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 parameter values, effectively reducing the number of variables upon which the given solution is dependent. For this reason, the Lasso and its variants are fundamental to the field of compressed sensing. Under certain conditions, it can recover the exact set of non-zero weights (see Compressive sensing: tomography reconstruction with L1 prior (Lasso)).

Mathematically, it consists of a linear model trained with ℓ_1 prior as regularizer. The objective function to minimize is:

$$\min_w \frac{1}{2n_{samples}}||Xw - y||_2^2 + \alpha||w||_1$$

The lasso estimate thus solves the minimization of the least-squares penalty with $\alpha||w||_1$ added, where α is a constant and $||w||_1$ is the ℓ_1-norm of the parameter vector.

The implementation in the class **Lasso** uses coordinate descent as the algorithm to fit the coefficients. See Least Angle Regression for another implementation:

```
>>>
>>> from sklearn import linear_model
>>> clf = linear_model.Lasso(alpha = 0.1)
>>> clf.fit([[0, 0], [1, 1]], [0, 1])
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
   normalize=False, positive=False, precompute=False, random_state=None,
   selection='cyclic', tol=0.0001, warm_start=False)
>>> clf.predict([[1, 1]])
array([ 0.8])
```

Also useful for lower-level tasks is the function **lasso_path** that computes the coefficients along the full path of possible values.