Feature Transformation

As a budding Data Scientist, I've been experimenting with Machine Learning models. As many Data Scientists will tell you, there is a general process for preparing your data for Machine Learning. Part of this process includes data transformations like scaling and/or standardization. Very generally, Machine Learning models may perform better when feature distributions are *approximately normal* and when feature *scales are similar*. A disparity in scale can slow down, or even prevent convergence for gradient-based models. This disparity also becomes a major issue for algorithms that use Euclidean distance between two data points in their computations.

After reading and watching many tutorials, I've grasped that this transformation step is important, but I didn't have much intuition on when it's appropriate to use which transformation. So, I decided to do some research and exploration into what several popular transformations actually do. This post is about my thoughts and findings after visualizing the effect of different scalers on some data. Scikit-Learn has great documentation on how their scalers handle outliers, so I used the documentation as an outline for my own exploration.

The Lingo

First and foremost, I think it's important that we're all clear on the terminology. In my opinion, the key to a clear explanation is consistency with, and a common understanding of, the lingo used in the explanation. Luckily, Jeff Hale agrees with me, so I'll use his definitions.

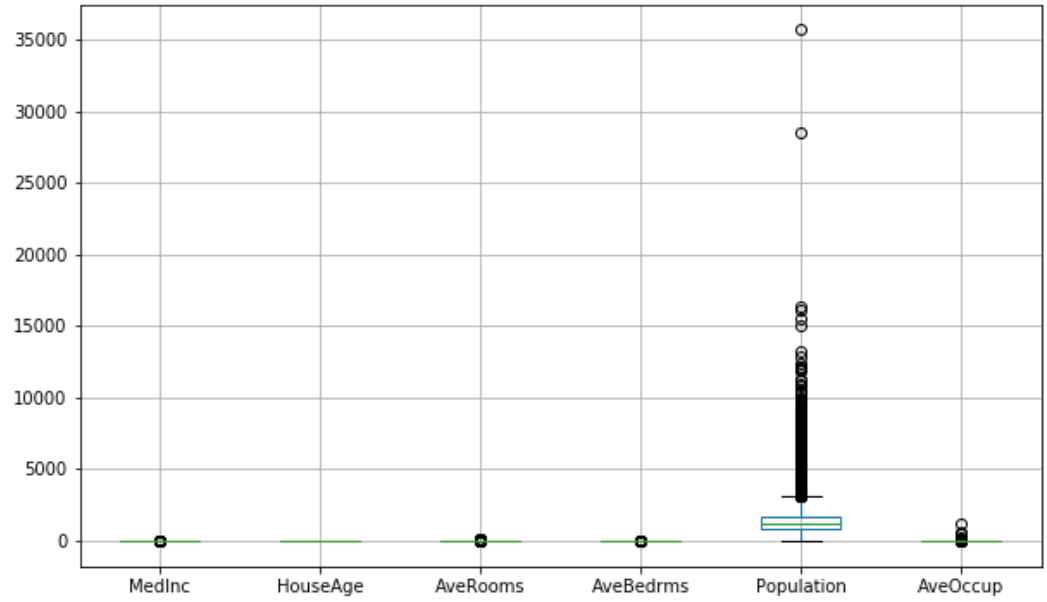
Scale — To change the scale of a dataset means changing **range of values** of the dataset. For example, it's possible to scale a range of ages [21–75] down to a range of [0–1]. Generally, changing the scale (or scaling) won't change the shape of the data's distribution.

Standardize — Standardizing generally means changing the data's values so that the standard deviation of the data = 1 (called **unit variance**). This often goes hand-in-hand with removing the distribution's mean (setting mean = 0). This tends to shift the shape of the data towards the shape of a normal distribution. Data is often scaled implicitly when standardized.

Normalize — apparently normalize is a more ambiguous term that can be used in place of either of the two terms above. Since I'm attempting to clear up ambiguity and confusion, I'm going to avoid this term for now.

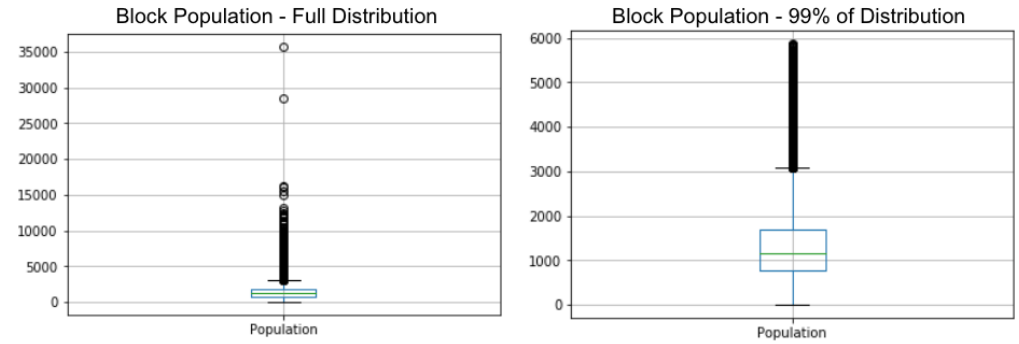
Playing with Transformers

In order to get a better grasp of this topic, I used the California Housing data accessible through Scikit-Learn.



Population is on a much bigger scale than the rest of the features

For quick reference, this box-plot shows how *different the scales are* for the features in this dataset. The Population scale is so much larger than the other features, that the other features barely even show up in this visualization.

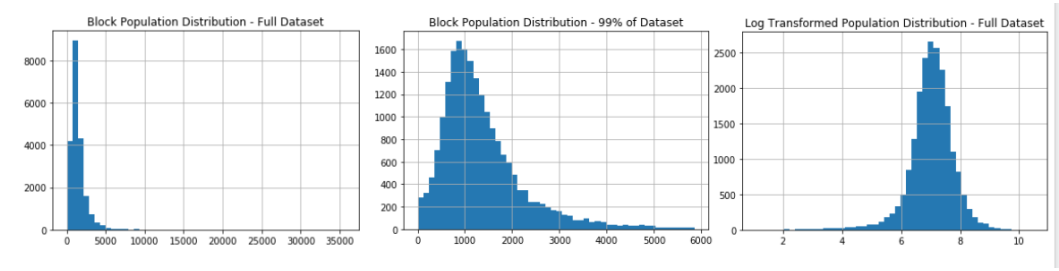


This dataset is also interesting because it has huge *outliers*. While there are times when it's best to remove outliers, there may be times where outliers actually contain valuable information. So I left these outliers in to see how data transformations handle outliers.

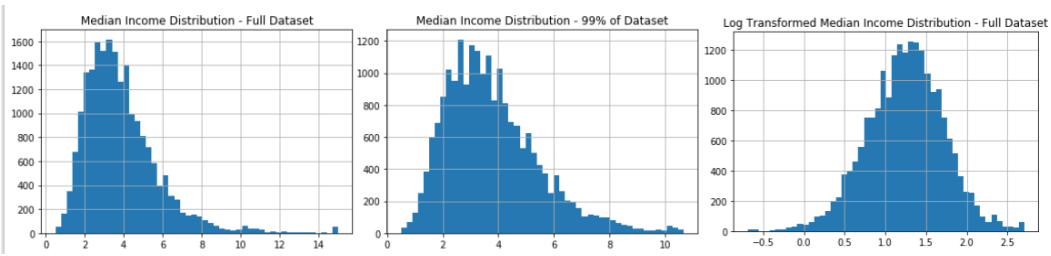
Log Transformations

A natural log is the inverse of the constant e. The constant e is the base rate of growth for all continuous processes, and the natural log gives you the time needed to reach a certain level of growth. For a better understanding of e and natural logs, I've found this article very helpful.

Log transformations tend to be used most often on skewed distributions.



Notice the scale went from 0–35K to 0–10.



Here, the scale decreased and the distribution is less skewed.

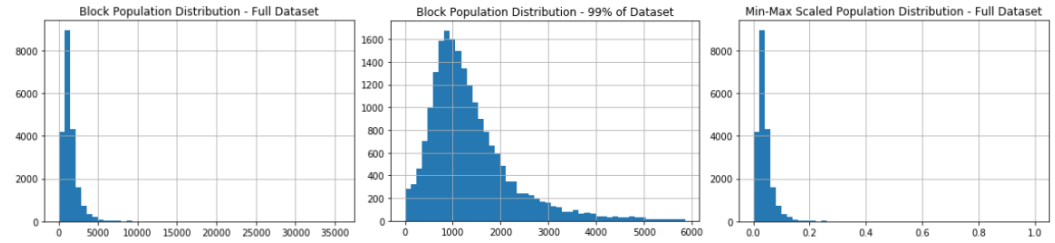
It seems that using a log transformation decreased the scale of the distributions, even with the huge range of the average population. It seems the outliers caused the log-transformed distributions to still be a bit skewed, but it is closer to normal than the original distribution.

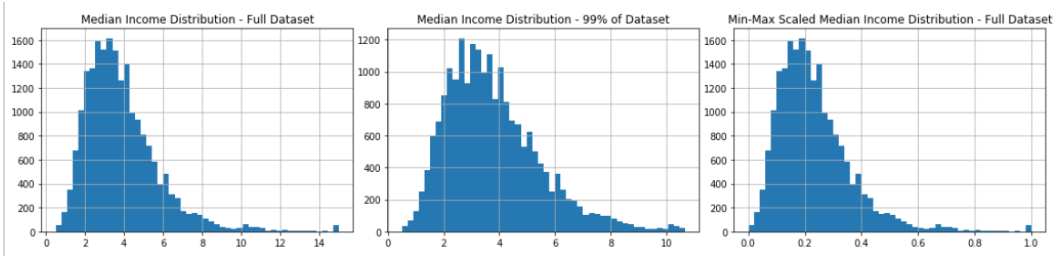
The pitfalls of log transformations are that they only work on non-zero and non-negative data, and doesn't transform to a predetermined scale (i.e. always [0–1]).

Min-Max Scaler

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

Min-Max Scaler rescales the data to a predefined range, typically 0–1, using the formula shown to the left.





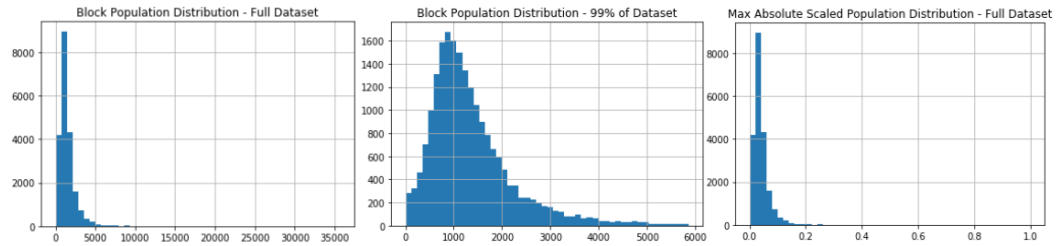
Here we can see a Min-Max scaler doesn't reduce the skewness of a distribution. It simply shifts the distribution to a smaller scale [0–1]. For this reason, it seems Min-Max scaler isn't the best choice for a distribution with outliers or severe skewness.

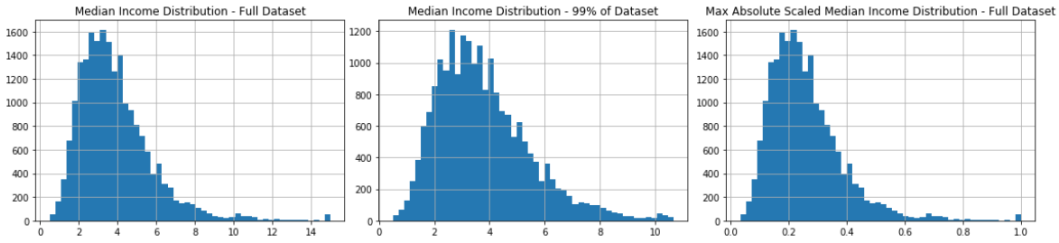
Max Absolute Scaler

The Scikit-Learn documentation actually has a pretty clear description of how the Max Abs Scaler works:

> *This estimator scales and translates each feature individually such that the maximal absolute value of each feature in the training set will be 1.0. It does not shift/center the data, and thus does not destroy any sparsity.*

So the Max Abs scaler scales data to a range of [-1–1], and doesn't change the shape of the distribution. This was reinforced by the histograms below. Since the data was all non-negative, the Max Abs scaler acted similarly the the Min-Max scaler. The distributions were shifted to the scale of [0–1], but the center of the data didn't shift.



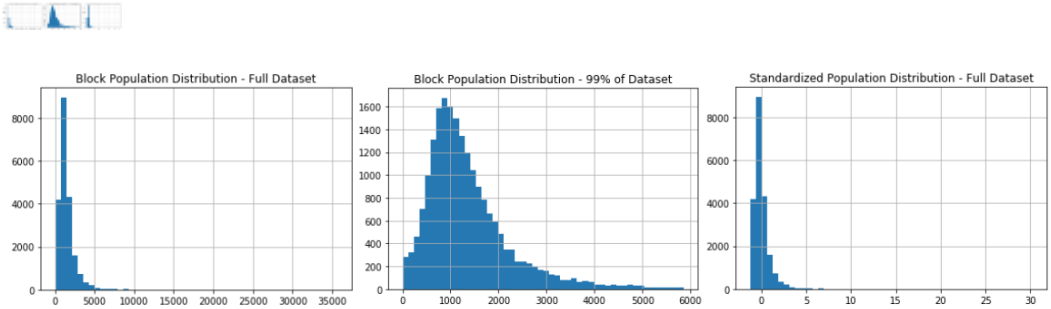


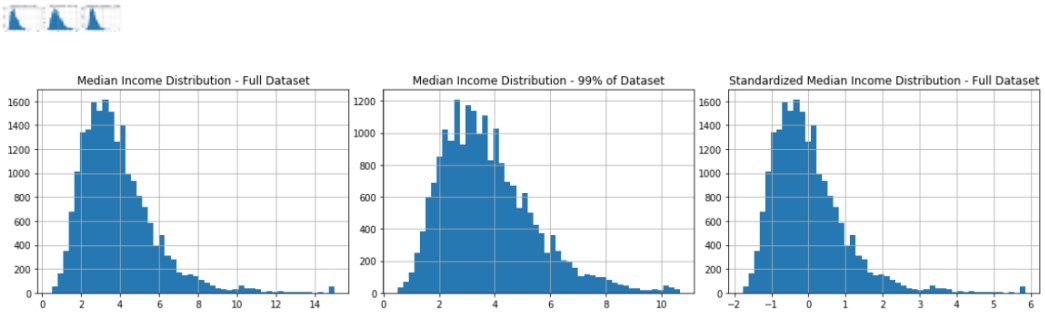
Standard Scaler

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

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

The standard scaler assumes features are normally distributed and will scale them to have a mean 0 and standard deviation of 1. Unlike Min-Max or Max-Abs scalers, the Standard scaler doesn't have a predetermined range to scale to.





We can see that using the Standard scaler on these distributions shifted the mean to 0, but didn't really change the center (skewness) of the data. This is because the outliers in this dataset have an impact on the feature mean and standard deviation used to scale the features. It's also interesting to note that since population had such huge outliers, the scaled population distribution still had a relatively large scale after transformation [0–30] compared to some of the other scalers.

Robust Scaler

Scikit-Learn boasts that the Robust Scaler better handles outliers. According to documentation,

$$\frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)}$$

> *This Scaler removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).*

If I'm being honest, the Robust scaled data looks almost exactly like the Standard scaled data. I think this is due to the amount and the size of the outliers in this data. I'm curious, too, if changing the default quartiles would change the results.

Power Transformer

The *PowerTransformer* from Scikit-Learn, aims to correct skewness in distributions. There are two options for Power transformation: Yeo-Johnson transform and Box-Cox transform. However, Box-Cox can only be applied to strictly positive data. For more details about the transformation calculations, see the documentation.

It looks like both the Yeo-Johnson and Box-Cox method standardize the data to resemble a normal distribution. However, these methods don't scale the data to a predetermined range.

Conclusion

I think my biggest takeaway from this exercise is how important it is to visualize your data before and after transformations. While we can understand generally what each scaler is *supposed* to do, each dataset is different; and things like outliers can make a difference in how these scalers perform. I think selecting a scaler and/or standardizer also depends on which algorithm you plan to use, since some algorithms have specific requirements (i.e. needing a range of 0–1).

TL;DR

Log Transformer

- Helps with skewness

- No predetermined range for scaled data
- Useful only on non-zero, non-negative data

Min-Max Scaler

- Rescales to predetermined range [0–1]
- Doesn’t change distribution’s center (doesn’t correct skewness)
- Sensitive to outliers

Max Abs Scaler

- Rescales to predetermined range [-1–1]
- Doesn’t change distribution’s center
- Sensitive to outliers

Standard Scaler

- Shifts distribution’s mean to 0 & unit variance
- No predetermined range
- Best to use on data that is approximately normally distributed

Robust Scaler

- 0 mean & unit variance
- Use of quartile ranges makes this less sensitive to (a few) outliers
- No predetermined range

Power Transformer

- Helps correct skewness
- 0 mean & unit variance
- No predetermined range
- Yeo-Johnson or Box-Cox
- Box-Cox can only be used on non-negative data

Let’s Talk!

If you’re a practicing Data Scientists or Statistician with any tips or notes, I’d love to learn from you! If you’re another Data Science student with different interpretations, I’d love to hear your thoughts! Feel free to send me a private note or leave a comment!
Samantha Jackson

Resources