Falguni Gaikwad

J077

# Sci-kit learn API

## 1. Linear Regression:

- ***Code***: *sklearn.linear_model.**LinearRegression**(*, fit_intercept=True, normalize=False, copy_X =True, n_jobs=None, positive=False)*

fit_intercept*bool, default=True*. Whether to calculate the intercept for this model.

normalize*bool, default=False*. This parameter is ignored when `fit_intercept` is set to False.

copy_X*bool, default=True*. If True, X will be copied; else, it may be overwritten.

n_jobs*int, default=None*. The number of jobs to use for the computation.

positive*bool, default=False*. When set to `True`, forces the coefficients to be positive.

- **"LinearRegression"** fits a linear model with coefficients w=(w1,...,wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. . In its fit method arrays X, y and will store the coefficients of the linear model in its coef_ member.

## 2. Logistic Regression:

- ***Code***: *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, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None)*

penalty*{'l1', 'l2', 'elasticnet', 'none'}, default='l2'*
Used to specify the norm used in the penalization.

dual*bool, default=False*
Dual or primal formulation.

tol*float, default=1e-4*
Tolerance for stopping criteria.

C*float, default=1.0*
Inverse of regularization strength; must be a positive float.

fit_intercept*bool, default=True*
Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.

intercept_scaling*float, default=1*
Useful only when the solver 'liblinear' is used and self.fit_intercept is set to True.

class_weight*dict or 'balanced', default=None*
Weights associated with classes in the form `{class_label: weight}`. If not given, all classes are supposed to have weight one.

random_state*int, RandomState instance, default=None*
Used when `solver` == 'sag', 'saga' or 'liblinear' to shuffle the data.

*solver='lbfgs'*
'lbfgs' handle multinomial loss

max_iter*int, default=100*
Maximum number of iterations taken for the solvers to converge.

multi_class*{'auto', 'ovr', 'multinomial'}, default='auto'*
If the option chosen is 'ovr', then a binary problem is fit for each label.

verbose*int, default=0*
For the liblinear and lbfgs solvers set verbose to any positive number for verbosity.

warm_start*bool, default=False*
When set to True, reuse the solution of the previous call to fit as initialization

n_jobs*int, default=None*
Number of CPU cores used when parallelizing over classes if multi_class='ovr'".

l1_ratio*float, default=None*
The Elastic-Net mixing parameter, with `0 <= l1_ratio <= 1`.

- "**LogisticRegression**", despite its name, is a linear model for classification rather than regression. Logistic regression is also known in the literature as logit regression, maximum-entropy classification (MaxEnt) or the log-linear classifier. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function.

## 3. Ridge:

***Code***: *sklearn.linear_model.**Ridge**(alpha=1.0, *, fit_intercept=True, normalize=False, copy_X=True, max_iter=None, tol=0.001, solver='auto', random_state=None)*
alpha*{float, ndarray of shape (n_targets,)}, default=1.0*
Regularization strength; must be a positive float.

fit_intercept*bool, default=True*
Whether to fit the intercept for this model.

normalize*bool, default=False*

This parameter is ignored when `fit_intercept` is set to False.

copy_X*bool, default=True*
If True, X will be copied; else, it may be overwritten.

max_iter*int, default=None*
Maximum number of iterations for conjugate gradient solver.

tol*float, default=1e-3*
Precision of the solution.

solver*{'auto'}*
'auto' chooses the solver automatically based on the type of data.

- "**Ridge**" regression addresses some of the problems of *'Ordinary Least Squares'* by imposing a penalty on the size of the coefficients. The ridge coefficients minimize a penalized residual sum of squares. The complexity parameter alpha>0 controls the amount of shrinkage: the larger the value of alpha, the greater the amount of shrinkage and thus the coefficients become more robust to collinearity. In its fit method arrays X, y and will store the coefficients of the linear model in its coef_ member.

## 4. Lasso:

- ***Code**: sklearn.linear_model.**Lasso**(alpha=1.0, *, fit_intercept=True, normalize=False, precompute=False, copy_X=True, max_iter=1000, tol=0.0001, warm_start=False, positive=False, random_state=None, selection='cyclic')*

alpha*float, default=1.0*
Constant that multiplies the L1 term. Defaults to 1.0. `alpha = 0` is equivalent to an ordinary least square, solved by the `LinearRegression` object.

fit_intercept*bool, default=True*
Whether to calculate the intercept for this model.

normalize*bool, default=False*
This parameter is ignored when `fit_intercept` is set to False.

precompute*bool or array-like of shape (n_features, n_features), default=False*
Whether to use a precomputed Gram matrix to speed up calculations.

copy_X*bool, default=True*
If `True`, X will be copied; else, it may be overwritten.

max_iter*int, default=1000*
The maximum number of iterations.

ol*float, default=1e-4*
The tolerance for the optimization: if the updates are smaller than `to l`, the optimization code checks the dual gap for optimality and continues until it is smaller than `to l`

warm_start*bool, default=False*

When set to True, reuse the solution of the previous call to fit as initialization, otherwise, just erase the previous solution.

positive*bool, default=False*
When set to `True`, forces the coefficients to be positive.

random_state*int, RandomState instance, default=None*
The seed of the pseudo random number generator that selects a random feature to update.

selection*{'cyclic', 'random'}, default='cyclic'*
If set to 'random', a random coefficient is updated every iteration rather than looping over features sequentially by default.

- The "**Lasso**" is a linear model that estimates sparse coefficients. It is useful in some contexts due to its tendency to prefer solutions with fewer non-zero coefficients, effectively reducing the number of features upon which the given solution is dependent.