Feature Scaling → Normalisation

Normalisation → technique often → As a part of → Data preparation → for Machine learning

goal is → To change a value of numerical Columns → make it one scale → without distorting difference → or losing information range

Type of Normalisation

- Min Max Scaling
- Mean normalisation
- Max Absolute scaling
- Robust Scaling

There are more Scaling function in Scikkarn

↳ Most important is → Min Max Scaling

Min Max Scaling :⇒

Weight
130
67
81
61
32
54

Normalize
Min Max Scaling

$$X_i = \frac{X_i - X_{min}}{X_{max} - X_{min}}$$

$X_i = 130$

① $= \frac{130 - 32}{130 - 32} = 1$

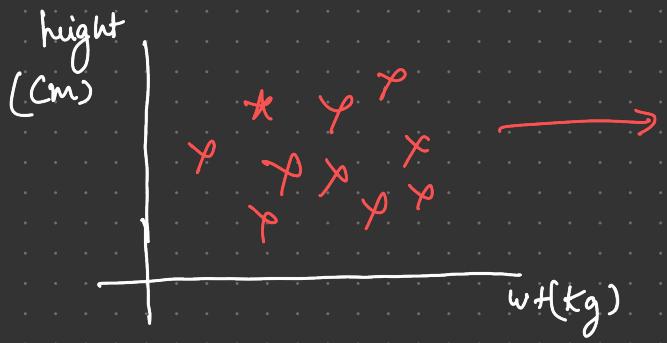
② $= \frac{67 - 32}{130 - 32} =$

By this we can see that Range will be [0, 1]

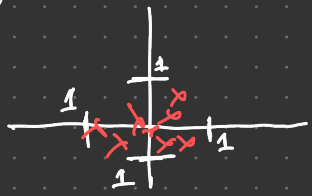
$\frac{130 - 32}{130 - 32}$ ① max

$\frac{32 - 32}{130 - 32} = 0$ min

Geomatrical Way



Applying min max Scaling



We are fusing all value

[0,1]

Code form

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

✓ 1.8s

```
df = pd.read_csv('wine_data.csv',header=None,usecols=[0,1,2])
df.columns=['ClassLabel', 'Alcohol', 'MalicAcid']
```

✓ 0.0s

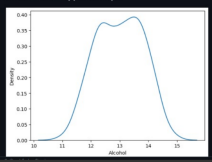
- The `pd.read_csv()` function is a Pandas function that is used to read CSV files into DataFrames.
- The `header=None` parameter tells Pandas to ignore the first row of the CSV file. This is useful if the first row of the CSV file does not contain column names.
- The `usecols=[0,1,2]` parameter tells Pandas to only read the first three columns of the CSV file. This is useful if you only need to use a subset of the columns in the CSV file.
- The `DataFrame` is a Pandas data structure that is used to store tabular data. It is similar to a spreadsheet.
- The `columns` attribute of a DataFrame is a list of the names of the columns in the DataFrame.
- The `rename()` method of a Series or DataFrame is used to rename the columns or index labels.

Data Visualization

Kdeplot is a visualization technique used to estimate and plot the probability density function (PDF) of a data set

```
sns.kdeplot(df['Alcohol'])
```

✓ 0.6s



In Data Visualization → KDE Plot (Seaborn library)

↓

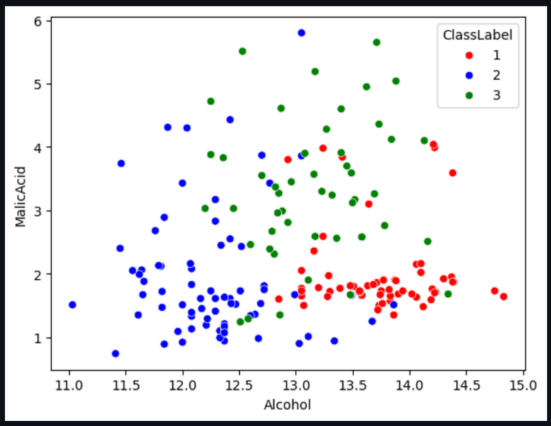
Help to plot → PDF (Probability density function)

↳ this Doesnt Make any assumption about the
↳ Underlying distruction of the Data

By this Versatile tool for → Exploring Data
↳ Especially Data → is not Normaly Distributed

```
color_dist={1:'red',3:'green',2:'blue'}
sns.scatterplot(x=df['Alcohol'], y=df['MalicAcid'], hue = df['ClassLabel'], palette=color_dist)
```
✓ 0.3s

```
<Axes: xlabel='Alcohol', ylabel='MalicAcid'>
```



this is for Data Preprocessing

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(df.drop('ClassLabel',axis=1),
                                                 df['ClassLabel'],test_size=0.3, random_state=0)
```
✓ 0.0s

→ Splitting Data to test and train

```
X_train.shape,X_test.shape
```
✓ 0.0s

```
((124, 2), (54, 2))
```

From Min Max Scaler we transformed

- When ever we fit it will get fit in training Data

transform train ko bhi karega test ko bhi karega

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
#fit the scaler to the train set, it will learn the parameters
scaler.fit(X_train)

#transform train and test sets
X_train_Scaled = scaler.transform(X_train)
X_test_Scaled = scaler.transform(X_test)
```

```
X_train_Scaled = pd.DataFrame(X_train_Scaled, columns= X_train.columns)
X_test_scaled = pd.DataFrame(X_test_Scaled, columns=X_test.columns)
```
[19] ✓ 0.0s

So when we use Scikit learn library it will change it into numpy Array

To handle that we converted as Dataframe.

```
np.round(X_train.describe(),1)
```
✓ 0.1s

	Alcohol	MalicAcid
count	124.0	124.0
mean	13.0	2.4
std	0.8	1.1
min	11.0	0.9
25%	12.4	1.6
50%	13.0	1.9
75%	13.6	3.2
max	14.8	5.6

```
np.round(X_train_Scaled.describe(),1)
```
✓ 0.0s

	Alcohol	MalicAcid
count	124.0	124.0
mean	0.5	0.3
std	0.2	0.2
min	0.0	0.0
25%	0.4	0.2
50%	0.5	0.2
75%	0.7	0.5
max	1.0	1.0

After Doing min max Value

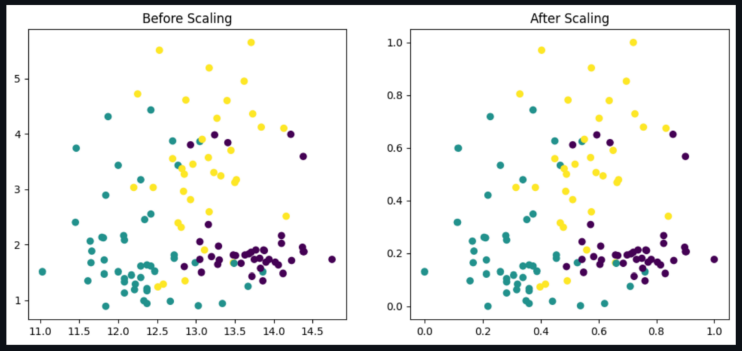
It will get zero

Mean → Standard deviation Behave differently

Mostly min and max work like that

```
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(12, 5))

ax1.scatter(X_train['Alcohol'],X_train['MalicAcid'],c=Y_train)
ax1.set_title("Before Scaling")
ax2.scatter(X_train_Scaled['Alcohol'],X_train_Scaled['MalicAcid'],c=Y_train)
ax2.set_title("After Scaling")
plt.show()
```
[27] ✓ 0.4s Python



May be in some distribution mai thoda sa change a sakta hai

But its ignored

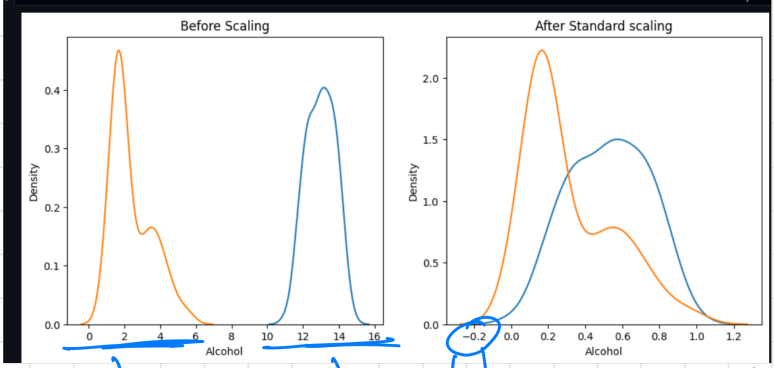
Because Data ka jo form hota hai usko Retain kar sakta hai mean max scalar.

```
fig, (ax1,ax2) = plt.subplots(ncols= 2, figsize=(12,5))
#Before Scaling
ax1.set_title('Before Scaling')
sns.kdeplot(X_train['Alcohol'], ax=ax1)
sns.kdeplot(X_train['MalicAcid'], ax=ax1)

#After Scaling
ax2.set_title("After Standard scaling")
sns.kdeplot(X_train_Scaled['Alcohol'], ax= ax2)
sns.kdeplot(X_train_Scaled['MalicAcid'], ax=ax2)

plt.show()
```

Python



Mean Alcohol was in the Range 0-6

MalicAcid → 10-16

After Doing mean max Scaling it came in same Range.

There is some sort going Negative

As we are doing KDE plot

It take reference Statically form

↳ There is a chance of changing → Shape after Distribution

↓

After Distribution → that isn't neccessory it should maintain same shape

↳ we have issue in Outlier → Outlier bhi Squuz hojata hai

In min max Scaling

Mean Normalization :->

↳ $$X_i' = \frac{X_i - X_{mean}}{X_{max} - X_{mean}}$$ → Here we mean Centoric doing it

↳ (-1 to 1) Range is Value

Mean < Value → +ve number

Mean > Value → -ve number

↳ It Rarely used → there isn't any Class in Scikit-learn

↳ It only useful in → Centered data

↓

Even though People use Standardization

Max Abs Scaling :→

Max Absolute Scaling →

$$X_i' = \frac{X_i}{|X_{max}|}$$

We have a Class in Scikit-learn → MaxAbsScaler

↓

We can use directly

It get utilized where → we have

Sparse data → "0's" Bhaut hai

Robust Scaling :→

$$X_i' = \frac{X_i - X_{median}}{IQR \{ 75^{th} \text{ Percentile} \} - \{ \text{Value} \}}$$

It is (Robust) → Because → If Data have lot of Outlier

↳ we cant tell Before which will do good work with Which Data.

By Seeing → tabular Data → we cant say

Normalization Vs Standardization

1) is feature Scaling needed ? → Decession Tree not needed

For this we need to understand Algo.

2) Mostly Problem will solved doing → Standardization

that is why Standard Scalar Class most using

↓

Most of the Time better result

④ Normalization → Min Max Scaler

↓

When we are aware of Min and Max for Example

CNN ← Image processing

↓

Pixel have

0 - 255 Range Min Max Scaling

↳ Outlier hai → Robust Scaling

↳ Don't Know → Standard Scaling

↳ Min Max → Min max Scaling

↳ Sparse Values ? → MaxAbs Scaling