**Scaling Techniques**

**Why Should we Use Feature Scaling?**

Some machine learning algorithms are sensitive to feature scaling while others are virtually invariant to it.

**01) Gradient Descent Based Algorithms**

Machine learning algorithms like linear regression, logistic regression, neural network, etc. that use gradient descent as an optimization technique require data to be scaled

**02) Distance-Based Algorithms**

Distance algorithms like KNN, K-means, and SVM are most affected by the range of features. This is because behind the scenes they are using distances between data points to determine their similarity.

**03) Tree-Based Algorithms**

Tree-based algorithms, on the other hand, are fairly insensitive to the scale of the features. Think about it, a decision tree is only splitting a node based on a single feature. The decision tree splits a node on a feature that increases the homogeneity of the node. This split on a feature is not influenced by other features.

So, there is virtually no effect of the remaining features on the split. This is what makes them invariant to the scale of the features.

01) Use MinMaxScaler as your default

02) Use RobustScaler if you have outliers and can handle a larger range

03) Use StandardScaler if you need normalized features

**What is Normalization?**

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

Formula for normalization: **x\_scaled = (x – x\_min)/(x\_max – x\_min)**

**What is Standardization?**

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

Formula for standardization: **x\_scaled = (x – mean)/std\_dev**

The Big Question – Normalize or Standardize?

# Normalization vs. standardization is an eternal question among machine learning newcomers. Let me elaborate

# on the answer in this section.

# 01) Normalization is good to use when you know that the distribution of your data does not follow a

# Gaussian distribution. This can be useful in algorithms that do not assume any distribution of the

# data like K-Nearest Neighbors and Neural Networks.

# 02) Standardization, on the other hand, can be helpful in cases where the data follows a Gaussian distribution.

# However, this does not have to be necessarily true. Also, unlike normalization, standardization does not have

# a bounding range. So, even if you have outliers in your data, they will not be affected by standardization.