Standardization and Normalization

- Standardization and Normalization are two methods to convert data values into same scale
- It is also called scaling of data
- In the data we have many columns are there, each column has different units as well as different values
- For example you have age and salary , age is very minimal number two digit number
- And salary is kind of 5 digit number
- When you multiply 2 digit number with 5 digit number it involves some complexity
- Imagine you are multiplying both are single digit number , this involves less complexity
- Scaling converts all the data into a same scale
- **Standardization**:
  - It is also called Z-score or Z-scale
  - It ranges -3 to 3
  - The mean =0 and std=1

$$Z = \frac{x - \mu}{\sigma}$$

- **Normalization**
  - Min max scalar
  - Normalization converts data into 0 to 1 range
  - min value =0 and max value =1
  - It mainly use in Deep learning for the image scaling
  - Generally images are color images the pixel value ranges from 0 to 255
  - We Normalize the values into 0 to 1
  - The value might be change but information never change

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Import the packages

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the data

In [2]:
```
file_location="C:\\Users\\omkar\\OneDrive\\Documents\\Data science\\Naresh 
visa_df=pd.read_csv(file_location)
visa_df.head()
```

Out[2]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_
0	EZYV01	Asia	High School	N	N	
1	EZYV02	Asia	Master's	Y	N	
2	EZYV03	Asia	Bachelor's	N	Y	
3	EZYV04	Asia	Bachelor's	N	N	
4	EZYV05	Africa	Master's	Y	N	

Z-score

- step-1: we read a specific column (Prevailing_Wage)
- step-2: calculate mean of the column
- step-3: calculate std of the column
- step-4: Nr= column-mean
- step-5: Nr/std

In [8]:
```
mean=visa_df['prevailing_wage'].mean()
std=visa_df['prevailing_wage'].std()
Nr=visa_df['prevailing_wage']-mean
out=Nr/std
visa_df['prevailing_wage_Zscore']=out
```

In [10]:
```
visa_df[['prevailing_wage','prevailing_wage_Zscore']]
```

Out[10]:

	prevailing_wage	prevailing_wage_Zscore
0	592.2029	-1.398510
1	83425.6500	0.169832
2	122996.8600	0.919060
3	83434.0300	0.169991
4	149907.3900	1.428576
...	...	...
25475	77092.5700	0.049923
25476	279174.7900	3.876083
25477	146298.8500	1.360253
25478	86154.7700	0.221504
25479	70876.9100	-0.067762

25480 rows × 2 columns

In [13]:
```
max_original=visa_df['prevailing_wage'].max()
max_z=visa_df['prevailing_wage_Zscore'].max()
max_original,max_z
```

Out[13]:
```
(319210.27, 4.634101837909902)
```

In [14]:
```
visa_df['prevailing_wage'].idxmax()
# prevailing_wage  column has maximum value at 21077 ID
```

Out[14]:
```
21077
```

In [15]:
```
visa_df['prevailing_wage_Zscore'].idxmax()
# prevailing_wage_Zscore  column has maximum value at 21077 ID
```

Out[15]:
```
21077
```

In [17]:
```
visa_df.iloc[[21077]]
```

Out[17]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
21077	EZYV21078	Asia	High School	N	

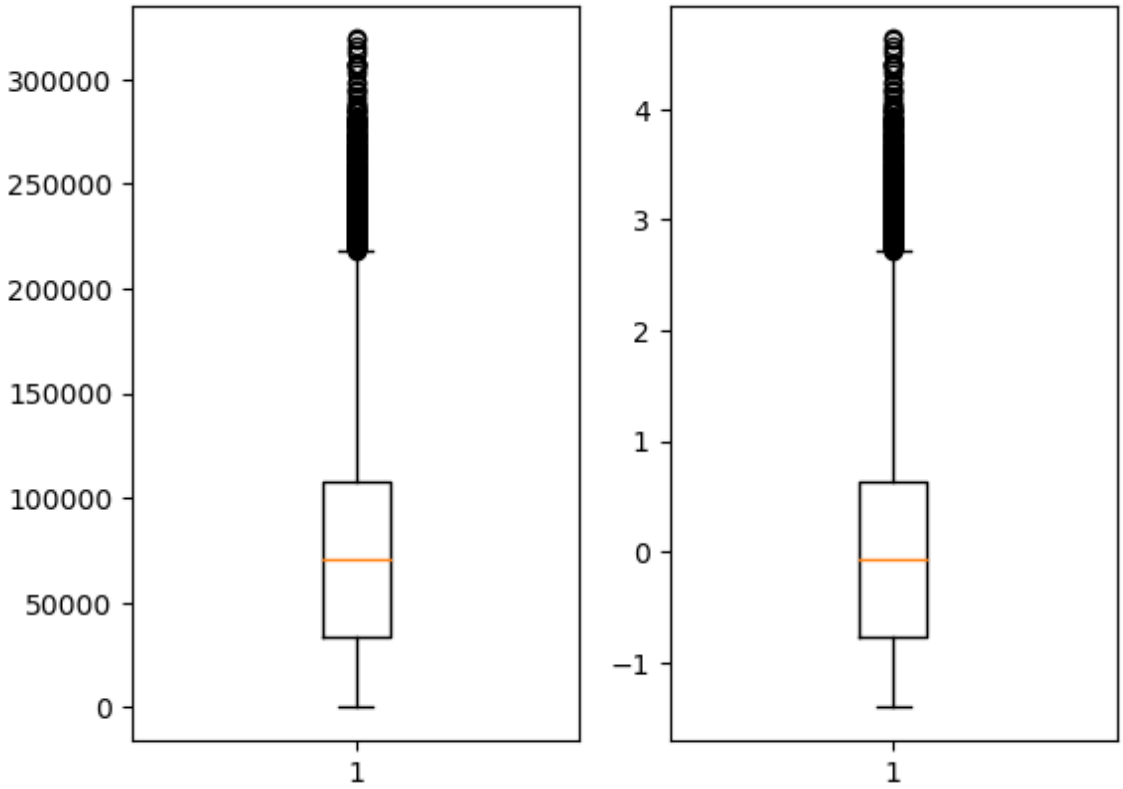
In [18]:
```
min_original=visa_df['prevailing_wage'].min()
min_z=visa_df['prevailing_wage_Zscore'].min()
print(min_original,min_z)
print(visa_df['prevailing_wage'].idxmin())
print(visa_df['prevailing_wage_Zscore'].idxmin())
```

```
2.1367 -1.4096818992891214
20575
20575
```

Influential outliers

- Generaly Outlier means very very huge observation, very very small observation
- If we found an observation as outlier before scaling
- The same observation again found as outlier after scaling
- Then that observation called influential outlier
- Some observation before scaling consider as outlier
- But after scaling it does not fall in outlier creteria
- At that time we might not consider that observation as outlier
- Z-score is used to identify influential outliers

In [21]:
```
plt.subplot(1,2,1)
plt.boxplot(visa_df['prevailing_wage'])
plt.subplot(1,2,2)
plt.boxplot(visa_df['prevailing_wage_Zscore'])
plt.show()
```



In [30]:
```
**Using package: StandardScalar**
```

```
  Cell In[30], line 1
    **Using package: StandardScalar**
    ^
SyntaxError: invalid syntax
```

In [24]:
```
######################### Read the data again#############
file_location="C:\\Users\\omkar\\OneDrive\\Documents\\Data science\\Naresh 
visa_df=pd.read_csv(file_location)
# instead of LabelEncoder use StandardScalar
# Read the package
# Save the pacakge
# Apply fit transform

from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
visa_df['prevailing_wage_Zscore']=ss.fit_transform(visa_df[['prevailing_wag

visa_df[['prevailing_wage','prevailing_wage_Zscore']]
```

Out[24]:

	prevailing_wage	prevailing_wage_Zscore
0	592.2029	-1.398537
1	83425.6500	0.169835
2	122996.8600	0.919079
3	83434.0300	0.169994
4	149907.3900	1.428604
...	...	...
25475	77092.5700	0.049924
25476	279174.7900	3.876159
25477	146298.8500	1.360280
25478	86154.7700	0.221509
25479	70876.9100	-0.067763

25480 rows × 2 columns

Normalization

- Read the again
- Step-1: Read the column
- Step-2: Calculate min value
- Step-3: Calculate max value
- Step-4: Nr= column-min
- Step-5: Dr= Max-min
- Step-6: out= Nr/Dr
- Step-7: Save in a new column

In [27]:
```
file_location="C:\\Users\\omkar\\OneDrive\\Documents\\Data science\\Naresh I
visa_df=pd.read_csv(file_location)
min_val=visa_df['prevailing_wage'].min()
max_val=visa_df['prevailing_wage'].max()
Nr=visa_df['prevailing_wage']-min_val
Dr=max_val-min_val
out=Nr/Dr
visa_df['prevailing_wage_norm']=out
visa_df[['prevailing_wage','prevailing_wage_norm']]
```

Out[27]:

	prevailing_wage	prevailing_wage_norm
0	592.2029	0.001849
1	83425.6500	0.261345
2	122996.8600	0.385312
3	83434.0300	0.261371
4	149907.3900	0.469616
...	...	...
25475	77092.5700	0.241505
25476	279174.7900	0.874579
25477	146298.8500	0.458311
25478	86154.7700	0.269895
25479	70876.9100	0.222033

25480 rows × 2 columns

In [29]:
```
visa_df.iloc[[20575,21077]]
```

Out[29]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
20575	EZYV20576	North America	Master's	N	
21077	EZYV21078	Asia	High School	N	

Using package: MinMaxScalar

In [31]:
```
# package name= MinMaxScalar
######################### Read the data again#############
file_location="C:\\Users\\omkar\\OneDrive\\Documents\\Data science\\Naresh 
visa_df=pd.read_csv(file_location)

from sklearn.preprocessing import MinMaxScaler
mms=MinMaxScaler()
visa_df['prevailing_wage_norm']=mms.fit_transform(visa_df[['prevailing_wage

visa_df[['prevailing_wage','prevailing_wage_norm']]
```

Out[31]:

	prevailing_wage	prevailing_wage_norm
0	592.2029	0.001849
1	83425.6500	0.261345
2	122996.8600	0.385312
3	83434.0300	0.261371
4	149907.3900	0.469616
...	...	...
25475	77092.5700	0.241505
25476	279174.7900	0.874579
25477	146298.8500	0.458311
25478	86154.7700	0.269895
25479	70876.9100	0.222033

25480 rows × 2 columns

fit_transform

- There are two terms fit and transform
- If you see in z-score
  - x: data
  - mean: mean of data
  - std: std of data
- Will find the value of mean and std, this is called fit
- Once we find the values we need apply on entire data, this is called transform
- fit transform calculate the measurements(parameters or statistic) and apply on data

In [ ]: