Data Normalization

Normalization is used to scale the data of an attribute so that it falls in a smaller range, such as -1.0 to 1.0 or 0.0 to 1.0. It is generally useful for classification algorithms.

Need of Normalization –

Normalization is generally required when we are dealing with attributes on a different scale, otherwise, it may lead to a dilution in effectiveness of an important equally important attribute(on lower scale) because of other attribute having values on larger scale.

In simple words, when multiple attributes are there but attributes have values on different scales, this may lead to poor data models while performing data mining operations. So they are normalized to bring all the attributes on the same scale.

person_name	Salary	Year_of_ experience	Expected Position Level
Aman	100000	10	2
Abhinav	78000	7	4
Ashutosh	32000	5	8
Dishi	55000	6	7
Abhishek	92000	8	3
Avantika	120000	15	1
Ayushi	65750	7	5

The attributes salary and year_of_experience are on different scale and hence attribute salary can take high priority over attribute year_of_experience in the model.

Methods of Data Normalization –

- Decimal Scaling
- Min-Max Normalization
- z-Score Normalization(zero-mean Normalization)

Decimal Scaling Method For Normalization –

It normalizes by moving the decimal point of values of the data. To normalize the data by this technique, we divide each value of the data by the maximum absolute value of data. The data value, v_i, of data is normalized to v_i' by using the formula below –

$$v_i' = \frac{v_i}{10^j}$$

where *j* is the smallest integer such that max(|v_i'|)<1.

Example –

Let the input data is: -10, 201, 301, -401, 501, 601, 701

To normalize the above data,
Step 1: *Maximum absolute value in given data(m): 701*
Step 2: *Divide the given data by 1000 (i.e j=3)*

Result: *The normalized data is: -0.01, 0.201, 0.301, -0.401, 0.501, 0.601, 0.701*

Min-Max Normalization –

In this technique of data normalization, linear transformation is performed on the original data. Minimum and maximum value from data is fetched and each value is replaced according to the following formula.

$$\mathbf{v' = \frac{v - min(A)}{max(A) - min(A)}(new\_max(A) - new\_min(A)) + new\_min(A)}$$

Where A is the attribute data,
Min(A), Max(A) are the minimum and maximum absolute value of A respectively.
v' is the new value of each entry in data.
v is the old value of each entry in data.

new_max(A), new_min(A) is the max and min value of the range(i.e boundary value of range required) respectively.

```
from sklearn.preprocessing import MinMaxScaler
MiMaScaler = MinMaxScaler()
MiMaScaler.fit(x)
x_test = np.array([[4.563,4,6,2],
[4.999,3,3,1]]) #假设新的数据集，可当作我们平时用到的测试集
MiMaScaler.transform(x_test) #如果新数据集中最大最小值超出原来fit的数据的最大最小值，请注意transform出来的数据将不在[0,1]内，需重新fit
```

Z-score normalization –

In this technique, values are normalized based on mean and standard deviation of the data A. The formula used is:

$$v' = \frac{v - \overline{A}}{\sigma_A}$$

v', v is the new and old of each entry in data respectively. σ_A, $\overline{A}$ is the standard deviation and mean of A respectively.

```
from sklearn.preprocessing import scale
iris_scale = scale(x)
print(iris_scale[:5]) #显示4个被scale后的特征
```

绝对值标准化（暂时这么叫）（将数据缩放到一定范围内，专为稀疏数据而生）
使用原因：
将每个要素缩放到[-1,1]范围，它不会移动/居中数据，因此不会破坏任何稀疏性。
该估计器单独地缩放每个特征，使得训练集中的每个特征的最大绝对值将是1.0。
该缩放器也可以应用于稀疏CSR或CSC矩阵。

```
from sklearn.preprocessing import MaxAbsScaler
MaAbScaler = MaxAbsScaler().fit(x)
MaAbScaler.transform(x)[:10]
```

鲁棒性标准化（暂时这么叫）（将数据缩放到一定范围内，专为异常值而生）
使用原因：

标准差标准化（第一种，最常用的那种）对数据中出现的异常值处理能力不佳，因此诞生了
robust_scale□这种不怕异常值扰动的数据缩放法。
此Scaler根据分位数范围（默认为IQR Interquartile Range)删除中位数并缩放数据。
IQR是第1四分位数（第25个分位数）和第3个四分位数（第75个分位数）之间的范围。

```
from sklearn.preprocessing import RobustScaler
Robust_Scaler = RobustScaler()
Robust_Scaler.fit(x)
Robust_Scaler.transform(x)[:10]
```