

 [LOGIN / REGISTER \(HTTPS://ID.ANALYTICSVIDHYA.COM/AUTH/LOGIN/?](https://id.analyticsvidhya.com/auth/login/)

[NEXT=HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2020/04/FEATURE-SCALING-MACHINE-LEARNING-NORMALIZATION-  
STANDARDIZATION/](https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/)




**Machine Learning Interview Guide**  
— 50 Most Common Questions —  
[Download For Free](#)

[https://www.analyticsvidhya.com/back-channel/download-starter-kit.php?utm\\_source=ml-interview-guide&id=10](https://www.analyticsvidhya.com/back-channel/download-starter-kit.php?utm_source=ml-interview-guide&id=10)

[BEGINNER \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/BEGINNER/\)](https://www.analyticsvidhya.com/blog/category/beginner/)

[DATA ENGINEERING \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/DATA-ENGINEERING/\)](https://www.analyticsvidhya.com/blog/category/data-engineering/)

[PYTHON \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/PYTHON-2/\)](https://www.analyticsvidhya.com/blog/category/python-2/)

[REGRESSION \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/REGRESSION/\)](https://www.analyticsvidhya.com/blog/category/regression/)

[STRUCTURED DATA \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/STRUCTURED-DATA/\)](https://www.analyticsvidhya.com/blog/category/structured-data/)

[TECHNIQUE \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/TECHNIQUE/\)](https://www.analyticsvidhya.com/blog/category/technique/)

## Feature Scaling for Machine Learning: Understanding the Difference Between Normalization vs. Standardization

[ANIRUDDHA BHANDARI \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/AUTHOR/ANIRUDDHA/\)](https://www.analyticsvidhya.com/blog/author/aniruddha/), APRIL 3, 2020 [LOGIN TO BOOKMARK THI...](#)

Article



Video Book



Interview Quiz

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our [Privacy Policy](#)

[Introduction to Feature Scaling](#) (<https://www.analyticsvidhya.com/privacy-policy/>) and [Terms of Use \(https://www.analyticsvidhya.com/terms/\)](https://www.analyticsvidhya.com/terms/).

Accept

I was recently working with a dataset that had multiple features spanning varying degrees of magnitude, range, and units. This is a significant obstacle as a few machine learning algorithms are highly sensitive to these features.

I'm sure most of you must have faced this issue in your projects or your learning journey. For example, one feature is entirely in kilograms while the other is in grams, another one is liters, and so on. How can we use these features when they vary so vastly in terms of what they're presenting?

*This is where I turned to the concept of feature scaling. It's a crucial part of the data preprocessing stage but I've seen a lot of beginners overlook it (to the detriment of their machine learning model).*



(<https://cdn.analyticsvidhya.com/wp-content/uploads/2020/04/Feature-image-Normalization-vs.-Standardization.png>).

Here's the cookie consent and privacy policy site it impeded (significantly) the performance of some machine learning algorithms and does not work at all for others. What could be the reason behind this quirk?

(<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept

Also, what's the difference between normalization and standardization? These are two of the most commonly used feature scaling techniques in machine learning but a level of ambiguity exists in their understanding. When should you use which technique?

I will answer these questions and more in this article on feature scaling. We will also implement feature scaling in Python to give you a practice understanding of how it works for different machine learning algorithms.

*Note: I assume that you are familiar with Python and core machine learning algorithms. If you're new to this, I recommend going through the below courses:*

- [Python for Data Science \(https://courses.analyticsvidhya.com/courses/introduction-to-data-science?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization\)](https://courses.analyticsvidhya.com/courses/introduction-to-data-science?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)
- [All free Machine Learning Courses by Analytics Vidhya \(https://courses.analyticsvidhya.com/collections?category=free\)](https://courses.analyticsvidhya.com/collections?category=free)
- [Applied Machine Learning \(https://courses.analyticsvidhya.com/courses/applied-machine-learning-beginner-to-professional?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization\)](https://courses.analyticsvidhya.com/courses/applied-machine-learning-beginner-to-professional?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)

## Table of Contents

1. Why Should we Use Feature Scaling?
2. What is Normalization?
3. What is Standardization?
4. The Big Question – Normalize or Standardize?
5. Implementing Feature Scaling in Python
  - Normalization using Sklearn
  - Standardization using Sklearn
6. Applying Feature Scaling to Machine Learning Algorithms
  - K-Nearest Neighbours (KNN)
  - Support Vector Regressor
  - Decision Tree

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept

The first question we need to address – why do we need to scale the variables in our dataset? Some machine learning algorithms are sensitive to feature scaling while others are virtually invariant to it. Let me explain that in more detail.

## Gradient Descent Based Algorithms

Machine learning algorithms like [linear regression](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)), [logistic regression](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)), [neural network](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)), etc. that use gradient descent as an optimization technique require data to be scaled. Take a look at the formula for gradient descent below:

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

(<https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/gradient-descent.png>).

The presence of feature value X in the formula will affect the step size of the gradient descent. The difference in 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.*

## Distance-Based Algorithms

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our [Privacy Policy](https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) and [Terms of Use](https://www.analyticsvidhya.com/terms/) (<https://www.analyticsvidhya.com/terms/>). [K-means](https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-) (<https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means->

[clustering/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)), and SVM ([https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) 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

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/knn\\_ex.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/knn_ex.png))

Since both the features have different scales, there is a chance that higher weightage is given to features with higher magnitude. This will impact the performance of the machine learning algorithm and 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	0.816120	0.209657

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept  
([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/knn\\_ex\\_scaled.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/knn_ex_scaled.png))

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

Scaling has brought both the features into the picture and the distances are now more comparable than they were before we applied scaling.

## Tree-Based Algorithms

Tree-based algorithms ([https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)), 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!

## 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.**

Here's the formula for normalization:

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

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/Normalizing.pdf>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Here,  $X_{max}$  and  $X_{min}$  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 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.

Here's the formula for standardization:

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

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/Stand\\_eq.gif](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/Stand_eq.gif)).

$\mu$  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 vs. standardization is an eternal question among machine learning newcomers. Let me elaborate on the answer in this section.

- 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.

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.

Accept

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 compare the performance for 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](https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)). I've taken the [DataHack](https://datahack.analyticsvidhya.com/contest/all/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://datahack.analyticsvidhya.com/contest/all/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://datahack.analyticsvidhya.com/contest/all/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) 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](https://www.analyticsvidhya.com/blog/2016/02/bigmart-sales-solution-top-20/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://www.analyticsvidhya.com/blog/2016/02/bigmart-sales-solution-top-20/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2016/02/bigmart-sales-solution-top-20/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)). Those steps will enable you to reach the top 20 percentile on the hackathon leaderboard so that's worth checking out!

So, let's first split our data into training and testing sets:

```
1 # splitting training and testing data
2 from sklearn.model_selection import train_test_split
3
4 X = df
5 y = target
6
7 X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=27)
```

[8623b5e202f79c3964bd559867/raw/a423c2def59d31259b0b1e358c0700112a4b42e1/NormalizationVsStandardization\\_1.py](https://gist.github.com/aniruddha27/02432c8623b5e202f79c3964bd559867)  
[NormalizationVsStandardization\\_1.py](https://gist.github.com/aniruddha27/02432c8623b5e202f79c3964bd559867#file-normalizationvsstandardization_1-py) ([https://gist.github.com/aniruddha27/02432c8623b5e202f79c3964bd559867#file-normalizationvsstandardization\\_1-py](https://gist.github.com/aniruddha27/02432c8623b5e202f79c3964bd559867#file-normalizationvsstandardization_1-py)) hosted with ❤ by GitHub (<https://github.com>)

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your

experience on the site. By using Analytics Vidhya, you agree to our [Privacy Policy](https://www.analyticsvidhya.com/privacy-policy/)

Before moving to the feature scaling part, let's glance at the details about our data using the `pd.describe()` (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept



	Item_Weight	Item_Fat_Content	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Tier 2	Tier 3	Supermarket Type1	Supermarket Type2
count	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000
mean	12.835420	0.355676	-2.940445	140.486413	15.154884	0.830302	0.326049	0.395571	0.651071	0.111616
std	4.233450	0.478753	0.791551	62.067053	8.389349	0.598352	0.468800	0.489009	0.476667	0.314917
min	4.555000	0.000000	-5.633875	31.290000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	9.300000	0.000000	-3.467944	93.385700	9.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	12.857645	0.000000	-2.862535	142.179900	14.000000	1.000000	0.000000	0.000000	1.000000	0.000000
75%	16.000000	1.000000	-2.331264	184.495000	26.000000	1.000000	1.000000	1.000000	1.000000	0.000000
max	21.350000	1.000000	-1.113550	266.888400	28.000000	2.000000	1.000000	1.000000	1.000000	1.000000

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand\\_1.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand_1.png)).

We can see that there is a huge difference in the range of values present in our numerical features: **Item\_Visibility**, **Item\_Weight**, **Item\_MRP**, and **Outlet\_Establishment\_Year**. Let's try and fix that using feature scaling!

*Note: You will notice negative values in the Item\_Visibility feature because I have taken log-transformation to deal with the skewness in the feature.*

## Normalization using sklearn

To normalize your data, you need to import the *MinMaxScaler* from the [sklearn](https://courses.analyticsvidhya.com/courses/get-started-with-scikit-learn-sklearn?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://courses.analyticsvidhya.com/courses/get-started-with-scikit-learn-sklearn?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://courses.analyticsvidhya.com/courses/get-started-with-scikit-learn-sklearn?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) library and apply it to our dataset. So, let's do that!

```

1 # data normalization with sklearn
2 from sklearn.preprocessing import MinMaxScaler
3
4 # fit scaler on training data
5 norm = MinMaxScaler().fit(X_train)
6
7 # transform training data
8 X_train_norm = norm.transform(X_train)
9
10 # transform testing data
11 X_test_norm = norm.transform(X_test)

```

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept

5725ec02006bb3156e5483cb184/raw/5a15a3fbae1a6967171185127c458674cd021f22/NormalizationVsStandarization\_2.py)

NormalizationVsStandardization\_2.py ([https://gist.github.com/aniruddha27/a41a35725ec02006bb3156e5483cb184#file-normalizationvsstandardization\\_2-py](https://gist.github.com/aniruddha27/a41a35725ec02006bb3156e5483cb184#file-normalizationvsstandardization_2-py)) hosted with ❤️ by GitHub (<https://github.com>)

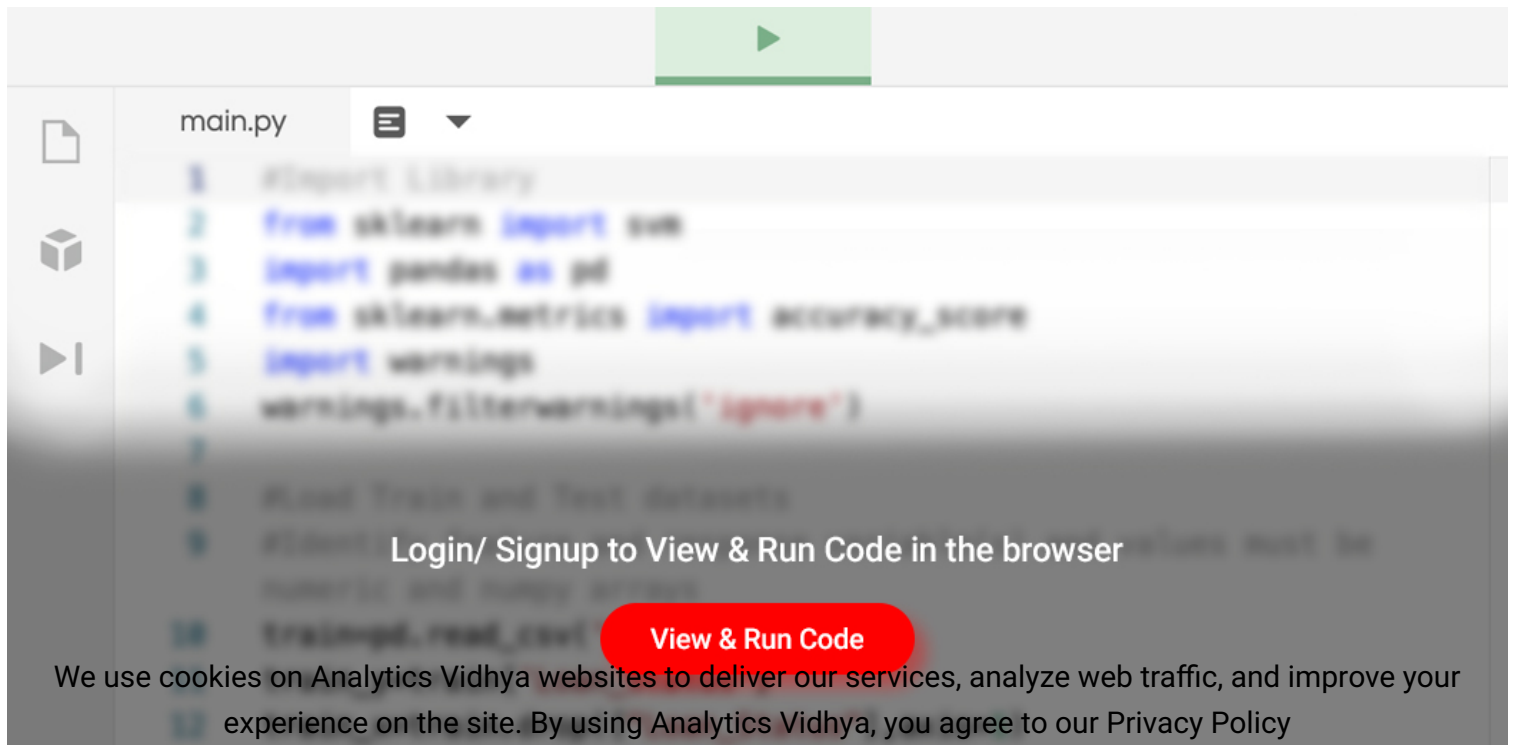
Let's see how normalization has affected our dataset:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Tier 2	Tier 3	Supermarket Type1	Supermarket Type2
count	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000
mean	0.493029	0.355676	0.595849	0.463485	0.464787	0.415151	0.326049	0.395571	0.651071	0.111616
std	0.252066	0.478753	0.175109	0.263444	0.349556	0.299176	0.468800	0.489009	0.476667	0.314917
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.282525	0.000000	0.479154	0.263566	0.208333	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.494352	0.000000	0.613084	0.470673	0.416667	0.500000	0.000000	0.000000	1.000000	0.000000
75%	0.681453	1.000000	0.730614	0.650280	0.916667	0.500000	1.000000	1.000000	1.000000	0.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand\\_2.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand_2.png))

All the features now have a minimum value of 0 and a maximum value of 1. Perfect!

Try out the above code in the live coding window below!!



```

1 # Import Library
2 from sklearn import preprocessing
3 import pandas as pd
4 from sklearn.metrics import accuracy_score
5 import warnings
6 warnings.filterwarnings("ignore")
7
8 # Load Train and Test datasets
9 train = pd.read_csv('train.csv')
10 test = pd.read_csv('test.csv')
11
12 # Normalizing the data
13 train = preprocessing.normalize(train)
14 test = preprocessing.normalize(test)
15
16 # Accuracy Score
17 accuracy = accuracy_score(test, train)
18 print("Accuracy Score: ", accuracy)

```

Login/ Signup to View & Run Code in the browser

[View & Run Code](#)

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy

(<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).  
<https://id.analyticsvidhya.com/auth/login/?next=https://www.analyticsvidhya.com/blog/2020/04/feature->  
 Accept

[scaling-machine-learning-normalization-standardization/?utm\\_source=coding-window-blog&source=coding-window-blog](https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/?utm_source=coding-window-blog&source=coding-window-blog)).

Next, let's try to standardize our data.

## Standardization using sklearn

To standardize your data, you need to import the *StandardScaler* from the sklearn library and apply it to our dataset. Here's how you can do it:

```
1 # data standardization with sklearn
2 from sklearn.preprocessing import StandardScaler
3
4 # copy of datasets
5 X_train_stand = X_train.copy()
6 X_test_stand = X_test.copy()
7
8 # numerical features
9 num_cols = ['Item_Weight', 'Item_Visibility', 'Item_MRP', 'Outlet_Establishment_Year']
10
11 # apply standardization on numerical features
12 for i in num_cols:
13
14     # fit on training data column
15     scale = StandardScaler().fit(X_train_stand[[i]])
16
17     # transform the training data column
18     X_train_stand[i] = scale.transform(X_train_stand[[i]])
19
20     # transform the testing data column
21     X_test_stand[i] = scale.transform(X_test_stand[[i]])
```

8b01e19de1cffdb5cbe703d5495/raw/948e4d05a4e65291ff54f6986f552ea398af89de/NormalizationVsStandarization\_3.py)

NormalizationVsStandarization\_3.py (https://gist.github.com/aniruddha27/965ff8b01e19de1cffdb5cbe703d5495#file-normalizationvsstandarization\_3-py) hosted With ❤️ by GitHub (https://github.com)

experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy

(https://www.analyticsvidhya.com/privacy-policy/) and Terms of Use (https://www.analyticsvidhya.com/terms/).

Accept

You would have noticed that I only applied standardization to my numerical columns and not the other One-Hot Encoded ([https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) features. Standardizing the One-Hot encoded features would mean assigning a distribution to categorical features. You don't want to do that!

But why did I not do the same while normalizing the data? Because One-Hot encoded features are already in the range between 0 to 1. So, normalization would not affect their value.

Right, let's have a look at how standardization has transformed our data:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Tier 2	Tier 3	Supermarket Type1	Supermarket Type2
count	6.818000e+03	6818.000000	6.818000e+03	6.818000e+03	6.818000e+03	6818.000000	6818.000000	6818.000000	6818.000000	6818.000000
mean	1.704754e-16	0.355676	2.342737e-16	1.233002e-16	2.051747e-17	0.830302	0.326049	0.395571	0.651071	0.111111
std	1.000073e+00	0.478753	1.000073e+00	1.000073e+00	1.000073e+00	0.598352	0.468800	0.489009	0.476667	0.314111
min	-1.956094e+00	0.000000	-3.402972e+00	-1.759459e+00	-1.329746e+00	0.000000	0.000000	0.000000	0.000000	0.000000
25%	-8.351767e-01	0.000000	-6.664603e-01	-7.589239e-01	-7.337084e-01	0.000000	0.000000	0.000000	0.000000	0.000000
50%	5.250371e-03	0.000000	9.843446e-02	2.728679e-02	-1.376709e-01	1.000000	0.000000	0.000000	1.000000	0.000000
75%	7.475728e-01	1.000000	7.696596e-01	7.091011e-01	1.292819e+00	1.000000	1.000000	1.000000	1.000000	0.000000
max	2.011410e+00	1.000000	2.308161e+00	2.036689e+00	1.531234e+00	2.000000	1.000000	1.000000	1.000000	1.000000

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand\\_3.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand_3.png)).

The numerical features are now centered on the mean with a unit standard deviation. Awesome!

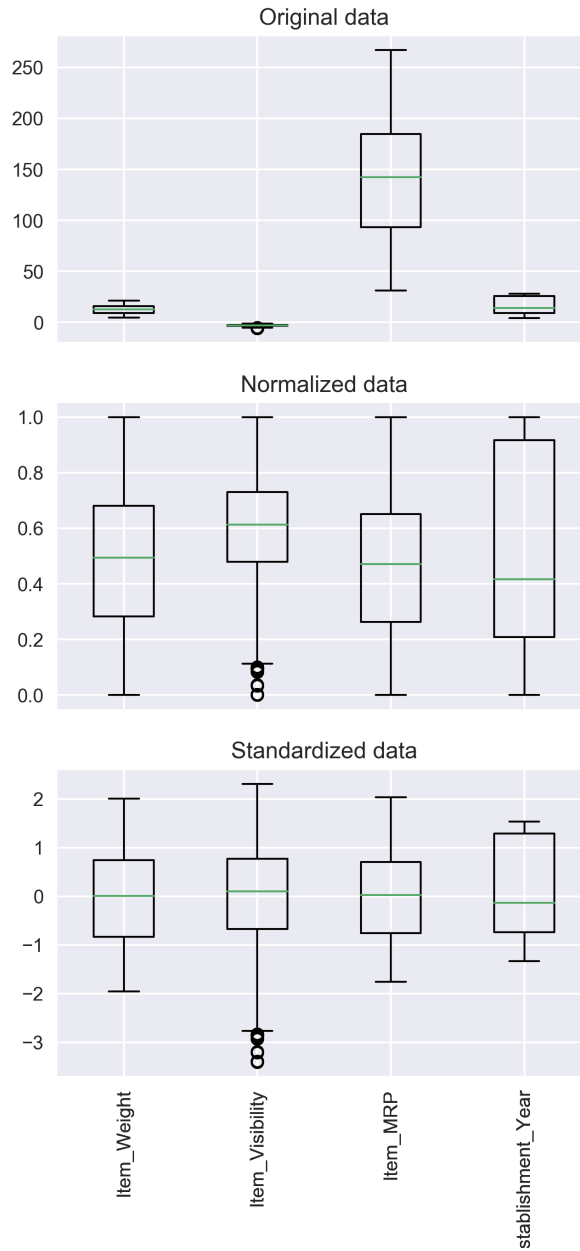
## Comparing unscaled, normalized and standardized data

It is always great to visualize your data to understand the distribution present. We can see the comparison between our unscaled and scaled data using boxplots.

You can learn more about data visualization here ([https://www.analyticsvidhya.com/blog/tag/data-visualization/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/tag/data-visualization/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)).

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept



([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand\\_box\\_plots-1.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand_box_plots-1.png))

You can notice how scaling the features brings everything into perspective. The features are now more comparable and will have a similar effect on the learning models.  
 We use cookies on Analytics Vidhya websites to deliver our services, analyze site usage, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept

## Applying Scaling to Machine Learning Algorithms

It's now time to train some machine learning algorithms on our data to compare the effects of different scaling techniques on the performance of the algorithm. I want to see the effect of scaling on three algorithms in particular: K-Nearest Neighbours, Support Vector Regressor, and Decision Tree.

### K-Nearest Neighbours

Like we saw before, KNN is a distance-based algorithm that is affected by the range of features. Let's see how it performs on our data, before and after scaling:

```

1 # training a KNN model
2 from sklearn.neighbors import KNeighborsRegressor
3 # measuring RMSE score
4 from sklearn.metrics import mean_squared_error
5
6 # knn
7 knn = KNeighborsRegressor(n_neighbors=7)
8
9 rmse = []
10
11 # raw, normalized and standardized training and testing data
12 trainX = [X_train, X_train_norm, X_train_stand]
13 testX = [X_test, X_test_norm, X_test_stand]
14
15 # model fitting and measuring RMSE
16 for i in range(len(trainX)):
17
18     # fit
19     knn.fit(trainX[i],y_train)
20     # predict
21     pred = knn.predict(testX[i])
22     # RMSE
23     rmse.append(np.sqrt(mean_squared_error(y_test,pred)))
24
25 # visualize the results
26 df_knn

```

Accept

NormalizationVsStandarization\_4.py (<https://gist.github.com/aniruddha27/66119a2050fc808d2bdb7d4544ae75b6#file->

a2050fc808d2bdb7d4544ae75b6/raw/f7d5d7854dfc734b910aefc9513ab7e3140c175e/NormalizationVsStandarization\_4.py) normalizationvsstandarization\_4-py) hosted with ❤️ by GitHub (<https://github.com>)

	RMSE
Original	1319.283626
Normalized	1174.205859
Standardized	1183.448734

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand\\_knn.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand_knn.png)).

You can see that scaling the features has brought down the RMSE score of our KNN model. Specifically, the normalized data performs a tad bit better than the standardized data.

*Note: I am measuring the RMSE here because this competition evaluates the RMSE.*

## Support Vector Regressor

SVR ([https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) is another distance-based algorithm. So let's check out whether it works better with normalization or standardization:

```

1  # training an SVR model
2  from sklearn.svm import SVR
3  # measuring RMSE score
4  from sklearn.metrics import mean_squared_error
5
6  # SVR
7  svr = SVR(kernel='rbf', C=5)
8
9  rmse = []
10
11 # raw, normalized and standardized training and testing data
12 trainX = [X_train, X_train_norm, X_train_stand]
13 testX = [X_test, X_test_norm, X_test_stand]
14
15 # model fitting and measuring RMSE
16 for i in range(len(trainX)):

```

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy

(<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept

```

17
18     # fit
19     svr.fit(trainX[i],y_train)
20     # predict
21     pred = svr.predict(testX[i])
22     # RMSE
23     rmse.append(np.sqrt(mean_squared_error(y_test,pred)))
24
25 # visualizing the result
26 df_svr = pd.DataFrame({'RMSE':rmse},index=['Original','Normalized','Standardized'])
27 df_svr

```

527ef006d7b58cae03ba59e9df/raw/00949b28470a33a67eca94629e52f586cacab13d/NormalizationVsStandarization\_5.py) NormalizationVsStandarization\_5.py ([https://gist.github.com/aniruddha27/a49f58527ef006d7b58cae03ba59e9df#file-normalizationvsstandarization\\_5-py](https://gist.github.com/aniruddha27/a49f58527ef006d7b58cae03ba59e9df#file-normalizationvsstandarization_5-py)) hosted with ❤️ by GitHub (<https://github.com>)

	RMSE
<b>Original</b>	1742.379686
<b>Normalized</b>	1655.643047
<b>Standardized</b>	1453.442942

([https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand\\_svr.png](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/NormVsStand_svr.png)).

We can see that scaling the features does bring down the RMSE score. And the standardized data has performed better than the normalized data. Why do you think that's the case?

The [sklearn documentation \(https://scikit-learn.org/stable/modules/preprocessing.html#standardization-or-mean-removal-and-variance-scaling\)](https://scikit-learn.org/stable/modules/preprocessing.html#standardization-or-mean-removal-and-variance-scaling) states that SVM, with RBF kernel, assumes that all the features are centered around zero and variance is of the same order. This is because a feature with a variance greater than that of others prevents the estimator from learning from all the features. Great!

## Decision Tree

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

```

1 # training a Decision Tree model
2 from sklearn.tree import DecisionTreeRegressor

```



```

3 # measuring RMSE score
4 from sklearn.metrics import mean_squared_error
5
6 # Decision tree
7 dt = DecisionTreeRegressor(max_depth=10,random_state=27)
8
9 rmse = []
10
11 # raw, normalized and standardized training and testing data
12 trainX = [X_train,X_train_norm,X_train_stand]
13 testX = [X_test,X_test_norm,X_test_stand]
14
15 # model fitting and measuring RMSE
16 for i in range(len(trainX)):
17
18     # fit
19     dt.fit(trainX[i],y_train)
20     # predict
21     pred = dt.predict(testX[i])
22     # RMSE
23     rmse.append(np.sqrt(mean_squared_error(y_test,pred)))
24
25 # visualizing the result
26 df_dt = pd.DataFrame({'RMSE':rmse},index=['Original','Normalized','Standardized'])
27 df_dt

```

5390dc75dacf1a9506a92182d6c/raw/049c34d17ffa4889edda3b89c7827dcf97e5b498/NormalizationVsStandarization\_6.py) NormalizationVsStandarization\_6.py ([https://gist.github.com/aniruddha27/6734a5390dc75dacf1a9506a92182d6c#file-normalizationvsstandarization\\_6-py](https://gist.github.com/aniruddha27/6734a5390dc75dacf1a9506a92182d6c#file-normalizationvsstandarization_6-py)) hosted with ❤️ by GitHub (<https://github.com>)

RMSE	
<b>Original</b>	1245.37439
<b>Normalized</b>	1245.37439
<b>Standardized</b>	1245.37439

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>). You can see that analyticsvidhya.com has moved a and Terms of Use (<https://www.analyticsvidhya.com/terms/>). using tree-based algorithms on your data!

Accept

## End Notes

This tutorial covered the relevance of using feature scaling on your data and how normalization and standardization have varying effects on the working of machine learning algorithms

Keep in mind that there is no correct answer to when to use normalization over standardization and vice-versa. It all depends on your data and the algorithm you are using.

As a next step, I encourage you to try out feature scaling with other algorithms and figure out what works best – normalization or standardization? I recommend you use the [BigMart Sales data](https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) ([https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/?utm\\_source=blog&utm\\_medium=feature-scaling-machine-learning-normalization-standardization](https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)) for that purpose to maintain the continuity with this article. And don't forget to share your insights in the comments section below!

You can also read this article on our Mobile APP

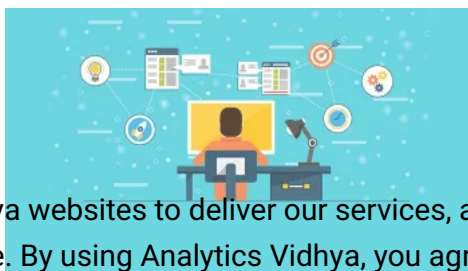


([https://play.google.com/store/apps/details?id=com.analyticsvidhya.android&utm\\_source=blog\\_article&utm\\_campaign=blog&pcampaignid=MKT-Other-global-all-co-prtnr-py-PartBadge-Mar2515-1](https://play.google.com/store/apps/details?id=com.analyticsvidhya.android&utm_source=blog_article&utm_campaign=blog&pcampaignid=MKT-Other-global-all-co-prtnr-py-PartBadge-Mar2515-1)).



(<https://apps.apple.com/us/app/analytics-vidhya/id1470025572>).

## Related Articles



We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>). (<https://www.analyticsvidhya.com/blog/2020/04/feature-engineering-techniques-of-feature-transformation-and-engineering-feature-improvements-> [feature-engineering-techniques-](https://www.analyticsvidhya.com/blog/2020/04/feature-engineering-techniques-of-feature-transformation-and-engineering-feature-improvements-) [of-feature-transformation-and-](https://www.analyticsvidhya.com/blog/2020/04/feature-engineering-techniques-of-feature-transformation-and-engineering-feature-improvements-)

[scaling/](#)).

Feature Engineering (Feature Improvements - Scaling) (<https://www.analyticsvidhya.com/blog/2020/12/feature-engineering-feature-improvements-scaling/>)

[machine-learning/](#)).

7 Feature Engineering Techniques in Machine Learning You Should Know (<https://www.analyticsvidhya.com/blog/2020/10/7-feature-engineering-techniques-machine-learning/>)

[scaling/](#)).

Feature Transformation and Scaling Techniques to Boost Your Model Performance (<https://www.analyticsvidhya.com/blog/2020/07/types-of-feature-transformation-and-scaling/>)



**TAGS :** [FEATURE SCALING](https://www.analyticsvidhya.com/blog/tag/feature-scaling/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/FEATURE-SCALING/](https://www.analyticsvidhya.com/blog/tag/feature-scaling/)), [FEATURE SCALING MACHINE LEARNING](https://www.analyticsvidhya.com/blog/tag/feature-scaling-machine-learning/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/FEATURE-SCALING-MACHINE-LEARNING/](https://www.analyticsvidhya.com/blog/tag/feature-scaling-machine-learning/)), [FEATURE SCALING PYTHON](https://www.analyticsvidhya.com/blog/tag/feature-scaling-python/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/FEATURE-SCALING-PYTHON/](https://www.analyticsvidhya.com/blog/tag/feature-scaling-python/)), [LIVE CODING](https://www.analyticsvidhya.com/blog/tag/live-coding/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/LIVE-CODING/](https://www.analyticsvidhya.com/blog/tag/live-coding/)), [NORMALIZATION VS. STANDARDIZATION](https://www.analyticsvidhya.com/blog/tag/normalization-vs-standardization/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/NORMALIZATION-VS-STANDARDIZATION/](https://www.analyticsvidhya.com/blog/tag/normalization-vs-standardization/)), [NORMALIZATION](https://www.analyticsvidhya.com/blog/tag/normalization/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/NORMALIZATION/](https://www.analyticsvidhya.com/blog/tag/normalization/)), [STANDARDIZATION](https://www.analyticsvidhya.com/blog/tag/standardization/) ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/STANDARDIZATION/](https://www.analyticsvidhya.com/blog/tag/standardization/))

NEXT ARTICLE

## Supervised Learning vs. Unsupervised Learning – A Quick Guide for Beginners

(<https://www.analyticsvidhya.com/blog/2020/04/supervised-learning-unsupervised-learning/>)

...

PREVIOUS ARTICLE

## 6 Open Source Data Science Projects to Make you Industry Ready!

(<https://www.analyticsvidhya.com/blog/2020/04/6-open-source-data-science-projects-industry-ready/>)

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our [Privacy Policy](https://www.analyticsvidhya.com/privacy-policy/) (<https://www.analyticsvidhya.com/privacy-policy/>) and [Terms of Use](https://www.analyticsvidhya.com/terms/) (<https://www.analyticsvidhya.com/terms/>).

Accept



(<https://www.analyticsvidhya.com/blog/author/aniruddha/>).

Aniruddha Bhandari (<https://www.analyticsvidhya.com/blog/author/aniruddha/>).

I am on a journey to becoming a data scientist. I love to unravel trends in data, visualize it and predict the future with ML algorithms! But the most satisfying part of this journey is sharing my learnings, from the challenges that I face, with the community to make the world a better place!

This article is quite old and you might not get a prompt response from the author. We request you to post this comment on Analytics Vidhya's **Discussion portal** (<https://discuss.analyticsvidhya.com/>) to get your queries resolved

## 16 COMMENTS



**ALI**

[Reply](#)

April 12, 2020 at 11:18 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161122>).

Excelent article! Thank you very much for sharing. I have one question. In the post you say: "It is a good practice to fit the scaler on the training data and then use it to transform the testing data.", but I didn't see that in the code you posted. Am I wrong? How would one "fit the scaler on the training data and then use it to transform the testing data"? Thanks a lot again



**ANIRUDDHA BHANDARI**

[Reply](#)

April 13, 2020 at 11:44 am (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161122>).

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy

(<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>). You fit the scaler on the training data so that it can calculate the necessary parameters, like mean and standard deviation for standardization, and store it for later use using the fit() method. Later you use the transform() method.

Accept

function to apply the same transformation on both, train and test dataset.

I have used this approach for both, normalization and standardization, in the article in the gists "NormalizationVsStandardization\_2.py" and "NormalizationVsStandardization\_3.py" respectively.

I hope this cleared your doubt.

Thanks



## SAHIL KAMBOJ

[Reply](#)

April 28, 2020 at 4:25 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161208>).

Good article! Thank you very much for sharing. I have one question.

What the difference between sklearn.preprocessing import MinMaxScaler Normalization and sklearn.preprocessing.Normalizer?

When to use MinMaxScaler and when to Normalize?



## ANIRUDDHA BHANDARI

[Reply](#)

April 29, 2020 at 2:15 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161217>).

Hi

I hope MinMaxScaler is already clear from the article.

[Normalizer \(https://scikit-](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html#sklearn.preprocessing.Normalizer)

[learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html#sklearn.preprocessing.Normalizer\)](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html#sklearn.preprocessing.Normalizer)

is also a normalization technique. The only difference is the way it computes the normalized values. By default, it is calculating the L2 norm of the row values i.e. each element of a row is normalized by the square root of the sum of squared values of all elements in that row.

As mentioned in the documentation, it is useful in text classification where the dot product of two [Tf-IDF](https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/) (<https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/>) vectors gives a cosine similarity between the different sentences/documents in the dataset. Other than that, as I mentioned in the article, there is no sure way to know which scaling technique should be used when. The best way is to create multiple scaled copies of the data and then try them out and see which one gives the best result.

Hope this helps.

We use [Analytics Vidhya](#) websites to deliver our services, analyze web traffic, and improve your [experience on the site](#). By using [Analytics Vidhya](#), you agree to our [Privacy Policy](#) (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

May 23, 2020 at 12:00 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/>)

(<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept

Excellent article! Easy to understand and good coverage

One question: I see that there is a `scale()` function as well from `sklearn` and short description suggest it to be similar to `StandardScaler` i.e. scaling to unit variance

I could not find more than this explanation. Please can you suggest which one to use which scenario?

Thanks in advance!



### ANIRUDDHA BHANDARI

[Reply](#)

May 24, 2020 at 12:41 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161579>).

Hi Subhash

I notice two differences between the two functions. First was that the `scale()` function allows you to standardize your data along any axis. This means that you could even standardize your data row-wise as opposed to feature-wise, which is what happens in `StandardScaler()`. Second difference was that the `scale` function has no `fit` and `transform` methods, so you cannot apply the same scaling to your test dataset.

I would suggest using the `StandardScaler()` function as I have never used the `scale()` personally.

I hope this helps!

### GOLLA KEDARKUMAR

[Reply](#)

May 24, 2020 at 9:56 am (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161575>).

Hi ANIRUDDHA,

If we use the same scaler for train and testing, does it affect the testing data because in standardization we need to use the mean of the data. If we take the mean of the train data and scale the test data, it will influence the test data, right?



### ANIRUDDHA BHANDARI

[Reply](#)

May 24, 2020 at 12:11 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161578>).

Scaling your test data according to the train data makes sure that the test data is on the same scale as the training data on which our model was trained on. This way our model will be able to apply the learnings from the training dataset to the testing dataset, which is exactly what we want. If, instead, we scale the test data differently, then our model might not be able to discern that difference, thereby giving incorrect outputs. That way we will never know how well our model is performing.

Accept

**INAS**[Reply](#)

May 27, 2020 at 6:42 am (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-161630>).

Excellent article, thank you for sharing.

**SOUMADIP ROY**[Reply](#)

July 4, 2020 at 12:59 am (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162136>).

This is an excellent write up. Thanks for this.

**HARSHVARDHAN BHATT**[Reply](#)

July 5, 2020 at 3:03 am (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162151>).

That graphs really helps in putting things in perspective...thanks !

**ARNOB**[Reply](#)

July 5, 2020 at 4:13 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162155>).

Hey bro! Great article. It covered a lots of topics that were unclear to me before. I have a basic question. How can I check my data after normalization. You have mentioned to use `pd.describe()` in "Normalization using sklearn" section. But when I use it I get an error – " module 'pandas' has no attribute 'describe'".

Can you tell me how to check my data after normalization?

Thank you for your time.

**ANIRUDDHA BHANDARI**[Reply](#)

August 22, 2020 at 8:06 pm (<https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162155>).

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).  
Hi Arnob, glad you liked the article. The command you are looking for is `df.describe()` not `pd.describe()`. Try using that, it should work.

Accept

**DEEPS**[Reply](#)

[July 15, 2020 at 7:59 pm \(https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162296\)](https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162296)

Excellent article !

---

**KUNAL**[Reply](#)

[July 31, 2020 at 1:19 pm \(https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162543\)](https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162543)

Thanks for Great Article..!!!

---

**ZINEB**[Reply](#)

[August 17, 2020 at 3:29 pm \(https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162737\)](https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/#comment-162737)

Thanks Bhandari.

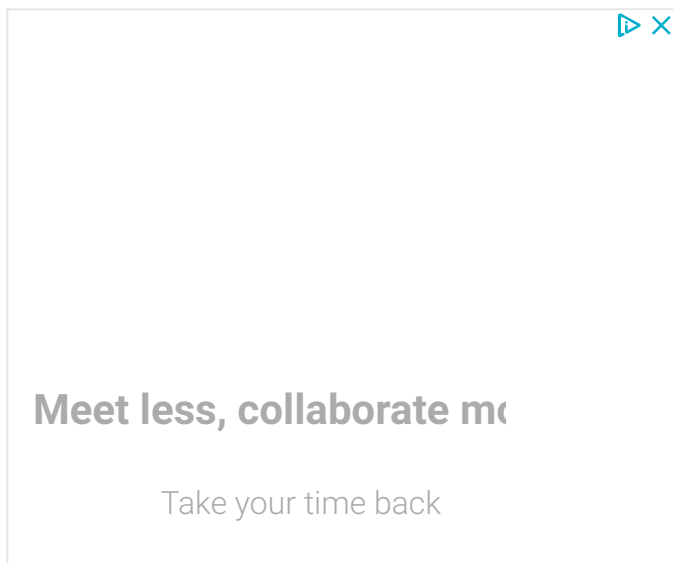
Easy to understand and very helpful.

---

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept





## POPULAR POSTS

---

Understanding Delimiters in Pandas read\_csv() Function

([https://www.analyticsvidhya.com/blog/2021/04/delimiters-in-pandas-read\\_csv-function/](https://www.analyticsvidhya.com/blog/2021/04/delimiters-in-pandas-read_csv-function/))

40 Questions to test a Data Scientist on Clustering Techniques (Skill test Solution)

(<https://www.analyticsvidhya.com/blog/2017/02/test-data-scientist-clustering/>)

How to Download Kaggle Datasets using Jupyter Notebook

(<https://www.analyticsvidhya.com/blog/2021/04/how-to-download-kaggle-datasets-using-jupyter-notebook/>)

Python List Programs For Absolute Beginners (<https://www.analyticsvidhya.com/blog/2021/04/python-list-programs-for-absolute-beginners/>)

Commonly used Machine Learning Algorithms (with Python and R Codes)

(<https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/>)

30 Questions to test a data scientist on K-Nearest Neighbors (kNN) Algorithm

(<https://www.analyticsvidhya.com/blog/2017/09/30-questions-test-k-nearest-neighbors-algorithm/>)

Introductory guide on Linear Programming for (aspiring) data scientists

(<https://www.analyticsvidhya.com/blog/2017/02/introductory-guide-on-linear-programming-explained-in-simple-english/>)

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy

(<https://www.analyticsvidhya.com/blog/2017/01/must-know-questions-deep-learning/>)

(<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

Accept



## CAREER RESOURCES



**16 Key Questions You Should Answer Before Transitioning into Data Science** (<https://www.analyticsvidhya.com/16-key-questions-data-science-career-transition/>)

&utm\_source=Blog&utm\_medium=CareerResourceWidget)

NOVEMBER 23, 2020

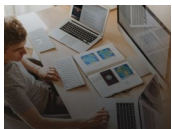


**Here's What You Need to Know to Become a Data Scientist!**

(<https://www.analyticsvidhya.com/blog/2021/01/heres-what-you-need-to-know-to-become-a-data-scientist/>)

&utm\_source=Blog&utm\_medium=CareerResourceWidget)

JANUARY 22, 2021



**These 7 Signs Show you have Data Scientist Potential!**

(<https://www.analyticsvidhya.com/blog/2020/12/these-7-signs-show-you-have-data-scientist-potential/>)

&utm\_source=Blog&utm\_medium=CareerResourceWidget)

DECEMBER 3, 2020



**How To Have a Career in Data Science (Business Analytics)?**

(<https://www.analyticsvidhya.com/blog/2020/11/how-to-have-a-career-in-data-science-business-analytics/>)

&utm\_source=Blog&utm\_medium=CareerResourceWidget)

Accept

NOVEMBER 26, 2020



## Should I become a data scientist (or a business analyst)?

(<https://www.analyticsvidhya.com/blog/2020/11/become-data-scientist-business-analyst/>)

&utm\_source=Blog&utm\_medium=CareerResourceWidget)

NOVEMBER 24, 2020

## RECENT POSTS

---

### How AI Supports Logistics Industry and Transportation Businesses

(<https://www.analyticsvidhya.com/blog/2021/05/how-ai-supports-logistics-industry-and-transportation-businesses/>)

MAY 11, 2021

### Making Programming with Date and Time, less painful

(<https://www.analyticsvidhya.com/blog/2021/05/making-programming-with-date-and-time-less-painless/>)

MAY 11, 2021

### Is there any need of Deep Learning? (<https://www.analyticsvidhya.com/blog/2021/05/is-there-any-need-of-deep-learning/>)

MAY 11, 2021

### Data Science Use Cases in Retail Industry (<https://www.analyticsvidhya.com/blog/2021/05/data-science-use-cases-in-retail-industry/>)

MAY 11, 2021

**Analytics Vidhya**

**Roadmap to Become a Data Scientist in 180 Days**

**WITH JOB GUARANTEE\***

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our Privacy Policy (<https://www.analyticsvidhya.com/privacy-policy/>) and Terms of Use (<https://www.analyticsvidhya.com/terms/>).

**Download Roadmap**

(<https://bootcamp.analyticsvidhya.com/?>)

utm\_source=Blog&utm\_medium=stickybanner1#av-roadmap)

**Data Science Blogathon**

- Assured INR 100 for every published article
- Total prizes worth INR 1 Lakh+
- A chance to win iPad(8th Gen)

6th April - 31st May 21

**Register Now**

https://www.analyticsvidhya.com/contest/data-science-



(https://www.analyticsvidhya.com/)

Download on the App Store. Get it on Google Play. See details?

App



id=com.analyticsvidhya.android)



(https://apps.apple.com/us/app/analytics-

vidhya/id1470025572)

### Analytics Vidhya

About Us (<https://www.analyticsvidhya.com/about-me/>)

Our Team (<https://www.analyticsvidhya.com/about-me/team/>)

Careers (<https://www.analyticsvidhya.com/about-me/career-analytics-vidhya/>)

Contact us (<https://www.analyticsvidhya.com/contact/>)

### Data Science

Blog (<https://www.analyticsvidhya.com/blog/>)

Hackathon (<https://datahack.analyticsvidhya.com/>)

Discussions (<https://discuss.analyticsvidhya.com/>)

Apply Jobs (<https://www.analyticsvidhya.com/jobs/>)

### Companies

Post Jobs (<https://www.analyticsvidhya.com/corporate/>)

Trainings (<https://courses.analyticsvidhya.com/>)

Hiring Hackathons (<https://datahack.analyticsvidhya.com/>)

Advertising (<https://www.analyticsvidhya.com/contact/>)

### Visit us

in



([https://www.linkedin.com/company/analytics-](https://www.linkedin.com/company/analytics-vidhya/)

<https://www.facebook.com/analyticsvidhya/>)

We use cookies on Analytics Vidhya websites to enhance our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our [Privacy Policy](#) and [Terms of Use](#).

Accept