1. **LINEAR REGRESSION**

**Description:**

Approximating the relation between two data point sets with an estimated linear function.

In y = mx +c, m and c are estimated with the help of given points such that the sum of square error distances is minimum of all.

**Internal Functionality:**

Iterating against each point set in the data to converge the estimators to a reasonable estimate for the relation. Also gives error tolerance on interval basis.

Where xi is element in predictor data, and yi is element in target data, with and being the respective sample mean.

**Derivation:**

From (1),

From (2),

This is expanded to target variable with multiple predictor variables as:

**Use-cases:**

Situations where we expect effect and cause to follow in an almost linear fashion.

**Limitations:**

1. Even in cases with minimum of least square errors, this minimum value may be large enough for the approximation to be invalid and rejected.
2. Also, if the relation deviates from linearity and we try to still fit or approximate with linear regression, results would be very incorrect.
3. Linear Regression is based on the assumption that attributes have very low collinearity i.e. almost independent to each other.

**Library:**

In R, base package required.

**Sample code:**

* X <- c (…..)
* Y <- c (…..)
* Rel = lm ( y ~ x )

This gives the sample estimate for this set of data (slope and intercept).

* plot(x,y,abline(Rel))

This plots the data points on a scatterplot and the regression line.

* Prediction <- predict ( Rel, data )

This gives the predicted values when data is fitted to the model.

**Alternate algorithms:**

1. Stepwise Regression
2. Polynomial Regression
3. MARS (Multivariate Adaptive Regression Splines)

**Suitable Business Areas:**

Extrapolating upcoming values based on previously trained estimator of the linear regression data.

(Profits and sales, downsizing and expenses, investment and risk factor, stock prices and neighbouring stocks variation).

1. **LOGISTICS REGRESSION**

**Description:**

Statistical method to analyze how the outcome varies with more than one predictor variable which are binary valued and are not independent of each other.

1. **MARS**

**Description:**

1. Multivariate - Able to generate model based on several input variables (high dimensionality).
2. Adaptive - Generates flexible models in passes each time adjusting the model.
3. Regression - Estimation of relationship among independent and dependent variables.
4. Spline - A piecewise defined polynomial function that is smooth (possesses higher order derivatives) where polynomial pieces connect.
5. Knot - The point at which two polynomial pieces connect.
6. **RIDGE REGRESSION**

**Description:**

Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors. It is hoped that the net effect will be to give estimates that are more reliable

**Alternate Algorithms:**

Principle Component Regression (based on PCA).

1. **LOWESS/LOESS**

**Description:**

LOWESS (Locally Weighted Scatterplot Smoothing), sometimes called LOESS (locally weighted smoothing), is a popular tool used in regression analysis that creates a smooth line through a time plot or scatter plot to help you to see relationship between variables and foresee trends.

LOWESS is typically used for:

1. Fitting a line to a scatter plot or time plot where noisy data values, sparse data points or weak interrelationships interfere with your ability to see a line of best fit.
2. Linear regression where least squares fitting doesn’t create a line of good fit or is too labor-intensive to use.
3. Data exploration and analysis in the social sciences, particularly in elections and voting behavior.

**Internal functioning:**

Parametric and Non-Parametric Fitting

LOWESS, and least squares fitting in general, are non-parametric strategies for fitting a smooth curve to data points. “Parametric” means that the researcher or analyst assumes in advance that the data fits some type of distribution (i.e. the normal distribution). Because some type of distribution is assumed in advance, parametric fitting can lead to fitting a smooth curve that misrepresents the data. In those cases, non-parametric smoothers may be a better choice. Non-parametric smoothers like LOESS try to find a curve of best fit without assuming the data must fit some distribution shape. In general, both types of smoothers are used for the same set of data to offset the advantages and disadvantages of each type of smoother.

Benefits of Non-Parametric Smoothing

1. Provides a flexible approach to representing data.
2. Ease of use.
3. Computations are relatively easy.

Disadvantages of Non-Parametric Smoothing

1. Can’t be used to obtain a simple equation for a set of data.
2. Less well understood than parametric smoothers.
3. Requires the analyst to use a little guesswork to obtain a result.

A simple “**lowess/loess**” curve is constructed using the **lowess**() function, which finds a “fitted” value for each data point; these can be plotted as individual symbols, but they are usually connected with lines. The **lowess**() function has a “span” argument (sometimes symbolized by l) that represents the proportion of the total number of points that contribute to each local fitted value. In practice, the **lowess**() function is often embedded in a points() or ’lines()` function.

The newer **loess**() function uses a formula to specify the response (and in its application as a scatter-diagram smoother) a single predictor variable. The **loess**() function creates an object that contains the results, and the **predict**() function retrieves the fitted values. These can then be plotted along with the response variable. However, the points must be plotted in increasing order of the predictor variable values in order for the **lines**() function to draw the line in an appropriate fashion. This is done by using the results of the **order**() function applied to the predictor variable values, and the explicit subscripting (in square brackets [ ]) to arrange the observations in ascending order.

1. **EXPECTATION MAXIMISATION**
2. **MULTIDIMENSIONAL SCALING**