='l2',
    C=0.1,
    solver='lbfgs',
    max_iter=200
).fit(X, y)

print(clf.predict(X[:2]))  # [0 0]
print(clf.score(X, y))     # ~0.97
```