**RESCALING** attribute data to values to scale the range in [0, 1] or [-1, 1] is useful for the optimization algorithms, such as gradient descent, that are used within machine learning algorithms that weight inputs (e.g. regression and neural networks). Rescaling is also used for algorithms that use distance measurements for example K-Nearest-Neighbors (KNN). Rescaling like this is sometimes called "normalization". MinMaxScaler class in python skikit-learn does this.

**STANDARDIZING** attribute data assumes a Gaussian distribution of input features and "standardizes" to a mean of 0 and a standard deviation of 1. This works better with linear regression, logistic regression and linear discriminate analysis. Python StandardScaler class in scikit-learn works for this.

**NORMALIZING** attribute data is used to rescale components of a feature vector to have the complete vector length of 1. This usually means dividing each component of the feature vector by the Euclidiean length of the vector but can also be Manhattan or other distance measurements. This pre-processing rescaling method is useful for sparse attribute features and algorithms using distance to learn such as KNN. Normalizer class in python skikit-learn does this.

**Scaling**

**Scaling** a vector means to add/substract a constant, then multiply/divide by another constant, so the features can lie between given minimum and maximum values. The motivation to use this scaling include robustness to very small standard deviations of features and preserving zero entries in sparse data. Normally, the given range is [0,1]

http://latex.codecogs.com/gif.latex?x%5E*_%7Bi,j%7D=%5Cfrac%7Bx_%7Bi,j%7D-x%5E%7Bmin%7D_%7Bj%7D%7D%7Bx%5E%7Bmax%7D_%7Bj%7D-x%5E%7Bmin%7D_%7Bj%7D%7D

For example, if we have a dataset like below,

http://latex.codecogs.com/gif.latex?%5Cleft%28%5Cbegin%7Barray%7D%7Brrr%7D1&-1&2%5C2&0&0%5C0&1&-1%5Cend%7Barray%7D%5Cright%29

The scaling process will be

1. take the first column vector x0 = [1, 2, 0]T;
2. x0max=2 and x0max=0
3. scale xi,0,

http://latex.codecogs.com/gif.latex?X=%5Cbegin%7Barray%7D%7Blll%7Dx_%7B0,0%7D&=%5Cfrac%7Bx_%7B0,0%7D-0%7D%7B2%7D&=0.5%5Cx_%7B1,0%7D&=%5Cfrac%7Bx_%7B1,0%7D-0%7D%7B2%7D&=1%5Cx_%7B2,0%7D&=%5Cfrac%7Bx_%7B2,0%7D-0%7D%7B2%7D&=0%5C%5Cend%7Barray%7D

Therefore, the scaling x0 is [0.5, 1, 0]T. Repeating the same process for x1 and x2, the scaling dataset is

http://latex.codecogs.com/gif.latex?%5Cleft%28%5Cbegin%7Barray%7D%7Brrr%7D0.5&0&1%5C1&0.5&0.33%5C0&1&0%5Cend%7Barray%7D%5Cright%29

**Standardizing**

**Standardization** of dataset enables the individual feature look like standard normally distributed data: Gaussian with zero mean and unit variance.

http://latex.codecogs.com/gif.latex?x%5E*_%7Bi,j%7D=%5Cfrac%7Bx_%7Bi,j%7D-%5Cbar%7Bx_j%7D%7D%7B%5Csigma%7D

where:

* x̄j is the mean of the vector, and
* σ is the standard deviation of the vector.

Let’t take X as the example again. The standardizing process will be

1. take the first column vector x0 = [1, 2, 0]T;
2. calculate the mean of x̄0 = (1+2+0)/3 = 1
3. calculate the standard deviation of x0,

http://latex.codecogs.com/gif.latex?%5Csigma=%5Csqrt%7B%5Cfrac%7B%281-1%29%5E2+%282-1%29%5E2+%280-1%29%5E2%7D%7B3%7D%7D=%5Csqrt%7B2/3%7D

1. calculate

Therefore, the standardizing x0 is [0, 1.22, -1.22]T. Repeating the same process for x1 and x2, the standardizing dataset is

http://latex.codecogs.com/gif.latex?%5Cleft%28%5Cbegin%7Barray%7D%7Brrr%7D0&-1.22&1.33%5C1.22&0&0.26%5C-1.22&1.221&-1.06%5Cend%7Barray%7D%5Cright%29

**Normalizing**

**Normalizing** a vector is the process of scaling vectors to have unit norm. The motivation is to qualify the similarity of any pair of vectors while using dot-product.

http://latex.codecogs.com/gif.latex?x%5E*_%7Bi,j%7D=%5Cfrac%7Bx_%7Bi,j%7D%7D%7B%7Cx_j%7C%7D

Let’t take X as the example again. The scaling process will be

1. take the first **row** vector x0 = [1, -1, 2];
2. calculate norm of x0

http://latex.codecogs.com/gif.latex?%7Cx_0%7C=%5Csqrt%7B1%5E2+%28-1%29%5E2+2%5E2%7D=%5Csqrt%7B6%7D

1. normalize x0,

Therefore, the normalizing x0 is [0.41, -0.41, 0.82]. Repeating the same process for x1 and x2, the normalizing dataset is