Here we’ll see impact of standardization, let’s say before standardization our data is like given in below image.

The range for f1 feature is 0-100 and for f2 is 0-1.

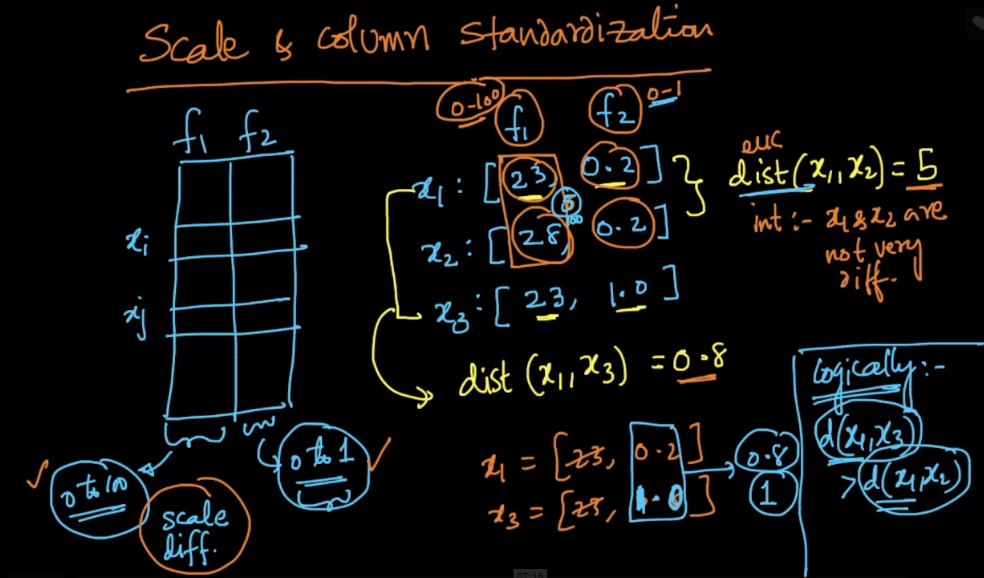
Now Euclidean dist(x1, x2) is 5, here difference is coming from f1, since f2 becomes 0, therefore it’s 5% of highest value of range 0-100.

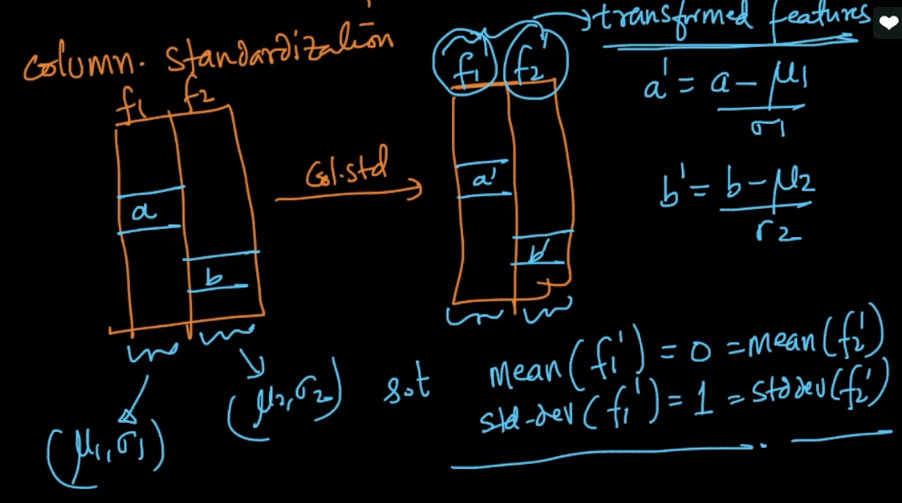
Dist(x1, x3) = 0.8, here difference is coming from f2, since f1 becomes 0, therefore it’s 80% of highest value of range 0-1.

Therefore for our calculation d(x1, x2) > d(x1, x3).

But logically d(x1, x3) > d(x1, x2). Because d(x1, x2) has only 5% of range, but d(x1, x3) has 80% of range.

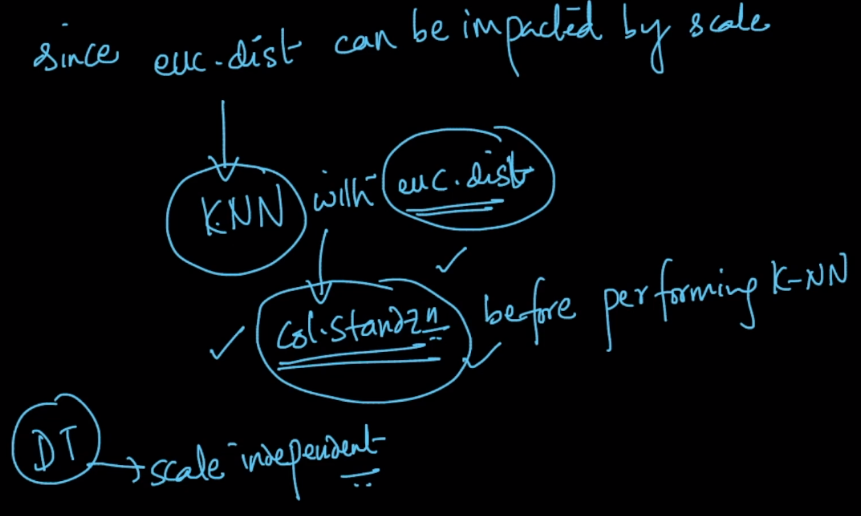
And this problem is because both features are in different scale, so we bring both feature in same scale using standardization.





Since we just saw that Euclidean distance can impacted by scale, and therefore if we use kNN with eculidean distance then we must column standardize before perfoming k-NN.

There are several algorithms like decision tree which doesn’t require columns to be standardized.



**Comments:**

Which is better normalization or standardization and why?

There's nothing like one of them is better and the other is not. In case if our problem requires all the features to contain values only in the interval [0,1], then we should prefer normalization. Whereas if our problem requires all the features to have same mean and variance, then we should prefer standardization. In ML algorithms, most of the algorithms give better performance when the data is standardized.

**Why, how and when to do scaling:** [**https://medium.com/greyatom/why-how-and-when-to-scale-your-features-4b30ab09db5e**](https://medium.com/greyatom/why-how-and-when-to-scale-your-features-4b30ab09db5e)