**NAME:** Harshitha MU

**ROLL NO:** J076

**Sklearn API**

1. **Linear Regression: -**

**Code:**

sklearn.linear\_model.**LinearRegression**(*\**, *fit\_intercept=True*, *normalize=False*, *copy\_X=True*, *n\_jobs=None*, *positive=False)*

Linear Regression 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.

|  |  |
| --- | --- |
| Fit(X,Y) | Fit the model |
| get\_params() | Get parameters for this estimator. |
| predict() | Get the prediction |
| score() | Return the coefficient of determination R^2 of the prediction. |

1. **Logistic Regression:-**

**Code:**

sklearn.linear\_model.**Logistic Regression**(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)

Logistic regression, 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.

|  |  |
| --- | --- |
| decision\_function(x) | Predict confidence scores for samples. |
| densify() | Convert coefficient matrix to dense array format |
| fit(X, y[, sample\_weight]) | Fit the model according to the given training data. |
| get\_params([deep]) | Get parameters for this estimator. |
| predict(X) | Predict class labels for samples in X. |
| predict\_log\_proba(X) | Predict logarithm of probability estimates. |
| predict\_proba(X) | Probability estimates. |
| score(X, y[, sample\_weight]) | Return the mean accuracy on the given test data and labels. |
| set\_params(\*\*params) | Set the parameters of this estimator. |
| sparsify() | Convert coefficient matrix to sparse format. |

1. **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)

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[, sample\_weight]) | Fit Ridge regression model. |
| get\_params([deep]) | Get parameters for this estimator. |
| predict(X) | Predict using the linear model. |
| score(X, y[, sample\_weight]) | Return the coefficient of determination R2 of the prediction. |
| set\_params(\*\*params) | Set the parameters of this estimator. |

1. **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')

LASSO regression penalizes the model based on the sum of magnitude of the coefficients.

|  |  |
| --- | --- |
| fit(X, y[, sample\_weight, check\_input]) | Fit model with coordinate descent. |
| get\_params([deep]) | Get parameters for this estimator |
| path(\*args, \*\*kwargs) | Compute elastic net path with coordinate descent |
| predict(X) | Predict using the linear model. |
| score(X, y[, sample\_weight]) | Return the coefficient of determination R2 of the prediction. |
| set\_params(\*\*params) | Set the parameters of this estimator. |