5/4/22
11:00 AM

"2" ⇒ Feature Scaling ⟹ Applicable Only For Continous Data

Feature Scaling ?

Ans:- it refers (or) Techniques used to normalize The range of independent Variables in our data, (or)
input Variables

* The methods to set The Features Value range within a Similar Scale.
same scale

* Variables with bigger magnitude / Larger Value range dominate over those with Smaller magnitude / value range.

Ex:- 10,000,00 , 10
So we Take 10 —— 10 → Both are Equal Import

* Scale of The features is an important consideration when building Machine Learning models

* Feature Scaling is generally the last step in the data pre processing pipe line, Performed Just Before Training the machine learning algorithms.

5. Gradient descent converges Faster when Features are on Similar scales

* Feature Scaling *

Example :- of a car

C.C	Milage
1600	14
1800	15
2200	16.5
2100	18
2600	22.5
1800	19.4
1900	25.4

Continous Data

Q. When we ask Machine which is important In The both columns ?

* Machine is Giving Importance To C.C Because it is Having Large Values.
* But, We know both are importa[nt]

So, we have train model Initially in such that, it should consider "Both The columns Similar"

So, we Reduce The Value

$= \frac{800}{1000} = \frac{80}{100} = \frac{8}{10} = \frac{4}{5}$ (Every one is Same)

with 100 — with 200

We are scaling down Dividing Every value with "10" In this case.

* To Reduce The Value.

* Feature Scaling Importance in Some ML Algorithms

1. Linear Regression & logistic Regression
   * The regression coefficients of linear models are Directly influenced by Scale of The Variable.

2. Support Vector Machines :-
   * Feature Scaling helps Decrease The Time to Find For SVM's Support Vectors

3. K-means clustering
   * Euclidean distances are Sensitive to Feature magnitude

4. Principal component analysis (PCA) :-
   * PCA requires The Features to be Centered at "0".

* Various Feature Scaling Techniques :-

1. standardisation
2. Normalisation Technique
   - → Scaling to minimum and Maximum values [Min Max Scaling]
   - → Scaling to Maximum value [Max Abs Scaling]
   - → scaling to quantiles and Median [Robust Scaling]

```
# import pandas as pd
# import matplotlib.pyplot as plt
  %matplotlib inline.
```

→ using (titanic Data Set)

code:

```
# titanic = pd.read_csv ("titanic.csv", usecols =["Age"
  titanic .head ()
```

Out

Age
22.0
38.0
26.0
35.0
35.0

Check Null Values in "Age" column.

```
# titanic.isnull().sum()
```

Out Age 177 → Null values

Replacing with "Median Value" For Null values

```
# titanic["Age"].fillna(titanic["Age"].median(),
                          inplace = True)

# titanic["Age"].isnull().sum()
```

Out : Age 0 ⟶ Zero Null Values

I * Standardisation

↓

Converting Each Value To Zscore

Ex :-

X	$\frac{x-\mu}{\sigma}$ = Zscore	
23	$\frac{23-60}{2}$	= -18.5
45	$\frac{45-60}{2}$	= -7.5
67	$\frac{67-60}{2}$	= 3.5
89	$\frac{89-60}{2}$	= 14.5
18	$\frac{78-60}{2}$	= 9

These Five values converting To Zscores is called as "Standard Scaling"

Avg/mean = 60.

(Std) σ = 2
Ex

Code :-

From sklearn.Preprocessing import Standard Scaler (sc)

```
# sc = Standard Scaler ()   # call The Function (Short cut)
```

main (don't Forget)

```
# titanic ["Age_sc"] = sc. fittranform (titanic [["Age"]]
```

#creating new column

#fit_transForm → Calculation → Converting The value

stores in

```
# titanic ["Age_sc"]
```

Out :-

	Age_sc
0	- 0.5657
1	0.6638
2	-0. 2583
3	0.43312
	⋮
890	0.20272

Here Every Value is Converted To Zscore.

$$\left[\frac{X-\mu}{\sigma}\right] = Z \text{ score}.$$

Ex :- Age [22] # First record X=22

#titanic.mean() $\mu = 29.361$

#titanic.std() $\sigma = 13.01$

$$\left[\frac{X-\mu}{\sigma}\right] = \left[\frac{22-29.361}{13.01}\right]$$

⇓

"Age_sc" → [-0.5657]

```
# titanic
```

Out

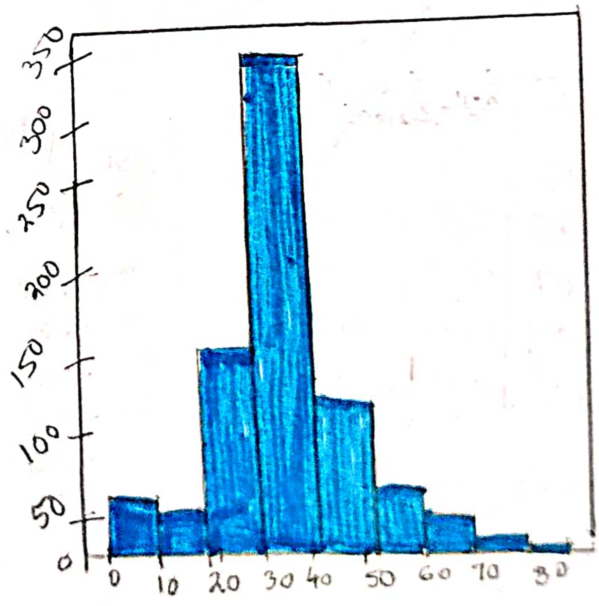
Original Data

Converted To Z_Score

	Age	Age_sc
0	22.0	-0.5657
1	38.0	-0.6638
2	25.0	-0.2583
	⋮	
889	26.0	-0.2583
890	32.0	0.20272

histogram of Original ["Age"]

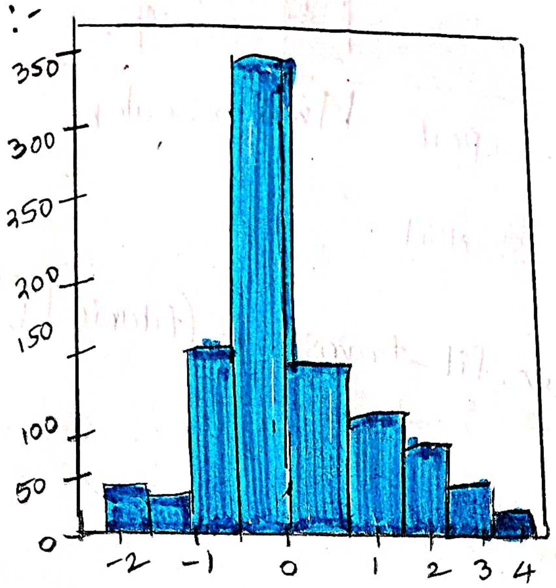
```
# titanic ["Age"]. hist()
plt. show
```



Histogram of Converted "Age"

```
# titanic["Age_sc"].hist()
# plt.show()
```

Out :-



Here, we see, Histogram is not going to change, But, Values does.

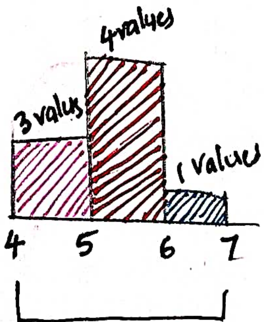
⇒ Example :-

* How ? Histogram Remains Same, But Changed Values

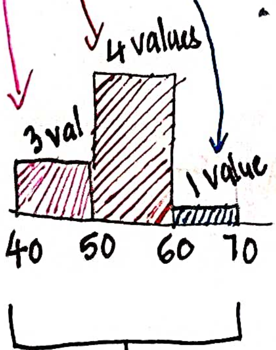
Ans :-

Ex :-

X	Changed X
40	4
50	5
42	4.2
43	4.3
54	5.4
58	5.8
59	5.9
62	6.2



Here only "X" values Only changes But Histogram Remains Same.



II # Normalisation

Ex:-

X	
65	$\frac{65-34}{53}$
45	$\frac{45-34}{53}$
34	$\frac{34-34}{53} = 0$
67	$\frac{67-34}{53}$
67	
87	$\frac{87-34}{53} = 1$

min. value ← 34

Max. value ← 87

$$\boxed{\frac{X - X_{min}}{X_{max} - X_{min}}}$$

* MinMax Scaling

→ MinMax Scaling Scales, The Values between $\boxed{\text{"0" to "1"}}$ (Every thing is (+ve) value No -ve value)

Code: From sklearn . Preprocessing import MinMaxScaler

```
# min_max = MinMaxScaler() # Shortcut
# titanic ["Age_mm"] = min_max.fit_transform (titanic [["Age"]])
```

titanic ["Age_mm"] → creating new column

fit_transform

```
# titanic
```

(calling / show Data)

Out :-

	Age	Age_SC	Age_mm
0	22.0	-0.5657	0.271
1	38.0	0.6637	0.472
2	26.0	-0.258	0.32
⋮	⋮		
889	26.0	-0.25	0.32
890	32.0	0.20	0.39

* Robust Scaling

$$\boxed{\frac{X - X_{median}}{IQR}}$$

$\therefore IQR = Q_3 - Q_1$

code :

From sklearn.preprocessing import RobustScaler

rs = RobustScaler() → shorcut()

titanic ["Age_rs] = rs.fit_transform(titanic [["Age"]])

titanic ["Age_rs] (or) = $\underbrace{\text{RobustScaler()}}_{\text{If not using any shortcut like rs = RobustScaler()}}$. fit_transform (. . . .)

titanic

Out

Age	Age_sc	Age_Mm	Age_rs
22.0	0.565	0.461	0.4615
38.0	0.66	0.769	0.769
26.0	-0.25	-0.153	-0.153
⋮			
26.0	-0.25	-0.153	-0.153
32.0	0.2	0.396	0.3076

* **Max Abs Scaling**

$$\boxed{\frac{X}{X_{max}}}$$

Ex:-

X	
68	68/80
44	44/80
52	52/80
69	69/80
80	80/80 = 1

Max. value

code

From sklearn.preprocessing Import MaxAbsScaler

mas = MaxAbsScaler()

titanic ["Age_mas"] = mas.fit_transform (titanic [["Age"]]

titanic

Out :-

Original Variable
Standard Scaler
Min max scaler
Robust scaler
Max Abs Scaler

	Age	Age-sc	Age-mm	Age-rs	Age-mas
0	.	.	.	.	0.2750
1	.	.	.	.	0.4950
2	.	.	.	.	0.3250
					⋮
889	.	.	.	.	0.3250
890	.	.	.	.	0.4000

891 x 5 col

5/04/22
4:58 pm.

Example :- Discretization

Continous Data

12.5
22.8
25.2
18.3
12.4
15.8
23.3

⇒ $\boxed{\text{Milage of a Car}}$ (Discrete . categorical) Data

<15 → low . milage
15-20 → Avg . milage
20-25 → good. milage
>25 → Very good

→ By our own choice To Divide them into parts