Normalization

Standardization: (Z-score Scaling)

It is a pre processing technique used in ML to transform a feature's values so that they have a mean of 0 and standard deviation of 1. This process makes features comparable and can improve the performance of various algorithms.

Why do we need scaling?

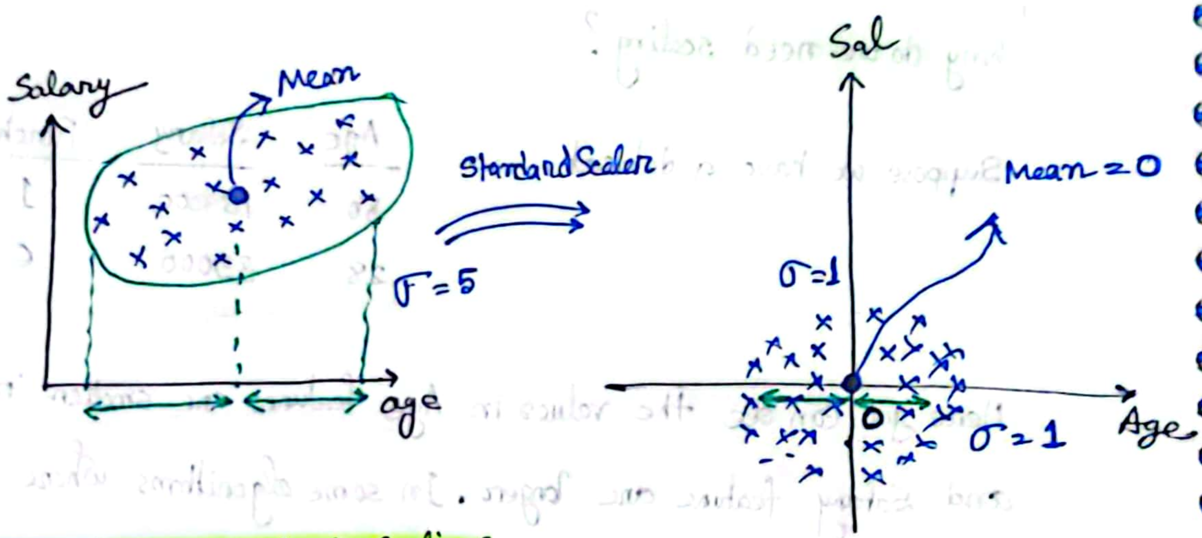
Suppose we have a dataset →

Age	Salary	Purchase
50	183000	1
28	39000	0

Here you can see the values in Age feature are smaller in number and Salary feature are larger. In some algorithms where Euclidean distance between the values are calculated, in those cases the distance results of salary feature will be really big numbers compared to age feature. Some algorithms are structured like that, where ~~we~~ the values of the independent features should be in similar range. If some feature has small ranged values and some has large ranged values, the ML model will not perform good in those scenarios. That's why we need feature scaling (Standardization) ~~to~~ so that our ML algorithms can perform well.

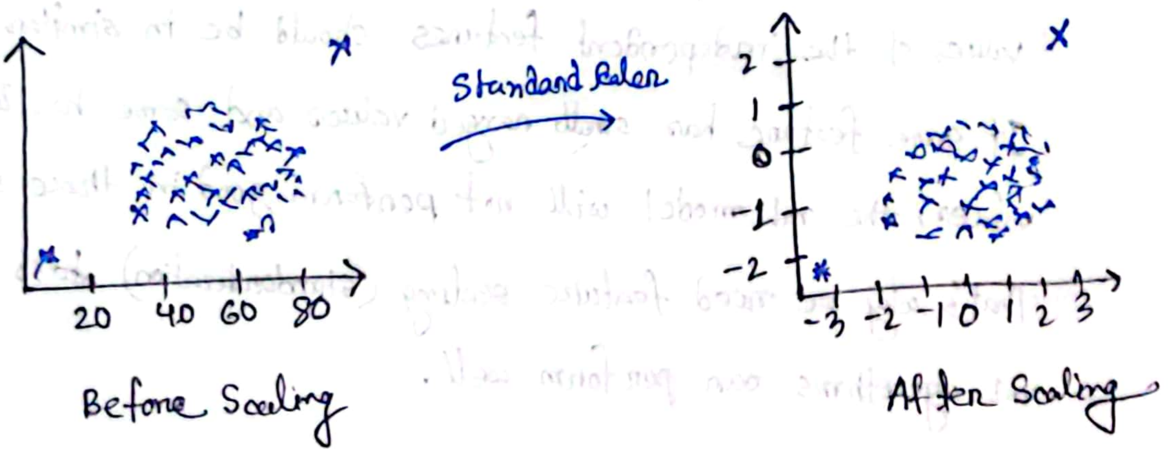
Standardization Formula → $x_i' = \frac{x_i - \bar{x}}{\sigma}$ → (Mean) → (std deviation)

Age		Scaled Age
27	Standardization (Standard Scaler) →	$\mu = 0$
15		$\sigma = 1$
23		
33		
63		
90		
7		



Outlier's Impact on Standard Scaling:

Outliers don't change after scaling. They behave the same.



When to use Standardization?

Standardization don't do any harm to the model performance even if you don't need it.

But in some algorithms, standardization helps the performance to be better.

- Classification
- K-Nearest Neighbours
- PCA
- Neural Network
- Gradient Descent (LR, Linear Regression, Logistic Regression)
- K-Means

Where we don't need them:

- Decision Tree
- Boosting Algorithms (Xgboost)

Normalization Techniques:

The goal of normalization is to change the values of numerical columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

Some popular normalization teqhniques →

- MinMax Scaling (used the most)
- Mean normalization Scaling
- Max absolute Scaling
- Robust Scaling

MinMaxScaling:

Weight Feature

130
67
81
61
32
51
⋮
100 values

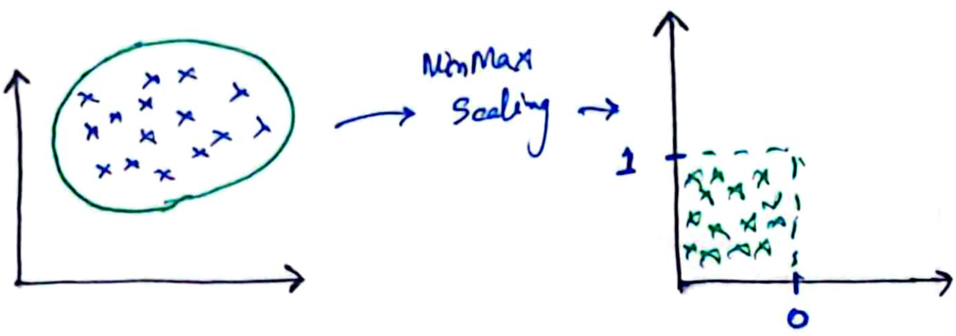
Normalize with MinMax Scaling

↓

$$x_i' = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Scaled weight

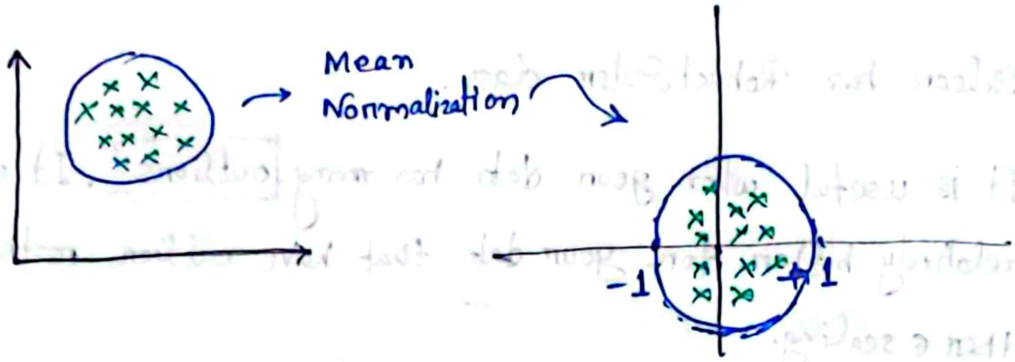
[0 – 1]



Mean Normalization :

Mean centering

Formula → $x_i' = \dfrac{\boxed{x_i - x_{mean}}}{x_{max} - x_{min}}$



This is another technique but rarely used but scikit learn has no class for this.

Max Abs Scaling :

Formula → $x_i' = \dfrac{x_i}{|x_{max}|}$

Scikit learn has MaxAbsScaler class for this.

It is useful for Sparse Data where there is more number of 0's.

Robust Scaling:

Formula → $$x_i' = \frac{x_i - x_{median}}{IQR \{75th\ percentile - 25th\ percentile)}$$

Sklearn has RobustScaler class.

It is useful when your data has many $\boxed{\text{outliers}}$. It works relatively better for your data that have outlier rather than the other scaling.

Normalization Vs Standardization:

→ First, check that if feature scaling is actually needed.

→ (Because suppose for Decision Tree feature scaling is not needed)

→ Use MinMaxScaling → (when you know min and max value of your data)

→ Use StandardScaler → When you have no idea about the data

→ Use MaxAbsScaler → When you have a sparse data

→ Use RobustScaler → When your data has outliers.