Feature Engineering: Scaling, Normalization, and Standardization (Updated 2023)

What is Feature Scaling?

Feature scaling is a data preprocessing technique that involves transforming the values of features or variables in a dataset to a similar scale. This is done to ensure that all features contribute equally to the model and to prevent features with larger values from dominating the model. Feature scaling is essential when working with datasets where the features have different ranges, units of measurement, or orders of magnitude. Common feature scaling techniques include standardization, normalization, and min-max scaling. By applying feature scaling, the data can be transformed to a more consistent scale, making it easier to build accurate and effective machine learning models.

Why Should We Use Feature Scaling?

Some machine learning algorithms are sensitive to feature scaling, while others are virtually invariant. Let's explore these in more depth:

1. Gradient Descent Based Algorithms

Machine learning algorithms like linear regression, logistic regression, neural network, PCA (principal component analysis), etc., that use gradient descent as an optimization technique require data to be scaled. Take a look at the formula for gradient descent below:

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

The presence of feature value X in the formula will affect the step size of the gradient descent. The difference in the ranges of features will cause different step sizes for each feature. To ensure that the gradient descent moves smoothly towards the minima and that the steps for gradient descent are updated at the same rate for all the features, we scale the data before feeding it to the model.

> Having features on a similar scale can help the gradient descent converge more quickly towards the minima.

2. Distance-Based Algorithms

Distance algorithms like KNN, K-means clustering, and SVM(support vector machines) 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.**

For example, let's say we have data containing high school CGPA scores of students (ranging from 0 to 5) and their future incomes (in thousands Rupees):

	Student	CGPA	Salary '000
0	1	3.0	60
1	2	3.0	40
2	3	4.0	40
3	4	4.5	50
4	5	4.2	52

Since both the features have different scales, there is a chance that higher weightage is given to features with higher magnitudes. This will impact the performance of the machine learning algorithm; obviously, we do not want our algorithm to be biased towards one feature.

Therefore, we scale our data before employing a distance based algorithm so that all the features contribute equally to the result.

	Student	CGPA	Salary '000
0	1	-1.184341	1.520013
1	2	-1.184341	-1.100699
2	3	0.416120	-1.100699
3	4	1.216350	0.209657
4	5	0.736212	0.471728

The effect of scaling is conspicuous when we compare the Euclidean distance between data points for students A and B, and between B and C, before and after scaling, as shown below:

- Distance AB before scaling =>$\sqrt{(40-60)^2+(3-3)^2}=20$
- Distance BC before scaling =>$\sqrt{(40-40)^2+(4-3)^2}=1$
- Distance AB after scaling =>$\sqrt{(1.1+1.5)^2+(1.18-1.18)^2}=2.6$
- Distance BC after scaling =>$\sqrt{(1.1-1.1)^2+(0.41+1.18)^2}=1.59$

3. 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 only splits a node based on a single feature. The decision tree splits a node on a feature that increases the homogeneity of the node. Other features do not influence this split on a feature.

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

What Is Normalization?

Normalization is a data preprocessing technique used to adjust the values of features in a dataset to a common scale. This is done to facilitate data analysis and modeling, and to reduce the impact of different scales on the accuracy of machine learning models.

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.

Here's the formula for normalization:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Here, Xmax and Xmin are the maximum and the minimum values of the feature, respectively.

- When the value of X is the minimum value in the column, the numerator will be 0, and hence X' is 0

- On the other hand, when the value of X is the maximum value in the column, the numerator is equal to the denominator, and thus the value of X' is 1
- If the value of X is between the minimum and the maximum value, then the value of X' is between 0 and 1

What Is Standardization?

Standardization is another scaling method 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.

Here's the formula for standardization:

$$X' = \frac{X - \mu}{\sigma}$$

$\mu$is the mean of the feature values and$\sigma$is the standard deviation of the feature values. Note that, in this case, the values are not restricted to a particular range.

Now, the big question in your mind must be when should we use normalization and when should we use standardization? Let's find out!

The Big Question – Normalize or Standardize?

Normalization	Standardization
Rescales values to a range between 0 and 1	Centers data around the mean and scales to a standard deviation of 1
Useful when the distribution of the data is unknown or not Gaussian	Useful when the distribution of the data is Gaussian or unknown
Sensitive to outliers	Less sensitive to outliers
Retains the shape of the original distribution	Changes the shape of the original distribution
May not preserve the relationships between the data points	Preserves the relationships between the data points

Normalization	Standardization
Equation: (x – min)/(max – min)	Equation: (x – mean)/standard deviation

However, at the end of the day, the choice of using normalization or standardization will depend on your problem and the machine learning algorithm you are using. There is no hard and fast rule to tell you when to normalize or standardize your data. **You can always start by fitting your model to raw, normalized, and standardized data and comparing the performance for the best results.**

It is a good practice to fit the scaler on the training data and then use it to transform the testing data. This would avoid any data leakage during the model testing process. Also, the scaling of target values is generally not required.

Implementing Feature Scaling in Python

Now comes the fun part – putting what we have learned into practice. I will be applying feature scaling to a few machine-learning algorithms on the Big Mart dataset. I've taken on the DataHack platform.

I will skip the preprocessing steps since they are out of the scope of this tutorial. But you can find them neatly explained in this article. Those steps will enable you to reach the top 20 percentile on the hackathon leaderboard, so that's worth checking out!