

APIs used in SKLearn

1.Linear Regression:

Linear Regression fits a linear model with coefficients $w = (w_1, \dots, w_p)$ to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

From the implementation point of view, this is just plain Ordinary Least Squares (`scipy.linalg.lstsq`) or Non Negative Least Squares (`scipy.optimize.nnls`) wrapped as a predictor object.

`Fit(X, y)`- fit the linear model.

`Predict(X)`-predict using linear model.

`Score(X,y)`-returns the coefficient of determination R^2 of the prediction.

Code:

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

2.Logistic Regression:

It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable.

`Fit(X,y)`-fit the model according to the given training data

`Predict(x)`-predict class labels

`Score(X,y)`-returns mean accuracy on the given test data and label

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)
```

3.Ridge :

This model solves a regression model where the loss function is the linear least squares function. This estimator has built-in support for multi-variate regression

Ridge regression penalizes the model based on the sum of squares of magnitude of the coefficients.

Alpha-Regularization strength; must be a positive float. Regularization improves the conditioning of the problem and reduces the variance of the estimates. Larger values specify stronger regularization.

`Fit(X,y)`-fits the regression model

Predict(x)-predicting using the linear model.

Score(X,y)-returns the coefficient of determination

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)
```

4.Lasso:

LASSO regression penalizes the model based on the sum of magnitude of the coefficients. Technically the Lasso model is optimizing the same objective function as the Elastic Net

Fit(X,y)-fit model with coordinate descent

Predict(X)-predict using linear model

Score(X,y)-returns the coefficient of determination of the prediction

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')
```