**ML ASSIGNMENT**

**SKLEARN API**

**Abhishek Mukundan Iyer**

**J021**

1. **Linear Regression:**

* **Code:** sklearn.linear\_model.**LinearRegression**(*\**, *fit\_intercept=True*, *normalize=False*, *copy\_X=True*, *n\_jobs=None*, *positive=False*)
* **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 will store the coefficients of the linear model in its coef\_ member.
* 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.
* **Implementation-**

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

1. **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)

* **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.
* 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

1. **Ridge Regression:**

* **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 addresses some of the problems of [Ordinary Least Squares](#gjdgxs) 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.

1. **Lasso Regression:**

* **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'*)
* 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.