

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

You have 2 free stories left this month. Sign up and get an extra one for free.

# Scale, Standardize, or Normalize with Scikit-Learn

When to use `MinMaxScaler`, `RobustScaler`, `StandardScaler`, and `Normalizer`



Jeff Hale

Mar 4, 2019 · 7 min read ★

Many machine learning algorithms work better when features are on a relatively similar scale and close to normally distributed. *MinMaxScaler*, *RobustScaler*, *StandardScaler*, and *Normalizer* are scikit-learn methods to preprocess data for machine learning. Which method you need, if any, depends on your model type and your feature values.

This guide will highlight the differences and similarities among these methods and help you learn when to reach for which tool.



We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

## Scales

As often as these methods appear in machine learning workflows, I found it difficult to find information about which of them to use when. Commentators often use the terms *scale*, *standardize*, and *normalize* interchangeably. However, there are some differences and the four scikit-learn functions we will examine do different things.

First, a few housekeeping notes:

- The Jupyter Notebook on which this article is based can be found [here](#).
- In this article, we aren't looking at log transformations or other transformations aimed at reducing the homoscedasticity of the errors.
- This guide is current as of scikit-learn v0.20.3.

## What do These Terms Mean?

Scale generally means to change the **range** of the values. The shape of the distribution doesn't change. Think about how a scale model of a building has the same proportions as the original, just smaller. That's why we say it is drawn to scale. The range is often set at 0 to 1.

Standardize generally means changing the values so that the distribution **standard** deviation from the mean equals one. It outputs something very close to a normal distribution. Scaling is often implied.

Normalize can be used to mean either of the above things (and more!). I suggest you avoid the term *normalize*, because it has many definitions and is prone to creating confusion.

If you use any of these terms in your communication, I strongly suggest you define them.

## Why Scale, Standardize, or Normalize?

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

- linear and logistic regression
- nearest neighbors
- neural networks
- support vector machines with radial bias kernel functions
- principal components analysis
- linear discriminant analysis

Scaling and standardizing can help features arrive in more digestible form for these algorithms.

The four scikit-learn preprocessing methods we are examining follow the API shown below. *X\_train* and *X\_test* are the usual numpy ndarrays or pandas DataFrames.

```
from sklearn import preprocessing

mm_scaler = preprocessing.MinMaxScaler()
X_train_minmax = mm_scaler.fit_transform(X_train)

mm_scaler.transform(X_test)
```

We'll look at a number of distributions and apply each of the four scikit-learn methods to them.

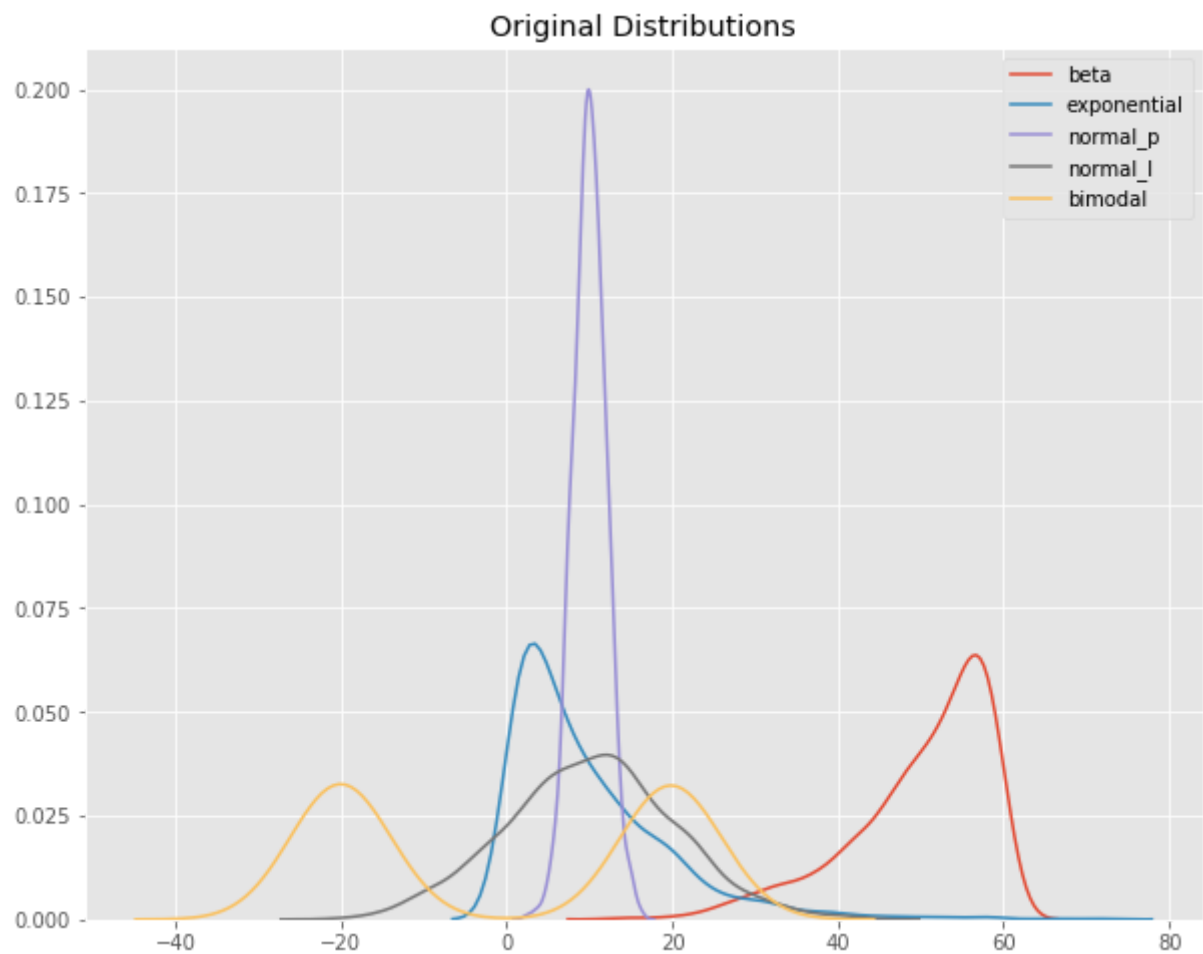
## Original Data

I created four distributions with different characteristics. The distributions are:

- **beta** — with negative skew
- **exponential** — with positive skew
- **normal\_p** — normal, platykurtic
- **normal\_l** — normal, leptokurtic

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



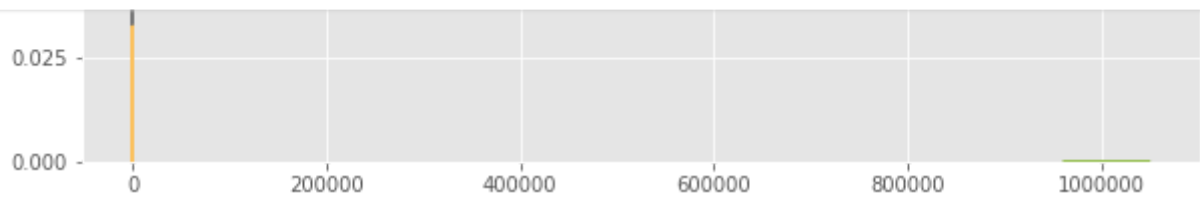
Then I added a fifth distribution with much larger values (normally distributed) — **normal\_big**.

Now our kdeplot looks like this:



We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



Squint hard at the monitor and you might notice the tiny green bar of big values to the right. Here are the descriptive statistics for our features.

	beta	exponential	normal_p	normal_l	bimodal	normal_big
<b>count</b>	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03
<b>mean</b>	50.024249	10.028387	9.994006	10.175745	-0.076622	1.000259e+06
<b>std</b>	8.474545	9.733928	2.013971	10.104004	20.165208	9.935564e+03
<b>min</b>	13.854022	0.007617	2.356844	-19.539980	-28.709856	9.692079e+05
<b>25%</b>	45.793283	2.951421	8.687478	3.566822	-19.995311	9.936191e+05
<b>50%</b>	52.337504	7.018565	9.983498	10.326331	0.237049	1.000241e+06
<b>75%</b>	56.722191	14.022485	11.306914	16.615057	19.891202	1.007335e+06
<b>max</b>	59.990640	71.344341	16.214364	42.072915	28.252151	1.040677e+06

Alright, let's start scaling!

## MinMaxScaler

For each value in a feature, MinMaxScaler subtracts the minimum value in the feature and then divides by the range. The range is the difference between the original maximum and original minimum.

MinMaxScaler preserves the shape of the original distribution. It doesn't meaningfully change the information embedded in the original data.

Note that MinMaxScaler doesn't reduce the importance of outliers.

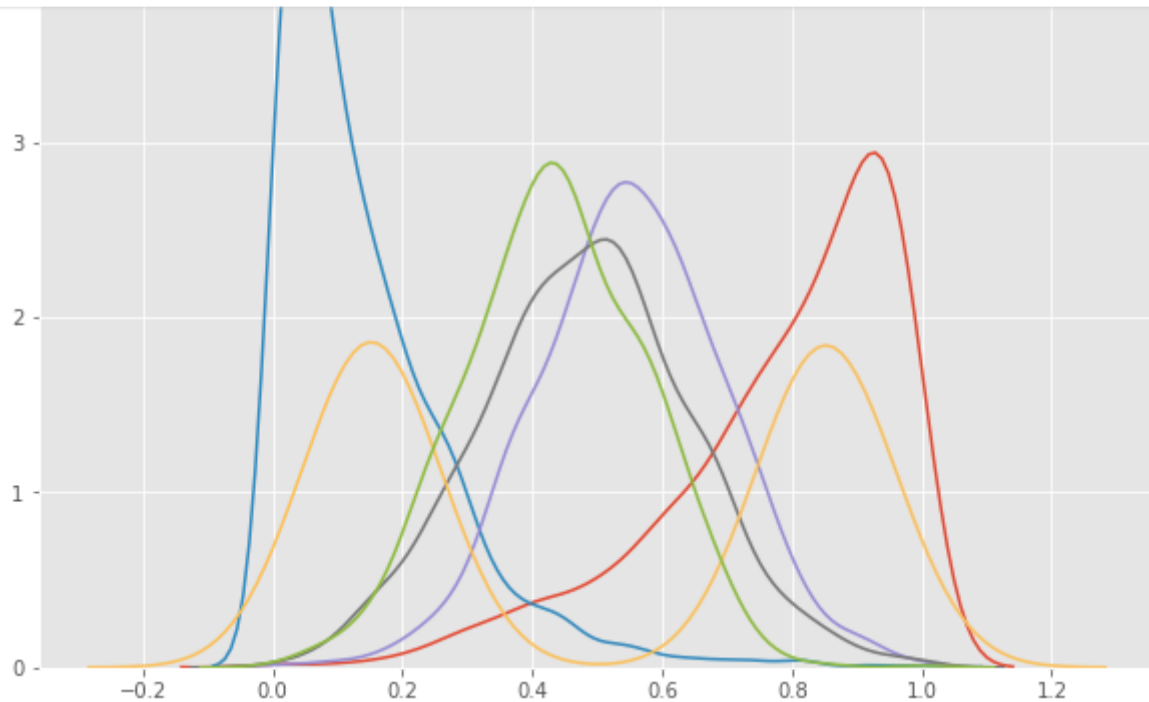
The default range for the feature returned by MinMaxScaler is 0 to 1.

Here's the kdeplot after MinMaxScaler has been applied.



We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



Notice how the features are all on the same relative scale. The relative spaces between each feature's values have been maintained.

MinMaxScaler is a good place to start unless you know you want your feature to have a normal distribution or want outliers to have reduced influence.

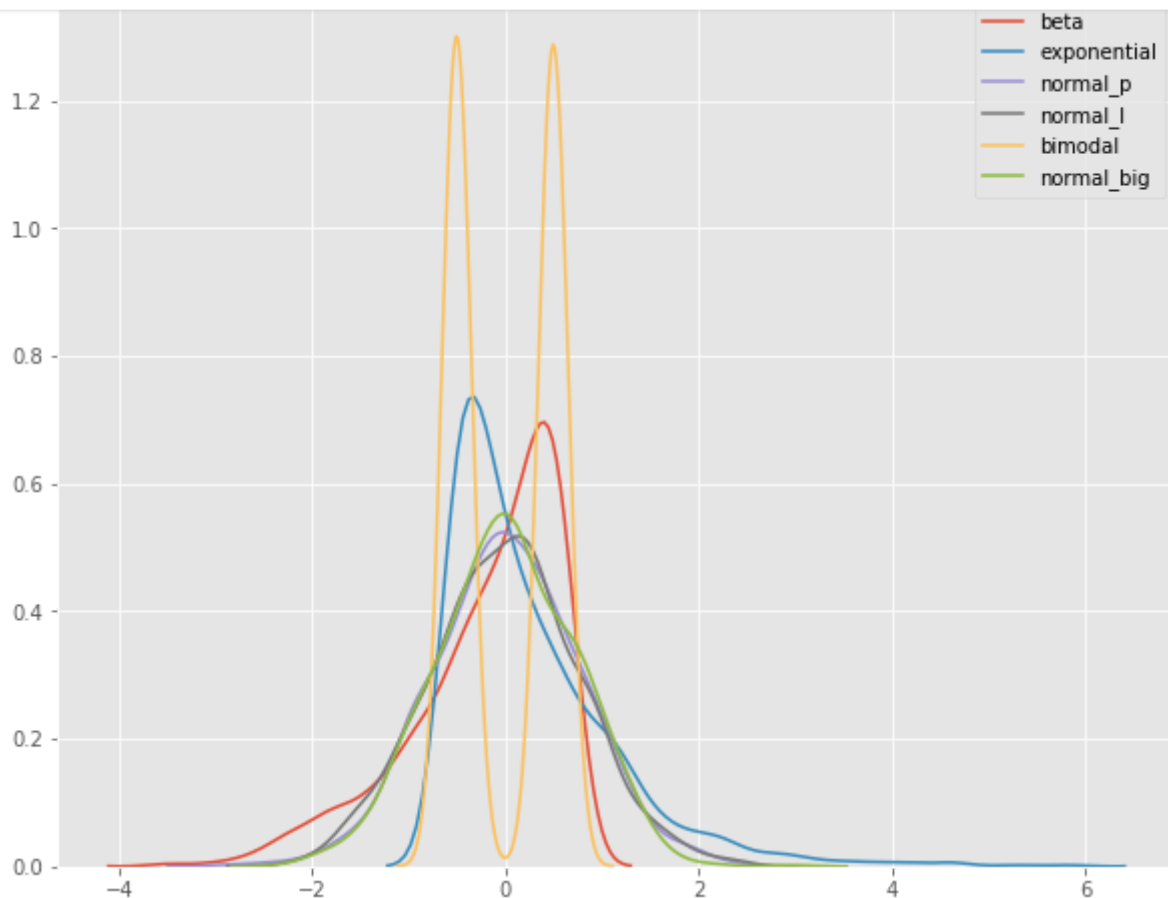


Different types of scales

## RobustScaler

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



Like `MinMaxScaler`, our feature with large values — *normal-big* — is now of similar scale to the other features. Note that `RobustScaler` does not scale the data into a predetermined interval like `MinMaxScaler`. It does not meet the strict definition of *scale* I introduced earlier.

Note that the range for each feature after `RobustScaler` is applied is larger than it was for `MinMaxScaler`.

Use `RobustScaler` if you want to reduce the effects of outliers, relative to `MinMaxScaler`.

Now let's turn to `StandardScaler`.

## StandardScaler

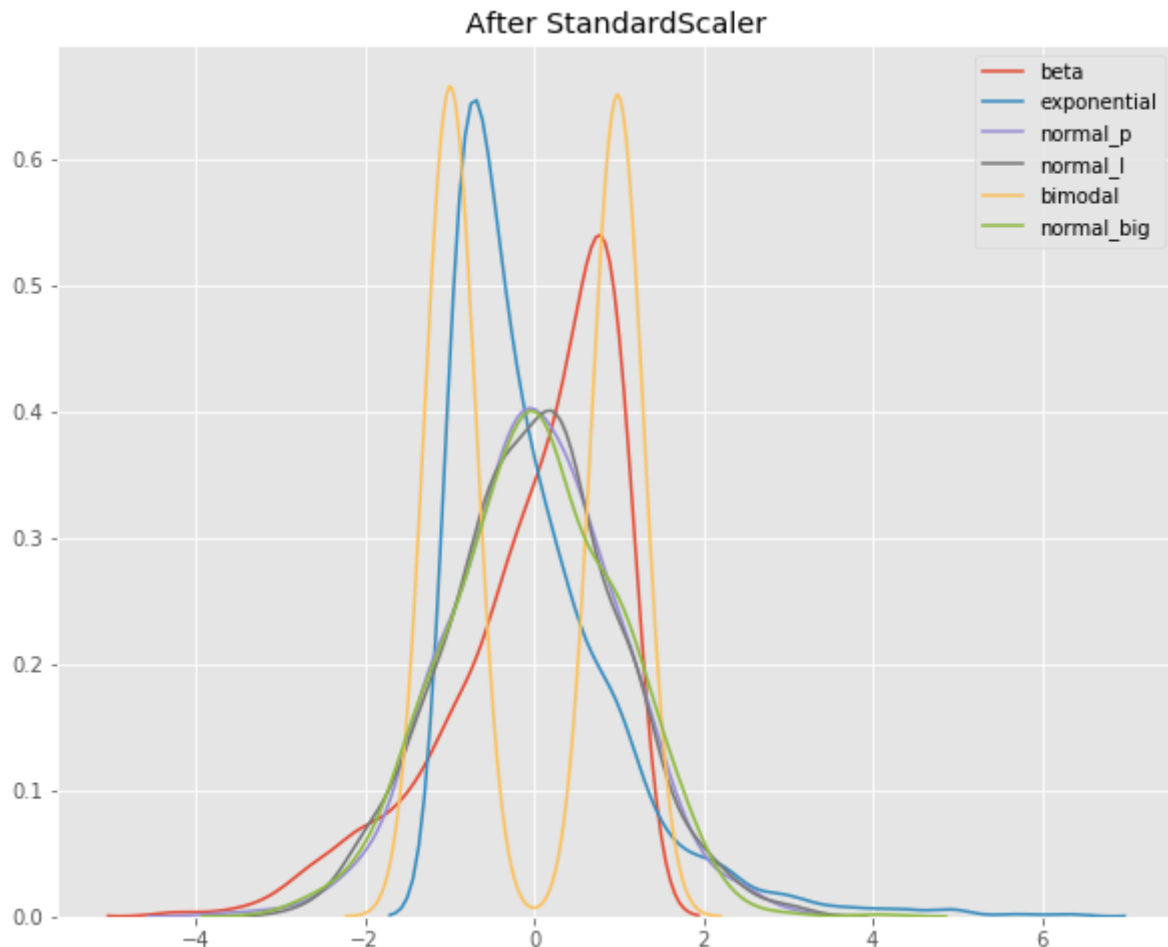
`StandardScaler` standardizes a feature by subtracting the mean and then scaling to unit variance. Unit variance means dividing all the values by the standard deviation.

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

squared = 1.

StandardScaler makes the mean of the distribution 0. About 68% of the values will lie between -1 and 1.



In the plot above, you can see that all four distributions have a mean close to zero and unit variance. The values are on a similar scale, but the range is larger than after MinMaxScaler.

Deep learning algorithms often call for zero mean and unit variance. Regression-type algorithms also benefit from normally distributed data with small sample sizes.

StandardScaler does distort the relative distances between the feature values, so it's generally my second choice in this family of transformations.

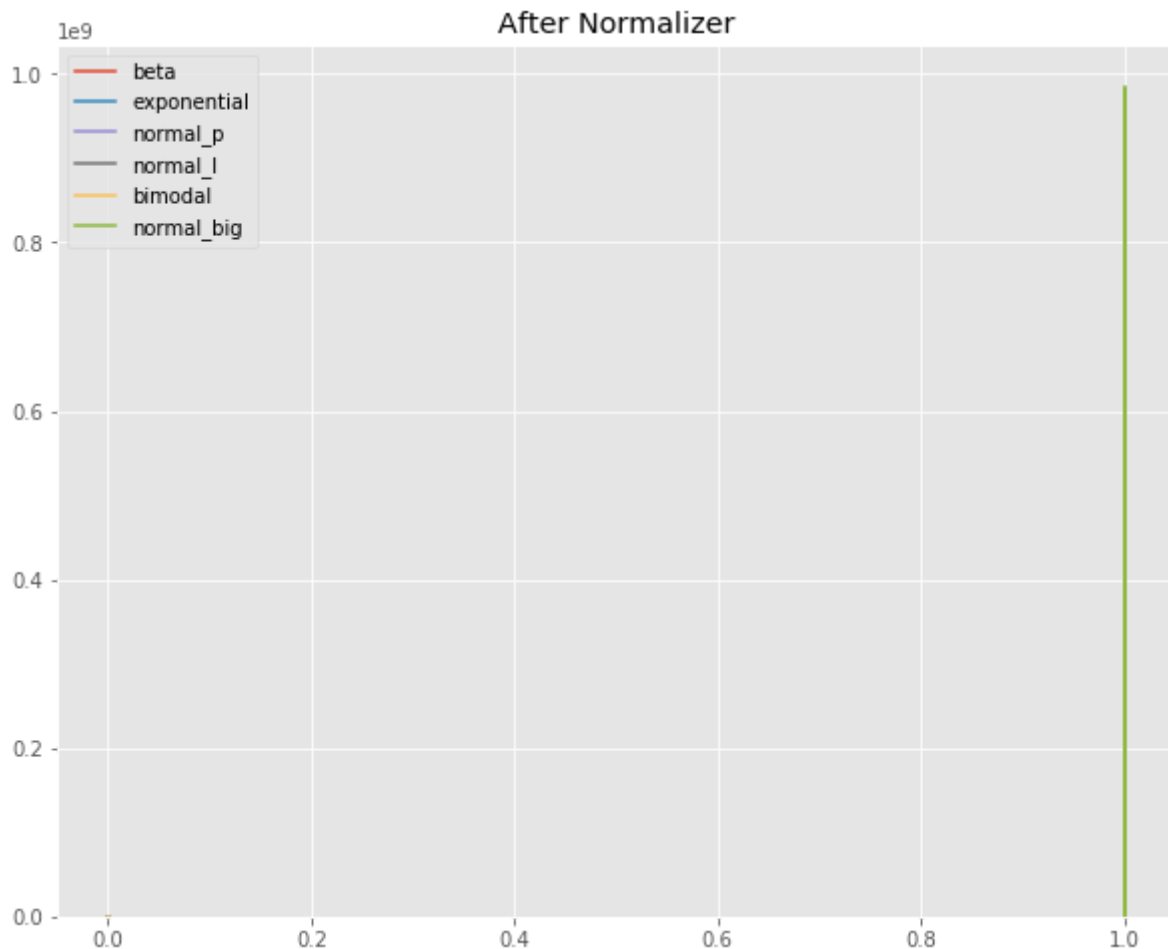
Now let's have a look at Normalizer.



We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

By default, L2 normalization is applied to each observation so that the values in a row have a unit norm. *Unit norm* with L2 means that if each element were squared and summed, the total would equal 1. Alternatively, L1 (aka taxicab or Manhattan) normalization can be applied instead of L2 normalization.



Normalizer does transform all the features to values between -1 and 1 (*this text updated July 2019*). In our example, *normal\_big* ends up with all its values transformed to .9999999.

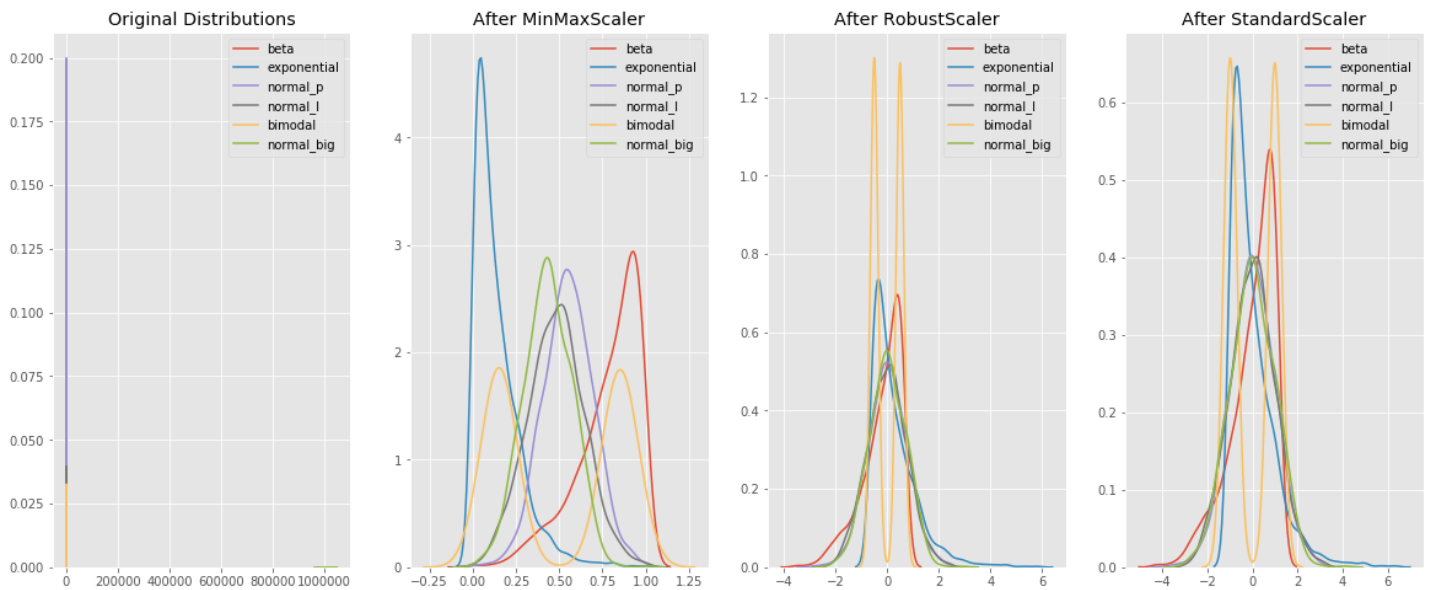
Have you found good use cases for Normalizer? If so, please let me know on Twitter @discover.

In most cases one of the other preprocessing tools above will be more helpful.

Again, scikit-learn's Normalizer works on the rows, not the columns.

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



Note that after any of these three transformations the values are on a similar scale.

Also notice that MinMaxScaler doesn't distort the distances between the values in each feature.

## Wrap

- Use MinMaxScaler as the default if you are transforming a feature. It's non-distorting.
- You could use RobustScaler if you have outliers and want to reduce their influence. However, you might be better off removing the outliers, instead.
- Use StandardScaler if you need a relatively normal distribution.
- Use Normalizer sparingly — it normalizes sample rows, not feature columns. It can use l2 or l1 normalization.

Here's a cheat sheet I made in a Google Sheet to help you keep the options straight:

Preprocessing Type	Scikit-learn Function	Range	Mean	Distribution Characteristics	When Use	Definition	Notes
Scale	MinMaxScaler	0 to 1 default, can override	varies	Bounded	Use first unless have theoretical reason to need stronger medicine.	Add or subtract a constant. Then multiply or divide by another constant. MinMaxScaler subtracts the minimum value in the column and then divides by the difference between the original maximum and original minimum.	Preserves the shape of the original distribution. Doesn't reduce the importance of outliers. Least disruptive to the information in the original data. Default range for MinMaxScaler is 0 to 1.
Standardize	RobustScaler	varies	varies	Unbounded	Use if have outliers and don't want them to have much influence.	RobustScaler standardizes a feature by removing the median and dividing each feature by the interquartile range.	Outliers have less influence than with MinMaxScaler. Range is larger than MinMaxScaler or StandardScaler.
Standardize	StandardScaler	varies	0	Unbounded,	When need to transform a feature	StandardScaler standardizes a feature by removing the	Results in a distribution with a standard deviation equal to 1

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

In this article you've seen how scikit-learn can help you scale, standardize, and normalize your data.

Resources to go deeper:

- Here's a scikit-learn doc on preprocessing data.
- Here's another doc about the effects of scikit-learn scalers on outliers.
- Here's a nice guide to probability distributions by Sean Owen.
- Here's another guide comparing these functions by Ben Alex Keen.

I hope you found this guide helpful. If you did, please share it on your favorite social media channel. 🙌

I write about Python, Docker, SQL, pandas, and other data science topics. If any of those topics interest you, read more here and follow me on Medium. 😊

Join my **Data Awesome** mailing list. One email per month of awesome curated content!

Email Address



We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

Scale on!

## Sign up for The Daily Pick

By Towards Data Science

Hands-on real-world examples, research, tutorials, and cutting-edge techniques delivered Monday to Thursday. Make learning your daily ritual. [Take a look](#)

Get this newsletter

Create a free Medium account to get The Daily Pick in your inbox.

Thanks to Kathleen Hale.

[Machine Learning](#)

[Data Science](#)

[Python](#)

[Towards Data Science](#)

[Statistics](#)

[About](#) [Help](#) [Legal](#)

Get the Medium app

