How to Scale Features[1]

There are four common methods to perform Feature Scaling:

1. **Standardization:**

Standardization replaces the values by their Z scores.

$$x' = \frac{x - \bar{x}}{\sigma}$$

This redistributes the features with their mean **μ = 0** and standard deviation **σ =1** . `sklearn.preprocessing.scale` helps us implement standardization in python.

2. Mean Normalization:

$$x' = \frac{x - \text{mean}(x)}{\max(x) - \min(x)}$$

This distribution will have values between **-1 and 1**with **μ=0**.
Standardization and **Mean Normalization** can be used for algorithms that assume zero centric data like **Principal Component Analysis(PCA).**

3. Min-Max Scaling:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

This scaling brings the value between 0 and 1.

4. Unit Vector:

$$x' = \frac{x}{||x||}$$

Scaling is done considering the whole feature vector to be of unit length.

---

[1] This information is taken from, Sudharsan Asaithambi, "Why, How and When to Scale your Features" *Medium*, Dec 3, 2017, https://medium.com/greyatom/why-how-and-when-to-scale-your-features-4b30ab09db5e.

Min-Max Scaling and **Unit Vector** techniques produce values of range [0,1]. When dealing with features with hard boundaries this is quite useful. For example, when dealing with image data, the colors can range from only 0 to 255.

When to Scale

Rule of thumb I follow here is any algorithm that computes distance or assumes normality, **scale your features!!!**

Some examples of algorithms where feature scaling matters are:

- **k-nearest neighbors** with an Euclidean distance measure is sensitive to magnitudes and hence should be scaled for all features to weigh in equally.
- Scaling is critical, while performing **Principal Component Analysis(PCA)**. PCA tries to get the features with maximum variance and the variance is high for high magnitude features. This skews the PCA towards high magnitude features.
- We can speed up **gradient descent** by scaling. This is because θ will descend quickly on small ranges and slowly on large ranges, and so will oscillate inefficiently down to the optimum when the variables are very uneven.
- **Tree based models** are not distance based models and can handle varying ranges of features. Hence, Scaling is not required while modeling trees.
- Algorithms like **Linear Discriminant Analysis(LDA), Naive Bayes** are by design equipped to handle this and give weights to the features accordingly. Performing features scaling in these algorithms may not have much effect.

Sudharsan Asaithambi

Why, How and When to Scale your Features