Transformation of Features

Feature Transformation is a technique we should always use regardless of the model we are using, whether it is a classification task or regression task, or be it an unsupervised learning model.

What is Feature Transformation?

It is a technique by which we can boost our model performance. Feature transformation is a mathematical transformation in which we apply a mathematical formula to a particular column(feature) and transform the values which are useful for our further analysis, also called Feature Engineering.This transforamtion techniques can also be used for Feature Reduction.

Why Transformation of Features Are Required?

1. Improves model performance
2. Improve numerical stability, training process and model interpretability
3. Promotes robustness of model

Some examples of algorithms where feature scaling matters are:

- k-nearest neighbors with an Euclidean distance measure if want all features to contribute equally
- k-means (see k-nearest neighbors)
- logistic regression, SVMs, perceptrons, neural networks etc. if you are using gradient descent/ascent-based optimization, otherwise some weights will update much faster than others
- linear discriminant analysis, principal component analysis, kernel principal component analysis since you want to find directions of maximizing the variance (under the constraints that those directions/eigenvectors/principal components are orthogonal); you want to have features on the same scale since you'd emphasize variables on "larger measurement scales" more.

There are many more cases.

Types Of Transformation

1. Normalization And Standardization
2. Scaling to Minimum And Maximum values
3. Scaling To Median And Quantiles
4. Guassian Transformation
5. Logarithmic Transformation
6. Reciprocal Trnasformation
7. Square Root Transformation
8. Exponential Trnasformation
9. Box Cox Transformation

In [1]:
```
import pandas as pd
import seaborn as sns
```

Preparing a DataFrame to try out All Features Transformation

In [2]:
```
titanic = sns.load_dataset('titanic')
titanic.head()
```

Out[2]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN

In [3]:
```
titanic=titanic[['pclass','age','fare','survived']]
titanic.head()
```

Out[3]:

	pclass	age	fare	survived
0	3	22.0	7.2500	0
1	1	38.0	71.2833	1
2	3	26.0	7.9250	1
3	1	35.0	53.1000	1
4	3	35.0	8.0500	0

In [4]:
```
titanic.age.isnull().sum()
```

Out[4]:
```
177
```

In [5]:
```
titanic['age'].fillna(titanic.age.median(),inplace=True)
```

In [6]:
```
titanic.isnull().sum()
```

Out[6]:
```
pclass      0
age         0
fare        0
survived    0
dtype: int64
```

In [7]:
```
### Independent and dependent features
X=titanic.iloc[:,0:-1]
y=titanic.iloc[:,-1]
```

In [8]:
```
y
```

Out[8]:
```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: survived, Length: 891, dtype: int64
```

In [9]:
```
X
```

Out[9]:

	pclass	age	fare
0	3	22.0	7.2500
1	1	38.0	71.2833
2	3	26.0	7.9250
3	1	35.0	53.1000
4	3	35.0	8.0500
...	...	...	...
886	2	27.0	13.0000
887	1	19.0	30.0000
888	3	28.0	23.4500
889	1	26.0	30.0000
890	3	32.0	7.7500

891 rows × 3 columns

In [10]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state
```

Standard Scaler:

- StandardScaler follows Standard Normal Distribution (SND). Therefore, it makes mean = 0 and scales the data to unit variance.
- The StandardScaler assumes your data is normally distributed within each feature and will scale them such that the distribution is now centred around 0, with a standard deviation of 1.
- If data is not normally distributed, this is not the best scaler to use.
- The mean and standard deviation are calculated for the feature and then the feature is scaled based on:

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

μ = Mean

σ = Standard Deviation

In [11]:
```
#### standarisation: We use the Standardscaler from sklearn library
from sklearn.preprocessing import StandardScaler
```

In [12]:
```
scaler=StandardScaler()
### fit vs fit_transform
X_train_scaled=scaler.fit_transform(X_train)
```

In [13]:
```
X_train_scaled
```

Out[13]:
```
array([[-1.62580285,  1.91349292,  0.38784185],
       [ 0.80576177, -0.09452019, -0.31908053],
       [-0.41002054, -0.32621401,  0.18792449],
       ...,
       [ 0.80576177,  0.90948636, -0.34613654],
       [-1.62580285, -1.17575802,  1.71250117],
       [-1.62580285, -0.63513911,  0.8821286 ]])
```

In [14]:
```
X_test_scaled=scaler.transform(X_test)
X_test_scaled
```

Out[14]:
```
array([[ 0.80576177, -0.09452019, -0.32402243],
       [-0.41002054,  0.13717363, -0.4162854 ],
       [ 0.80576177, -0.71237038, -0.46634591],
       [-0.41002054, -1.79360821,  0.0211365 ],
       [ 0.80576177, -1.17575802, -0.40186603],
       [-1.62580285, -0.24898274,  0.91250512],
       [ 0.80576177, -0.09452019, -0.46974808],
       [ 0.80576177, -1.02129548, -0.2704781 ],
       [ 0.80576177, -1.02129548, -0.46974808],
       [-1.62580285, -0.78960166, -0.10944269],
       [-1.62580285,  0.60056127,  0.41190006],
       [ 0.80576177,  1.14118018, -0.46391578],
       [ 0.80576177, -0.09452019, -0.12531819],
       [ 0.80576177,  0.05994235, -0.47995459],
       [-0.41002054,  0.52332999, -0.36768297],
       [-1.62580285, -1.02129548,  0.14555873],
       [-1.62580285,  0.98671764,  0.40128918],
       [ 0.80576177, -0.09452019, -0.46820835],
       [-0.41002054, -0.17175147, -0.36768297],
       [-1.62580285,  1.372874  ,  0.39051499],
       [ 0.80576177, -0.40344529, -0.46691358],
       [-1.62580285,  0.36886745, -0.10425778],
       [ 0.80576177, -0.78960166, -0.46772233],
       [ 0.80576177, -0.71237038, -0.44107264],
       [ 0.80576177, -0.09452019, -0.33852151],
       [ 0.80576177, -1.48468312, -0.07801247],
       [-1.62580285,  0.83225509, -0.08149629],
       [-0.41002054,  0.13717363, -0.36768297],
       [ 0.80576177, -1.94807076, -0.07801247],
       [ 0.80576177,  0.13717363, -0.46772233],
       [ 0.80576177, -0.78960166, -0.46691358],
       [ 0.80576177, -0.55790784, -0.46974808],
       [-1.62580285, -0.09452019, -0.02365939],
       [ 0.80576177, -0.09452019, -0.47363627],
       [ 0.80576177, -0.86683293, -0.46391578],
       [ 0.80576177, -0.17175147, -0.3394119 ],
       [-1.62580285, -0.09452019,  0.06973893],
       [ 0.80576177, -0.09452019, -0.46974808],
       [-1.62580285,  0.05994235, -0.01774545],
       [ 0.80576177, -0.09452019, -0.46691358],
       [-0.41002054, -0.63513911,  0.80849591],
       [ 0.80576177, -0.01728892, -0.48343841],
       [ 0.80576177, -0.09452019, -0.46391578],
       [ 0.80576177, -0.09452019, -0.46974808],
       [ 0.80576177,  1.21841146, -0.3394119 ],
       [ 0.80576177, -1.02129548,  0.04786783],
       [ 0.80576177, -0.78960166, -0.46181033],
       [ 0.80576177, -0.48067656, -0.44107264],
       [ 0.80576177, -0.40344529, -0.48335676],
       [-1.62580285,  2.22241801,  1.58176062],
       [ 0.80576177, -1.87083949, -0.24601552],
       [-1.62580285,  1.75903037,  1.19731538],
       [ 0.80576177,  0.83225509, -0.07801247],
       [-1.62580285, -1.40745185,  1.71250117],
       [ 0.80576177,  2.76303693, -0.46974808],
       [-1.62580285, -0.86683293,  0.9280579 ],
       [-0.41002054,  0.2144049 , -0.4162854 ],
       [-1.62580285,  1.60456783,  4.19162962],
       [-0.41002054,  0.44609872, -0.21215518],
       [ 0.80576177, -0.78960166, -0.4672363 ],
```

```
[ 0.80576177, -0.63513911, -0.46926205],
[-0.41002054, -1.2529893 , -0.24131664],
[-0.41002054, -0.09452019, -0.11495032],
[-1.62580285, -0.09452019, -0.11640839],
[ 0.80576177, -0.09452019, -0.46974808],
[-0.41002054,  2.14518674, -0.4162854 ],
[-0.41002054, -0.63513911,  0.80849591],
[ 0.80576177, -0.01728892, -0.46731796],
[ 0.80576177, -0.9440642 , -0.47987294],
[-1.62580285,  1.75903037,  0.9011652 ],
[-0.41002054, -0.24898274, -0.11495032],
[-1.62580285, -0.86683293,  4.48040973],
[-1.62580285, -0.24898274, -0.03718642],
[-1.62580285,  0.75502382,  0.99626266],
[ 0.80576177, -0.48067656, -0.46772233],
[ 0.80576177,  0.52332999, -0.31810849],
[ 0.80576177,  0.09855799, -0.46974808],
[-1.62580285, -0.55790784,  0.34191255],
[-0.41002054,  0.67779254, -0.36768297],
[ 0.80576177,  0.90948636, -0.22746495],
[ 0.80576177, -1.02129548,  0.15114801],
[-0.41002054, -0.01728892, -0.4162854 ],
[-1.62580285,  0.83225509,  1.99439528],
[ 0.80576177, -0.09452019, -0.46999109],
[-1.62580285,  2.06795547, -0.02365939],
[-1.62580285,  3.22642457,  0.34199421],
[-1.62580285,  0.13717363,  1.58176062],
[-1.62580285,  0.05994235,  0.48634343],
[-1.62580285,  0.67779254, -0.62041562],
[ 0.80576177, -0.09452019, -0.47946856],
[ 0.80576177, -1.56191439, -0.22138964],
[ 0.80576177, -1.56191439, -0.01239918],
[-1.62580285,  2.45411184,  0.00793219],
[ 0.80576177, -0.09452019, -0.47015245],
[ 0.80576177, -0.09452019, -0.46691358],
[ 0.80576177, -0.09452019, -0.46772233],
[-1.62580285,  1.45010528,  0.39051499],
[ 0.80576177,  1.21841146, -0.46391578],
[-0.41002054,  0.52332999, -0.36768297],
[ 0.80576177, -0.17175147, -0.46691358],
[-1.62580285,  2.68580566, -0.11495032],
[ 0.80576177,  0.44609872, -0.48189869],
[-1.62580285, -0.78960166,  1.15025267],
[ 0.80576177,  0.67779254, -0.48335676],
[ 0.80576177,  0.09855799, -0.46391578],
[ 0.80576177, -0.32621401, -0.46926205],
[-0.41002054,  0.83225509,  0.13778234],
[-1.62580285,  1.372874  ,  0.12806185],
[ 0.80576177,  0.05994235, -0.43572637],
[-1.62580285,  0.83225509, -0.01774545],
[-1.62580285, -0.09452019,  0.39051499],
[ 0.80576177, -0.67375475, -0.47946856],
[-1.62580285, -0.09452019,  1.11185675],
[-1.62580285,  0.52332999, -0.10936104],
[-1.62580285, -0.09452019,  3.69118787],
[ 0.80576177, -0.71237038, -0.42900369],
[-1.62580285,  3.22642457,  0.05329575],
[-0.41002054,  0.52332999, -0.37011309],
[-0.41002054, -0.55790784, -0.0566274 ],
[ 0.80576177,  1.45010528,  0.04786783],
```

```
[-0.41002054, -0.48067656, -0.39684443],
[-1.62580285,  0.2144049 ,  0.86276928],
[-1.62580285,  2.60857438,  0.8951696 ],
[-0.41002054, -2.1928939 , -0.0566274 ],
[ 0.80576177, -0.32621401, -0.47946856],
[-1.62580285,  0.44609872, -0.10425778],
[-1.62580285,  1.52733655, -0.11632674],
[-1.62580285, -0.40344529,  0.91930946],
[ 0.80576177, -0.09452019, -0.3393322 ],
[ 0.80576177, -0.63513911, -0.46391578],
[ 0.80576177, -0.09452019, -0.4672363 ],
[ 0.80576177, -0.09452019, -0.32402243],
[-0.41002054, -0.09452019, -0.62041562],
[ 0.80576177, -1.87083949, -0.37788948],
[-1.62580285,  1.52733655,  0.48634343],
[ 0.80576177, -1.09852675, -0.3394119 ],
[-1.62580285, -0.09452019,  1.53526365],
[ 0.80576177, -0.01728892, -0.46926205],
[ 0.80576177,  0.98671764, -0.47363627],
[ 0.80576177, -0.86683293, -0.2743663 ],
[ 0.80576177, -0.63513911, -0.45646406],
[-0.41002054,  1.14118018, -0.11495032],
[-0.41002054, -0.09452019, -0.36768297],
[-0.41002054,  0.60056127, -0.11495032],
[ 0.80576177,  0.75502382, -0.05419727],
[-0.41002054,  0.36886745,  0.01141601],
[-0.41002054,  0.05994235, -0.36768297],
[-0.41002054,  2.8402682 , -0.4162854 ],
[-1.62580285, -0.09452019,  2.22809133],
[ 0.80576177,  1.06394891, -0.46391578],
[ 0.80576177, -0.09452019, -0.45103614],
[ 0.80576177, -0.40344529, -0.47468997],
[ 0.80576177, -0.86683293, -0.46885768],
[ 0.80576177, -0.09452019, -0.15091612],
[ 0.80576177, -0.09452019, -0.46391578],
[ 0.80576177, -0.09452019, -0.48335676],
[ 0.80576177, -0.09452019,  0.47791772],
[ 0.80576177, -0.09452019, -0.46901904],
[-0.41002054,  0.52332999, -0.11495032],
[ 0.80576177,  0.2144049 , -0.31908053],
[ 0.80576177, -0.44206093, -0.47987294],
[-1.62580285, -0.09452019, -0.10425778],
[-0.41002054,  0.75502382, -0.36768297],
[-1.62580285, -0.09452019,  0.9770647 ],
[ 0.80576177, -0.09452019, -0.45597804],
[-1.62580285,  2.76303693,  0.58452035],
[-0.41002054, -0.48067656, -0.4162854 ],
[-1.62580285, -0.09452019,  0.38784185],
[ 0.80576177, -0.55790784, -0.46691358],
[ 0.80576177,  0.44609872, -0.48335676],
[-1.62580285, -0.48067656,  0.61133139],
[-0.41002054, -0.86683293, -0.36768297],
[-1.62580285,  0.52332999, -0.10741694],
[ 0.80576177, -1.56191439, -0.32402243],
[ 0.80576177, -0.9440642 , -0.48189869],
[ 0.80576177, -0.09452019, -0.47995459],
[ 0.80576177,  0.67779254, -0.01021207],
[-0.41002054, -0.9440642 , -0.4162854 ],
[ 0.80576177, -1.94807076, -0.29575137],
[-1.62580285,  1.25702709, -0.06634788],
```

```
[-0.41002054, -0.48067656, -0.36768297],
[ 0.80576177,  0.2144049 , -0.46634591],
[ 0.80576177, -0.24898274, -0.46772233],
[ 0.80576177, -1.79360821, -0.01239918],
[-1.62580285, -0.40344529,  4.19162962],
[-1.62580285,  1.21841146, -0.10425778],
[-0.41002054, -0.01728892, -0.08149629],
[ 0.80576177, -0.09452019, -0.46691358],
[-1.62580285, -0.09452019,  0.06973893],
[-0.41002054,  0.98671764, -0.36768297],
[ 0.80576177,  0.52332999, -0.15091612],
[-0.41002054,  0.29163617, -0.38177767],
[ 0.80576177, -0.9440642 , -0.48327511],
[ 0.80576177, -0.01728892, -0.43572637],
[-0.41002054,  1.60456783, -0.11495032],
[-1.62580285,  0.44609872,  1.12927197],
[-1.62580285,  0.67779254,  3.80289181],
[-0.41002054,  0.36886745, -0.36768297],
[-1.62580285, -0.9440642 ,  0.48771985],
[ 0.80576177, -1.40745185, -0.01239918],
[ 0.80576177,  2.45411184, -0.49915255],
[ 0.80576177,  0.05994235, -0.45200819],
[-0.41002054, -1.71637694, -0.11009007],
[ 0.80576177,  2.60857438, -0.43402529],
[ 0.80576177, -0.71237038, -0.47987294],
[ 0.80576177, -0.09452019, -0.18574851],
[ 0.80576177, -0.01728892, -0.43605104],
[-1.62580285,  0.52332999,  1.71250117],
[ 0.80576177, -0.09452019, -0.3393322 ],
[ 0.80576177,  1.60456783, -0.46391578],
[-1.62580285, -0.17175147,  3.49135022],
[ 0.80576177,  0.05994235, -0.47946856],
[-1.62580285,  0.29163617,  1.12927197],
[ 0.80576177, -0.09452019, -0.4702341 ],
[ 0.80576177, -0.09452019, -0.12531819],
[ 0.80576177, -2.10253331, -0.21069711],
[-0.41002054, -0.32621401, -0.03718642],
[-1.62580285,  1.6817991 ,  0.57285576],
[-0.41002054, -0.32621401, -0.11495032],
[ 0.80576177, -0.09452019, -0.47946856],
[ 0.80576177, -0.09452019, -0.46691358],
[-0.41002054, -0.40344529, -0.36768297],
[ 0.80576177, -0.86683293, -0.22746495],
[-1.62580285, -0.09452019, -0.02746594],
[ 0.80576177, -0.32621401, -0.48335676],
[-0.41002054, -0.40344529, -0.33852151],
[ 0.80576177, -0.55790784, -0.47420395],
[-1.62580285, -2.18594308,  2.32586387],
[-0.41002054, -0.40344529, -0.11495032],
[ 0.80576177, -0.24898274, -0.46634591],
[-0.41002054,  0.36886745, -0.21215518],
[-1.62580285, -0.63513911,  4.48040973],
[ 0.80576177, -0.09452019, -0.46974808],
[ 0.80576177, -0.09452019, -0.47987294],
[ 0.80576177, -0.09452019, -0.45200819],
[ 0.80576177, -0.55790784, -0.46926205],
[-1.62580285,  2.53134311,  0.93486224],
[ 0.80576177, -0.86683293, -0.42908339],
[-1.62580285,  0.98671764,  3.80289181],
[-0.41002054,  2.14518674, -0.3803196 ],
```

```
       [ 0.80576177, -0.78960166, -0.62041562],
       [-0.41002054,  0.98671764, -0.11495032],
       [ 0.80576177, -0.09452019, -0.47995459],
       [ 0.80576177, -0.71237038, -0.45200819],
       [ 0.80576177,  0.40748308, -0.49526435],
       [ 0.80576177, -0.09452019, -0.43572637],
       [ 0.80576177, -0.09452019, -0.46391578],
       [ 0.80576177, -0.48067656, -0.46691358],
       [ 0.80576177, -0.55790784, -0.47946856],
       [-1.62580285,  1.75903037, -0.02746594],
       [ 0.80576177,  0.05994235, -0.37788948],
       [-0.41002054,  0.56194563, -0.11495032],
       [ 0.80576177, -0.09452019, -0.46974808],
       [ 0.80576177, -0.55790784, -0.47987294],
       [-1.62580285,  1.60456783,  1.97787045],
       [ 0.80576177, -0.17175147, -0.48481483],
       [ 0.80576177, -0.55790784, -0.46391578],
       [-1.62580285,  1.6817991 ,  0.8951696 ],
       [-0.41002054,  0.52332999, -0.4162854 ],
       [-0.41002054, -0.01728892, -0.11495032],
       [-1.62580285, -0.55790784,  2.32586387],
       [-1.62580285,  1.60456783,  1.44858996],
       [-0.41002054,  0.83225509, -0.36768297],
       [ 0.80576177, -0.17175147, -0.45200819],
       [-1.62580285, -0.09452019, -0.62041562],
       [-1.62580285,  2.76303693, -0.10425778],
       [ 0.80576177, -0.9440642 , -0.45200819],
       [-1.62580285, -0.09452019, -0.10425778],
       [-1.62580285, -1.94807076,  0.9709894 ],
       [ 0.80576177, -0.09452019, -0.46691358],
       [ 0.80576177, -2.17976458, -0.40397343],
       [-0.41002054,  0.52332999, -0.08092861],
       [-0.41002054,  1.06394891, -0.11009007],
       [-1.62580285,  0.67779254,  2.36304473],
       [-0.41002054,  0.13717363, -0.11009007],
       [ 0.80576177, -0.01728892, -0.46391578],
       [ 0.80576177, -0.86683293, -0.45905554],
       [-0.41002054, -0.09452019, -0.32782897],
       [-1.62580285,  0.75502382,  1.53526365],
       [-0.41002054,  0.75502382, -0.36768297],
       [ 0.80576177, -0.24898274, -0.45200819],
       [ 0.80576177, -0.71237038, -0.48335676],
       [-1.62580285, -0.09452019,  1.97787045],
       [ 0.80576177,  1.52733655, -0.62041562],
       [-0.41002054, -0.48067656, -0.32791062],
       [ 0.80576177, -2.17976458,  0.15114801],
       [ 0.80576177, -0.09452019, -0.4672363 ],
       [ 0.80576177, -0.09452019, -0.1645248 ],
       [-0.41002054, -2.02530203, -0.11495032],
       [ 0.80576177, -1.94807076, -0.40397343],
       [ 0.80576177, -0.78960166, -0.47169217],
       [ 0.80576177,  0.87087073, -0.46974808],
       [ 0.80576177, -2.17976458, -0.31438165],
       [ 0.80576177, -0.09452019, -0.32402243],
       [ 0.80576177, -0.63513911, -0.46634591]])
```

In [15]:
```
### Model Building
from sklearn.linear_model import LogisticRegression
classification=LogisticRegression()
```

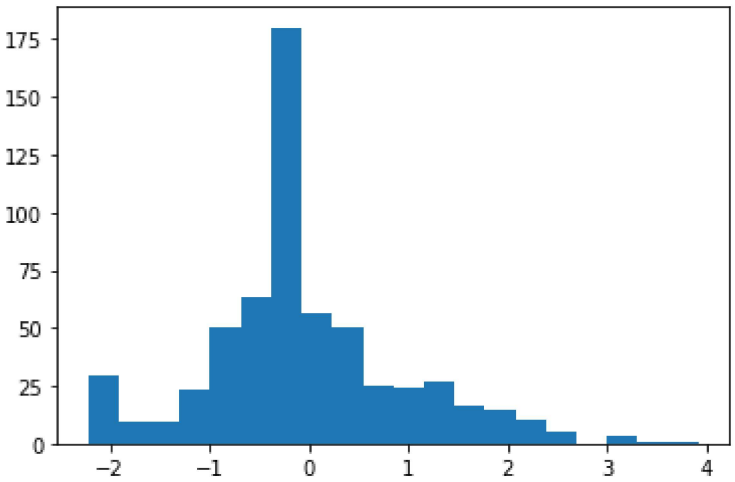
In [16]:
```
## fit() for training and predict for test
classification.fit(X_train_scaled,y_train)
classification.predict(X_test_scaled)
```

Out[16]:
```
array([0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1,
       1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0,
       1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 0], dtype=int64)
```

In [17]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

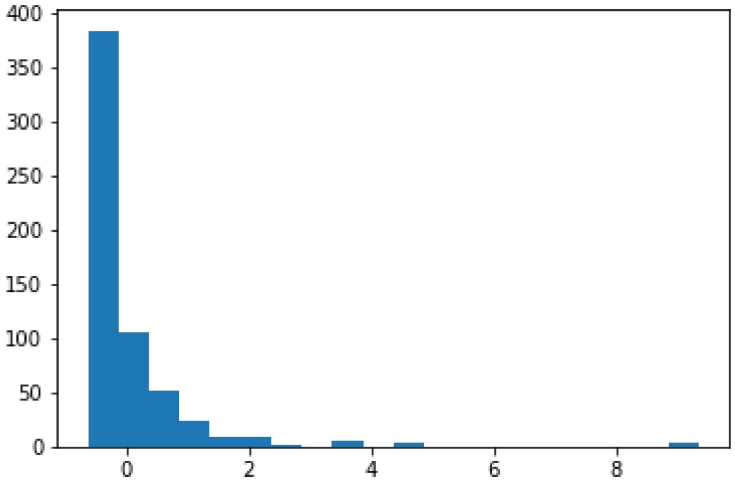
In [18]:
```
plt.hist(X_train_scaled[:,1],bins=20)
```

Out[18]:
```
(array([ 29.,   9.,   9.,  23.,  50.,  63., 180.,  56.,  50.,  25.,  24.,
         27.,  16.,  15.,  10.,   5.,   0.,   3.,   1.,   1.]),
 array([-2.22455872, -1.91725548, -1.60995224, -1.30264901, -0.99534577,
        -0.68804253, -0.38073929, -0.07343606,  0.23386718,  0.54117042,
         0.84847366,  1.15577689,  1.46308013,  1.77038337,  2.07768661,
         2.38498985,  2.69229308,  2.99959632,  3.30689956,  3.6142028 ,
         3.92150603]),
 <BarContainer object of 20 artists>)
```



In [19]:
```
plt.hist(X_train_scaled[:,2],bins=20)
```

Out[19]:
```
(array([384., 106.,  51.,  24.,   8.,   9.,   2.,   0.,   5.,   0.,   4.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   3.]),
 array([-0.62041562, -0.12240671,  0.37560221,  0.87361112,  1.37162003,
         1.86962894,  2.36763786,  2.86564677,  3.36365568,  3.8616646 ,
         4.35967351,  4.85768242,  5.35569133,  5.85370025,  6.35170916,
         6.84971807,  7.34772699,  7.8457359 ,  8.34374481,  8.84175372,
         9.33976264]),
 <BarContainer object of 20 artists>)
```



Min Max Scaling (### CNN)---Deep Learning Techniques

Min Max Scaling scales the values between 0 to 1. X_scaled = (X - X.min / (X.max - X.min)

- MinMaxScaler rescales the data set such that all feature values are in the range [0, 1]. This is done feature-wise in an independent way.
- The MinMax Scaling is that it is highly influenced by the maximum and minimum values in our data so if our data contains outliers it is going to be biased.
- The MinMaxScaler scaling might compress all inliers in a narrow range.

Min-Max scaling:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- **Min Max Scaling scales the values between 0 to 1.**

In [20]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state
```

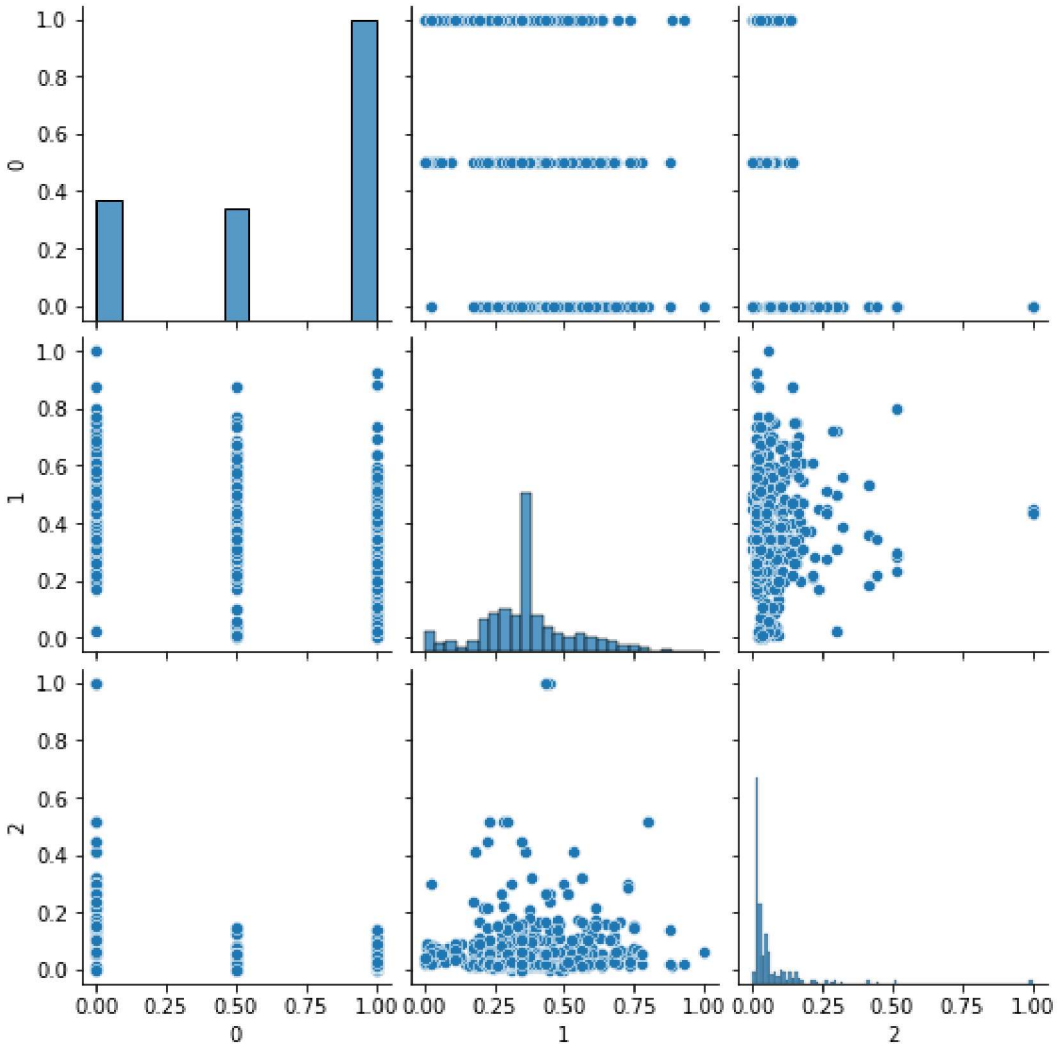
In [21]:
```
from sklearn.preprocessing import MinMaxScaler
min_max=MinMaxScaler()
df_minmax=pd.DataFrame(min_max.fit_transform(X_train))
df_minmax.head()
```

Out[21]:

	0	1	2
0	0.0	0.673285	0.101229
1	1.0	0.346569	0.030254
2	0.5	0.308872	0.081157
3	1.0	0.321438	0.028213
4	1.0	0.271174	0.020527

In [22]:
```
sns.pairplot(df_minmax)
```

Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x17ac1f4b580>
```



In [ ]:

Robust Scaler

Robust Scaler are robust to outliers.It is used to scale the feature to median and quantiles Scaling using median and quantiles consists of substracting the median to all the observations, and then dividing by the interquantile difference. The interquantile difference is the difference between the 75th and 25th quantile:

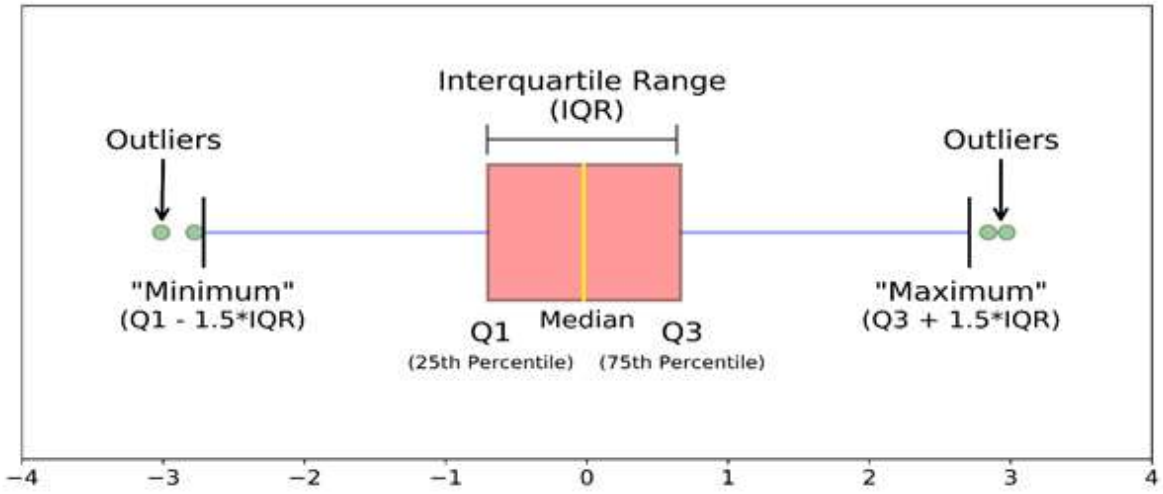
IQR = 75th quantile - 25th quantile

X_scaled = (X - X.median) / IQR

0,1,2,3,4,5,6,7,8,9,10

9-90 percentile---90% of all values in this group is less than 9 1-10 precentile---10% of all values in this group is less than 1 4-40%

$$\frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)}$$



In [23]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state

from sklearn.preprocessing import RobustScaler
scaler=RobustScaler()
df_robust_scaler=pd.DataFrame(scaler.fit_transform(X_train))
df_robust_scaler.head()
```

Out[23]:

	0	1	2
0	-2.0	2.000000	1.602069
1	0.0	0.000000	0.044788
2	-1.0	-0.230769	1.161670
3	0.0	-0.153846	0.000000
4	0.0	-0.461538	-0.168630

In [24]:
```
scaler.transform(X_test)
```

Out[24]:
```
array([[ 0.00000000e+00,  0.00000000e+00,  3.39014989e-02],
       [-1.00000000e+00,  2.30769231e-01, -1.69344754e-01],
       [ 0.00000000e+00, -6.15384615e-01, -2.79623126e-01],
       [-1.00000000e+00, -1.69230769e+00,  7.94252677e-01],
       [ 0.00000000e+00, -1.07692308e+00, -1.37580300e-01],
       [-2.00000000e+00, -1.53846154e-01,  2.75785011e+00],
       [ 0.00000000e+00,  0.00000000e+00, -2.87117773e-01],
       [ 0.00000000e+00, -9.23076923e-01,  1.51854390e-01],
       [ 0.00000000e+00, -9.23076923e-01, -2.87117773e-01],
       [-2.00000000e+00, -6.92307692e-01,  5.06599572e-01],
       [-2.00000000e+00,  6.92307692e-01,  1.65506638e+00],
       [ 0.00000000e+00,  1.23076923e+00, -2.74269807e-01],
       [ 0.00000000e+00,  0.00000000e+00,  4.71627409e-01],
       [ 0.00000000e+00,  1.53846154e-01, -3.09601713e-01],
       [-1.00000000e+00,  6.15384615e-01, -6.22783726e-02],
       [-2.00000000e+00, -9.23076923e-01,  1.06834261e+00],
       [-2.00000000e+00,  1.07692308e+00,  1.63169165e+00],
       [ 0.00000000e+00,  0.00000000e+00, -2.83725910e-01],
       [-1.00000000e+00, -7.69230769e-02, -6.22783726e-02],
       [-2.00000000e+00,  1.46153846e+00,  1.60795717e+00],
       [ 0.00000000e+00, -3.07692308e-01, -2.80873662e-01],
       [-2.00000000e+00,  4.61538462e-01,  5.18021413e-01],
       [ 0.00000000e+00, -6.92307692e-01, -2.82655246e-01],
       [ 0.00000000e+00, -6.15384615e-01, -2.23948608e-01],
       [ 0.00000000e+00,  0.00000000e+00,  1.96145610e-03],
       [ 0.00000000e+00, -1.38461538e+00,  5.75837259e-01],
       [-2.00000000e+00,  9.23076923e-01,  5.68162741e-01],
       [-1.00000000e+00,  2.30769231e-01, -6.22783726e-02],
       [ 0.00000000e+00, -1.84615385e+00,  5.75837259e-01],
       [ 0.00000000e+00,  2.30769231e-01, -2.82655246e-01],
       [ 0.00000000e+00, -6.92307692e-01, -2.80873662e-01],
       [ 0.00000000e+00, -4.61538462e-01, -2.87117773e-01],
       [-2.00000000e+00,  0.00000000e+00,  6.95571734e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.95683084e-01],
       [ 0.00000000e+00, -7.69230769e-01, -2.74269807e-01],
       [ 0.00000000e+00, -7.69230769e-02,  0.00000000e+00],
       [-2.00000000e+00,  0.00000000e+00,  9.01319058e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.87117773e-01],
       [-2.00000000e+00,  1.53846154e-01,  7.08599572e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.80873662e-01],
       [-1.00000000e+00, -5.38461538e-01,  2.52872805e+00],
       [ 0.00000000e+00,  7.69230769e-02, -3.17276231e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.74269807e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.87117773e-01],
       [ 0.00000000e+00,  1.30769231e+00,  0.00000000e+00],
       [ 0.00000000e+00, -9.23076923e-01,  8.53139186e-01],
       [ 0.00000000e+00, -6.92307692e-01, -2.69631692e-01],
       [ 0.00000000e+00, -3.84615385e-01, -2.23948608e-01],
       [ 0.00000000e+00, -3.07692308e-01, -3.17096360e-01],
       [-2.00000000e+00,  2.30769231e+00,  4.23215418e+00],
       [ 0.00000000e+00, -1.76923077e+00,  2.05743041e-01],
       [-2.00000000e+00,  1.84615385e+00,  3.38525910e+00],
       [ 0.00000000e+00,  9.23076923e-01,  5.75837259e-01],
       [-2.00000000e+00, -1.30769231e+00,  4.52016274e+00],
       [ 0.00000000e+00,  2.84615385e+00, -2.87117773e-01],
       [-2.00000000e+00, -7.69230769e-01,  2.79211135e+00],
       [-1.00000000e+00,  3.07692308e-01, -1.69344754e-01],
       [-2.00000000e+00,  1.69230769e+00,  9.98143897e+00],
       [-1.00000000e+00,  5.38461538e-01,  2.80334047e-01],
       [ 0.00000000e+00, -6.92307692e-01, -2.81584582e-01],
```

```
[ 0.00000000e+00, -5.38461538e-01, -2.86047109e-01],
[-1.00000000e+00, -1.15384615e+00,  2.16094218e-01],
[-1.00000000e+00,  0.00000000e+00,  4.94466809e-01],
[-2.00000000e+00,  0.00000000e+00,  4.91254818e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.87117773e-01],
[-1.00000000e+00,  2.23076923e+00, -1.69344754e-01],
[-1.00000000e+00, -5.38461538e-01,  2.52872805e+00],
[ 0.00000000e+00,  7.69230769e-02, -2.81764454e-01],
[ 0.00000000e+00, -8.46153846e-01, -3.09421842e-01],
[-2.00000000e+00,  1.84615385e+00,  2.73286938e+00],
[-1.00000000e+00, -1.53846154e-01,  4.94466809e-01],
[-2.00000000e+00, -7.69230769e-01,  1.06175931e+01],
[-2.00000000e+00, -1.53846154e-01,  6.65773019e-01],
[-2.00000000e+00,  8.46153846e-01,  2.94235974e+00],
[ 0.00000000e+00, -3.84615385e-01, -2.82655246e-01],
[ 0.00000000e+00,  6.15384615e-01,  4.69293362e-02],
[ 0.00000000e+00,  1.92307692e-01, -2.87117773e-01],
[-2.00000000e+00, -4.61538462e-01,  1.50089079e+00],
[-1.00000000e+00,  7.69230769e-01, -6.22783726e-02],
[ 0.00000000e+00,  1.00000000e+00,  2.46608137e-01],
[ 0.00000000e+00, -9.23076923e-01,  1.08065525e+00],
[-1.00000000e+00,  7.69230769e-02, -1.69344754e-01],
[-2.00000000e+00,  9.23076923e-01,  5.14114775e+00],
[ 0.00000000e+00,  0.00000000e+00, -2.87653105e-01],
[-2.00000000e+00,  2.15384615e+00,  6.95571734e-01],
[-2.00000000e+00,  3.30769231e+00,  1.50107066e+00],
[-2.00000000e+00,  2.30769231e-01,  4.23215418e+00],
[-2.00000000e+00,  1.53846154e-01,  1.81905782e+00],
[-2.00000000e+00,  7.69230769e-01, -6.19023555e-01],
[ 0.00000000e+00,  0.00000000e+00, -3.08531049e-01],
[ 0.00000000e+00, -1.46153846e+00,  2.59991435e-01],
[ 0.00000000e+00, -1.46153846e+00,  7.20376874e-01],
[-2.00000000e+00,  2.53846154e+00,  7.65164882e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.88008565e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.80873662e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.82655246e-01],
[-2.00000000e+00,  1.53846154e+00,  1.60795717e+00],
[ 0.00000000e+00,  1.30769231e+00, -2.74269807e-01],
[-1.00000000e+00,  6.15384615e-01, -6.22783726e-02],
[ 0.00000000e+00, -7.69230769e-02, -2.80873662e-01],
[-2.00000000e+00,  2.76923077e+00,  4.94466809e-01],
[ 0.00000000e+00,  5.38461538e-01, -3.13884368e-01],
[-2.00000000e+00, -6.92307692e-01,  3.28158458e+00],
[ 0.00000000e+00,  7.69230769e-01, -3.17096360e-01],
[ 0.00000000e+00,  1.92307692e-01, -2.74269807e-01],
[ 0.00000000e+00, -2.30769231e-01, -2.86047109e-01],
[-1.00000000e+00,  9.23076923e-01,  1.05121199e+00],
[-2.00000000e+00,  1.46153846e+00,  1.02979872e+00],
[ 0.00000000e+00,  1.53846154e-01, -2.12171306e-01],
[-2.00000000e+00,  9.23076923e-01,  7.08599572e-01],
[-2.00000000e+00,  0.00000000e+00,  1.60795717e+00],
[ 0.00000000e+00, -5.76923077e-01, -3.08531049e-01],
[-2.00000000e+00,  0.00000000e+00,  3.19700214e+00],
[-2.00000000e+00,  6.15384615e-01,  5.06779443e-01],
[-2.00000000e+00,  0.00000000e+00,  8.87901499e+00],
[ 0.00000000e+00, -6.15384615e-01, -1.97361884e-01],
[-2.00000000e+00,  3.30769231e+00,  8.65096360e-01],
[-1.00000000e+00,  6.15384615e-01, -6.76316916e-02],
[-1.00000000e+00, -4.61538462e-01,  6.22946467e-01],
[ 0.00000000e+00,  1.53846154e+00,  8.53139186e-01],
```

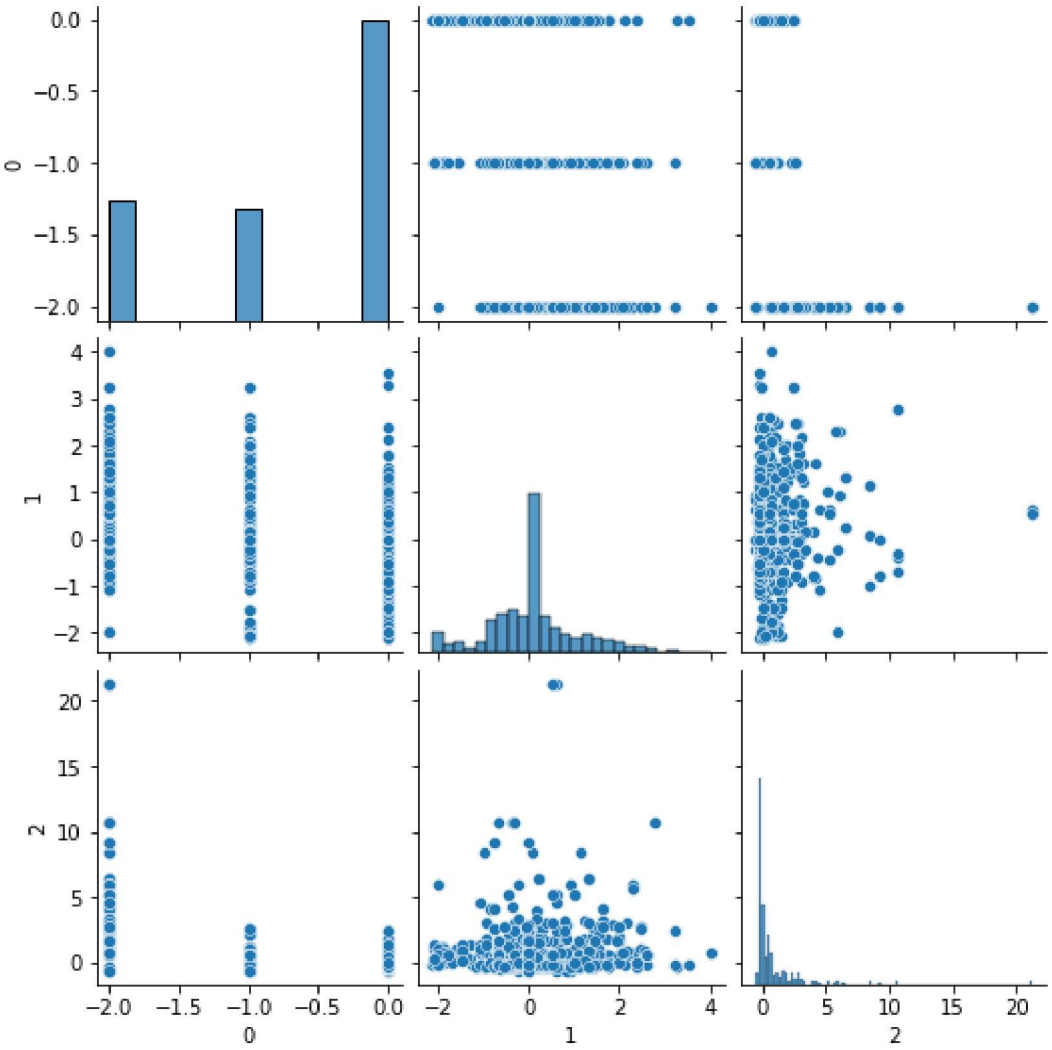
```
[-1.00000000e+00, -3.84615385e-01, -1.26518201e-01],
[-2.00000000e+00,  3.07692308e-01,  2.64828694e+00],
[-2.00000000e+00,  2.69230769e+00,  2.71966167e+00],
[-1.00000000e+00, -2.09000000e+00,  6.22946467e-01],
[ 0.00000000e+00, -2.30769231e-01, -3.08531049e-01],
[-2.00000000e+00,  5.38461538e-01,  5.18021413e-01],
[-2.00000000e+00,  1.61538462e+00,  4.91434690e-01],
[-2.00000000e+00, -3.07692308e-01,  2.77283940e+00],
[ 0.00000000e+00,  0.00000000e+00,  1.75588865e-04],
[ 0.00000000e+00, -5.38461538e-01, -2.74269807e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.81584582e-01],
[ 0.00000000e+00,  0.00000000e+00,  3.39014989e-02],
[-1.00000000e+00,  0.00000000e+00, -6.19023555e-01],
[ 0.00000000e+00, -1.76923077e+00, -8.47623126e-02],
[-2.00000000e+00,  1.61538462e+00,  1.81905782e+00],
[ 0.00000000e+00, -1.00000000e+00,  0.00000000e+00],
[-2.00000000e+00,  0.00000000e+00,  4.12972591e+00],
[ 0.00000000e+00,  7.69230769e-02, -2.86047109e-01],
[ 0.00000000e+00,  1.07692308e+00, -2.95683084e-01],
[ 0.00000000e+00, -7.69230769e-01,  1.43289079e-01],
[ 0.00000000e+00, -5.38461538e-01, -2.57854390e-01],
[-1.00000000e+00,  1.23076923e+00,  4.94466809e-01],
[-1.00000000e+00,  0.00000000e+00, -6.22783726e-02],
[-1.00000000e+00,  6.92307692e-01,  4.94466809e-01],
[ 0.00000000e+00,  8.46153846e-01,  6.28299786e-01],
[-1.00000000e+00,  4.61538462e-01,  7.72839400e-01],
[-1.00000000e+00,  1.53846154e-01, -6.22783726e-02],
[-1.00000000e+00,  2.92307692e+00, -1.69344754e-01],
[-2.00000000e+00,  0.00000000e+00,  5.65595717e+00],
[ 0.00000000e+00,  1.15384615e+00, -2.74269807e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.45897216e-01],
[ 0.00000000e+00, -3.07692308e-01, -2.98004283e-01],
[ 0.00000000e+00, -7.69230769e-01, -2.85156317e-01],
[ 0.00000000e+00,  0.00000000e+00,  4.15237687e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.74269807e-01],
[ 0.00000000e+00,  0.00000000e+00, -3.17096360e-01],
[ 0.00000000e+00,  0.00000000e+00,  1.80049679e+00],
[ 0.00000000e+00,  0.00000000e+00, -2.85511777e-01],
[-1.00000000e+00,  6.15384615e-01,  4.94466809e-01],
[ 0.00000000e+00,  3.07692308e-01,  4.47880086e-02],
[ 0.00000000e+00, -3.46153846e-01, -3.09421842e-01],
[-2.00000000e+00,  0.00000000e+00,  5.18021413e-01],
[-1.00000000e+00,  8.46153846e-01, -6.22783726e-02],
[-2.00000000e+00,  0.00000000e+00,  2.90006852e+00],
[ 0.00000000e+00,  0.00000000e+00, -2.56783726e-01],
[-2.00000000e+00,  2.84615385e+00,  2.03533191e+00],
[-1.00000000e+00, -3.84615385e-01, -1.69344754e-01],
[-2.00000000e+00,  0.00000000e+00,  1.60206852e+00],
[ 0.00000000e+00, -4.61538462e-01, -2.80873662e-01],
[ 0.00000000e+00,  5.38461538e-01, -3.17096360e-01],
[-2.00000000e+00, -3.84615385e-01,  2.09439400e+00],
[-1.00000000e+00, -7.69230769e-01, -6.22783726e-02],
[-2.00000000e+00,  6.15384615e-01,  5.11062099e-01],
[ 0.00000000e+00, -1.46153846e+00,  3.39014989e-02],
[ 0.00000000e+00, -8.46153846e-01, -3.13884368e-01],
[ 0.00000000e+00,  0.00000000e+00, -3.09601713e-01],
[ 0.00000000e+00,  7.69230769e-01,  7.25194861e-01],
[-1.00000000e+00, -8.46153846e-01, -1.69344754e-01],
[ 0.00000000e+00, -1.84615385e+00,  9.61798715e-02],
[-2.00000000e+00,  1.34615385e+00,  6.01533191e-01],
```

```
[-1.00000000e+00, -3.84615385e-01, -6.22783726e-02],
[ 0.00000000e+00,  3.07692308e-01, -2.79623126e-01],
[ 0.00000000e+00, -1.53846154e-01, -2.82655246e-01],
[ 0.00000000e+00, -1.69230769e+00,  7.20376874e-01],
[-2.00000000e+00, -3.07692308e-01,  9.98143897e+00],
[-2.00000000e+00,  1.30769231e+00,  5.18021413e-01],
[-1.00000000e+00,  7.69230769e-02,  5.68162741e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.80873662e-01],
[-2.00000000e+00,  0.00000000e+00,  9.01319058e-01],
[-1.00000000e+00,  1.07692308e+00, -6.22783726e-02],
[ 0.00000000e+00,  6.15384615e-01,  4.15237687e-01],
[-1.00000000e+00,  3.84615385e-01, -9.33276231e-02],
[ 0.00000000e+00, -8.46153846e-01, -3.16916488e-01],
[ 0.00000000e+00,  7.69230769e-02, -2.12171306e-01],
[-1.00000000e+00,  1.69230769e+00,  4.94466809e-01],
[-2.00000000e+00,  5.38461538e-01,  3.23536617e+00],
[-2.00000000e+00,  7.69230769e-01,  9.12508779e+00],
[-1.00000000e+00,  4.61538462e-01, -6.22783726e-02],
[-2.00000000e+00, -8.46153846e-01,  1.82208994e+00],
[ 0.00000000e+00, -1.30769231e+00,  7.20376874e-01],
[ 0.00000000e+00,  2.53846154e+00, -3.51892934e-01],
[ 0.00000000e+00,  1.53846154e-01, -2.48038544e-01],
[-1.00000000e+00, -1.61538462e+00,  5.05173448e-01],
[ 0.00000000e+00,  2.69230769e+00, -2.08423983e-01],
[ 0.00000000e+00, -6.15384615e-01, -3.09421842e-01],
[ 0.00000000e+00,  0.00000000e+00,  3.38505353e-01],
[ 0.00000000e+00,  7.69230769e-02, -2.12886510e-01],
[-2.00000000e+00,  6.15384615e-01,  4.52016274e+00],
[ 0.00000000e+00,  0.00000000e+00,  1.75588865e-04],
[ 0.00000000e+00,  1.69230769e+00, -2.74269807e-01],
[-2.00000000e+00, -7.69230769e-02,  8.43879229e+00],
[ 0.00000000e+00,  1.53846154e-01, -3.08531049e-01],
[-2.00000000e+00,  3.84615385e-01,  3.23536617e+00],
[ 0.00000000e+00,  0.00000000e+00, -2.88188437e-01],
[ 0.00000000e+00,  0.00000000e+00,  4.71627409e-01],
[ 0.00000000e+00, -2.00000000e+00,  2.83546039e-01],
[-1.00000000e+00, -2.30769231e-01,  6.65773019e-01],
[-2.00000000e+00,  1.76923077e+00,  2.00963597e+00],
[-1.00000000e+00, -2.30769231e-01,  4.94466809e-01],
[ 0.00000000e+00,  0.00000000e+00, -3.08531049e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.80873662e-01],
[-1.00000000e+00, -3.07692308e-01, -6.22783726e-02],
[ 0.00000000e+00, -7.69230769e-01,  2.46608137e-01],
[-2.00000000e+00,  0.00000000e+00,  6.87186296e-01],
[ 0.00000000e+00, -2.30769231e-01, -3.17096360e-01],
[-1.00000000e+00, -3.07692308e-01,  1.96145610e-03],
[ 0.00000000e+00, -4.61538462e-01, -2.96933619e-01],
[-2.00000000e+00, -2.08307692e+00,  5.87134047e+00],
[-1.00000000e+00, -3.07692308e-01,  4.94466809e-01],
[ 0.00000000e+00, -1.53846154e-01, -2.79623126e-01],
[-1.00000000e+00,  4.61538462e-01,  2.80334047e-01],
[-2.00000000e+00, -5.38461538e-01,  1.06175931e+01],
[ 0.00000000e+00,  0.00000000e+00, -2.87117773e-01],
[ 0.00000000e+00,  0.00000000e+00, -3.09421842e-01],
[ 0.00000000e+00,  0.00000000e+00, -2.48038544e-01],
[ 0.00000000e+00, -4.61538462e-01, -2.86047109e-01],
[-2.00000000e+00,  2.61538462e+00,  2.80710064e+00],
[ 0.00000000e+00, -7.69230769e-01, -1.97537473e-01],
[-2.00000000e+00,  1.07692308e+00,  9.12508779e+00],
[-1.00000000e+00,  2.23076923e+00, -9.01156317e-02],
```

```
       [ 0.00000000e+00, -6.92307692e-01, -6.19023555e-01],
       [-1.00000000e+00,  1.07692308e+00,  4.94466809e-01],
       [ 0.00000000e+00,  0.00000000e+00, -3.09601713e-01],
       [ 0.00000000e+00, -6.15384615e-01, -2.48038544e-01],
       [ 0.00000000e+00,  5.00000000e-01, -3.43327623e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.12171306e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.74269807e-01],
       [ 0.00000000e+00, -3.84615385e-01, -2.80873662e-01],
       [ 0.00000000e+00, -4.61538462e-01, -3.08531049e-01],
       [-2.00000000e+00,  1.84615385e+00,  6.87186296e-01],
       [ 0.00000000e+00,  1.53846154e-01, -8.47623126e-02],
       [-1.00000000e+00,  6.53846154e-01,  4.94466809e-01],
       [ 0.00000000e+00,  0.00000000e+00, -2.87117773e-01],
       [ 0.00000000e+00, -4.61538462e-01, -3.09421842e-01],
       [-2.00000000e+00,  1.69230769e+00,  5.10474518e+00],
       [ 0.00000000e+00, -7.69230769e-02, -3.20308351e-01],
       [ 0.00000000e+00, -4.61538462e-01, -2.74269807e-01],
       [-2.00000000e+00,  1.76923077e+00,  2.71966167e+00],
       [-1.00000000e+00,  6.15384615e-01, -1.69344754e-01],
       [-1.00000000e+00,  7.69230769e-02,  4.94466809e-01],
       [-2.00000000e+00, -4.61538462e-01,  5.87134047e+00],
       [-2.00000000e+00,  1.69230769e+00,  3.93879229e+00],
       [-1.00000000e+00,  9.23076923e-01, -6.22783726e-02],
       [ 0.00000000e+00, -7.69230769e-02, -2.48038544e-01],
       [-2.00000000e+00,  0.00000000e+00, -6.19023555e-01],
       [-2.00000000e+00,  2.84615385e+00,  5.18021413e-01],
       [ 0.00000000e+00, -8.46153846e-01, -2.48038544e-01],
       [-2.00000000e+00,  0.00000000e+00,  5.18021413e-01],
       [-2.00000000e+00, -1.84615385e+00,  2.88668522e+00],
       [ 0.00000000e+00,  0.00000000e+00, -2.80873662e-01],
       [ 0.00000000e+00, -2.07692308e+00, -1.42222698e-01],
       [-1.00000000e+00,  6.15384615e-01,  5.69413276e-01],
       [-1.00000000e+00,  1.15384615e+00,  5.05173448e-01],
       [-2.00000000e+00,  7.69230769e-01,  5.95324625e+00],
       [-1.00000000e+00,  2.30769231e-01,  5.05173448e-01],
       [ 0.00000000e+00,  7.69230769e-02, -2.74269807e-01],
       [ 0.00000000e+00, -7.69230769e-01, -2.63563169e-01],
       [-1.00000000e+00,  0.00000000e+00,  2.55160600e-02],
       [-2.00000000e+00,  8.46153846e-01,  4.12972591e+00],
       [-1.00000000e+00,  8.46153846e-01, -6.22783726e-02],
       [ 0.00000000e+00, -1.53846154e-01, -2.48038544e-01],
       [ 0.00000000e+00, -6.15384615e-01, -3.17096360e-01],
       [-2.00000000e+00,  0.00000000e+00,  5.10474518e+00],
       [ 0.00000000e+00,  1.61538462e+00, -6.19023555e-01],
       [-1.00000000e+00, -3.84615385e-01,  2.53361884e-02],
       [ 0.00000000e+00, -2.07692308e+00,  1.08065525e+00],
       [ 0.00000000e+00,  0.00000000e+00, -2.81584582e-01],
       [ 0.00000000e+00,  0.00000000e+00,  3.85259101e-01],
       [-1.00000000e+00, -1.92307692e+00,  4.94466809e-01],
       [ 0.00000000e+00, -1.84615385e+00, -1.42222698e-01],
       [ 0.00000000e+00, -6.92307692e-01, -2.91400428e-01],
       [ 0.00000000e+00,  9.61538462e-01, -2.87117773e-01],
       [ 0.00000000e+00, -2.07692308e+00,  5.51391863e-02],
       [ 0.00000000e+00,  0.00000000e+00,  3.39014989e-02],
       [ 0.00000000e+00, -5.38461538e-01, -2.79623126e-01]])
```

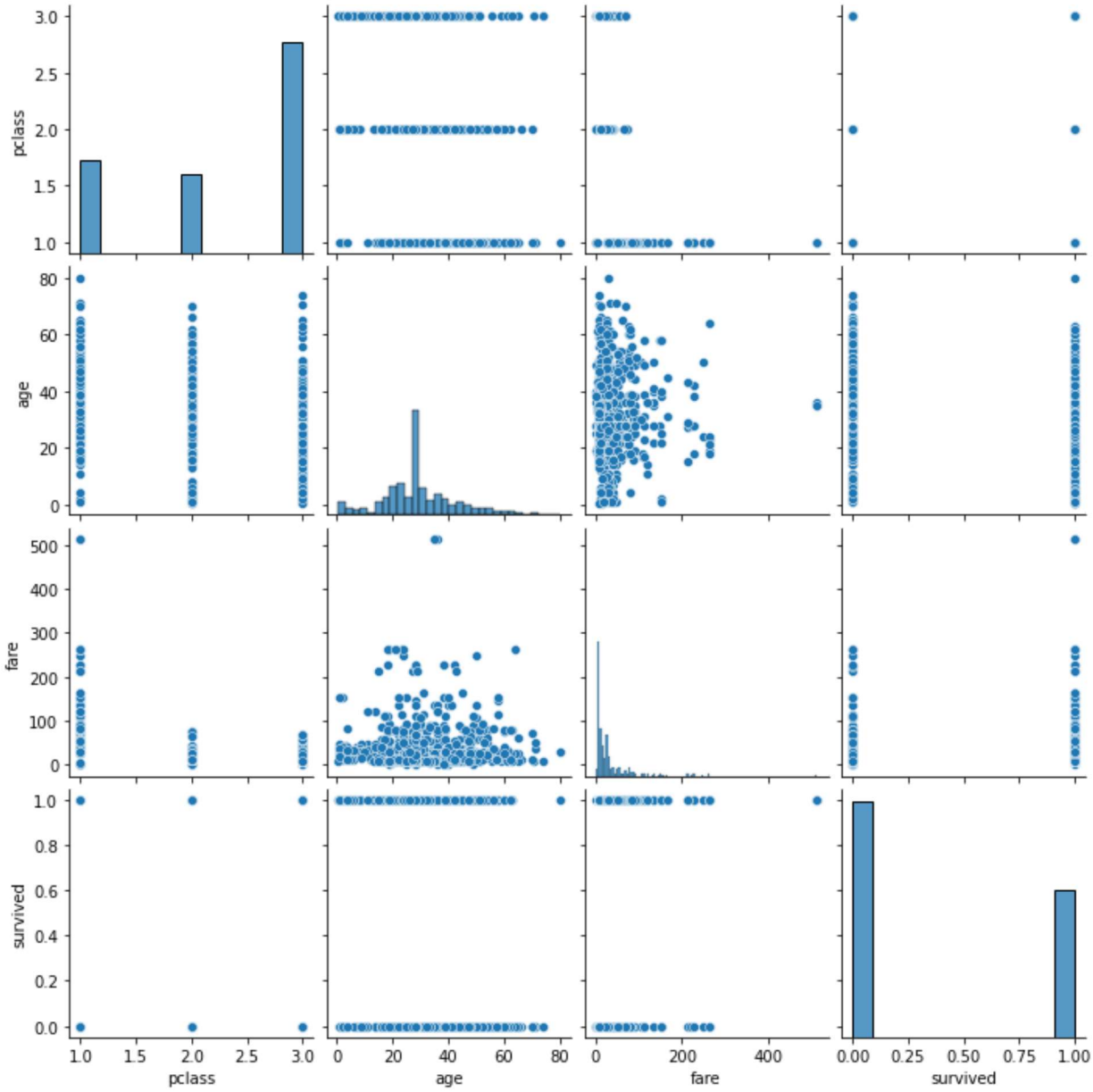
In [25]:
```
import seaborn as sns
sns.pairplot(df_robust_scaler)
```

Out[25]: <seaborn.axisgrid.PairGrid at 0x17ac2658f70>



In [26]:
```
import seaborn as sns
sns.pairplot(titanic)
```

Out[26]: <seaborn.axisgrid.PairGrid at 0x17ac2699640>



Guassian Transformation

Some machine learning algorithms like linear and logistic assume that the features are normally distributed -Accuracy -Performance

- logarithmic transformation
- reciprocal transformation
- square root transformation
- exponential transformation (more general, you can use any exponent)
- boxcox transformation

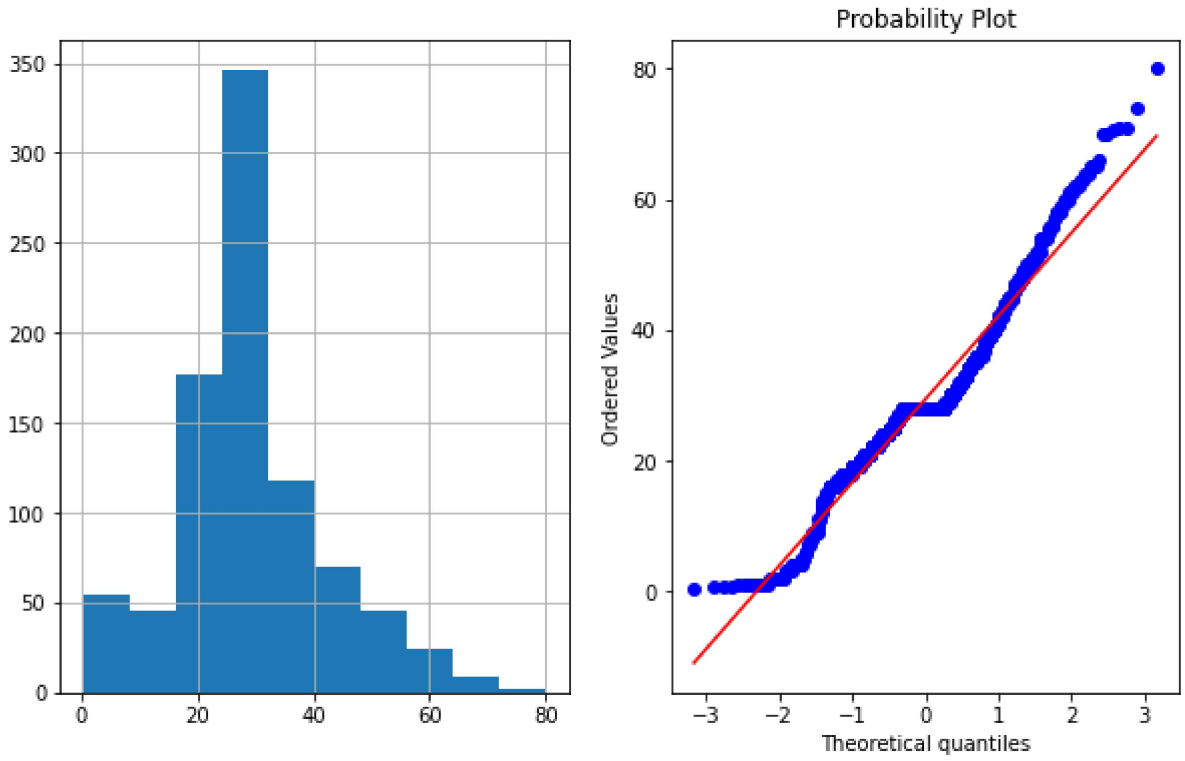
In [27]:
```
import matplotlib.pyplot as plt
import scipy.stats as stat
import pylab
```

Q-Q plot

If you want to check whether feature is guassian or normal distributed

In [28]:
```
def plot_data(df,feature):
    plt.figure(figsize=(10,6))
    plt.subplot(1,2,1)
    df[feature].hist()
    plt.subplot(1,2,2)
    stat.probplot(df[feature],dist='norm',plot=pylab)
    plt.show()
```

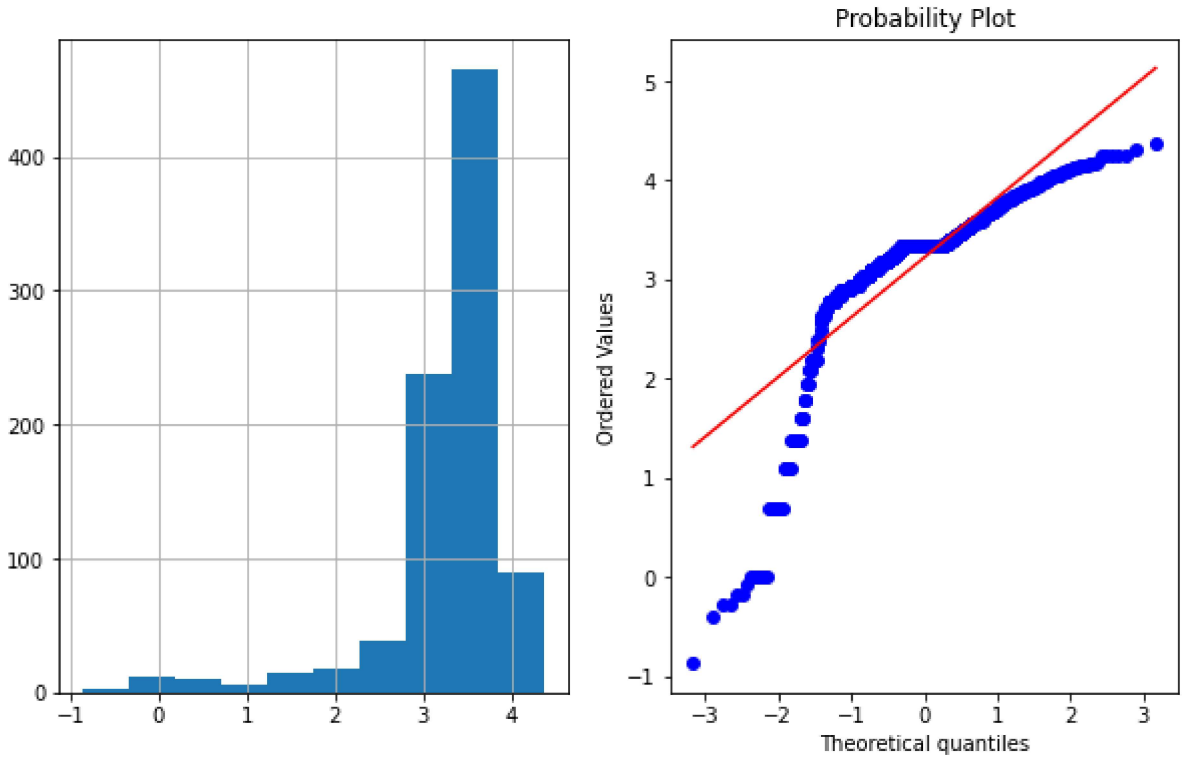
In [29]:
```
plot_data(titanic,'age')
```



Logarithmic Transformation

- Generally, these transformations make our data close to a normal distribution but are not able to exactly abide by a normal distribution.
- his transformation is not applied to those features which have negative values.
- This transformation is mostly applied to right-skewed data.
- Convert data from addictive Scale to multiplicative scale i,e, linearly distributed data.

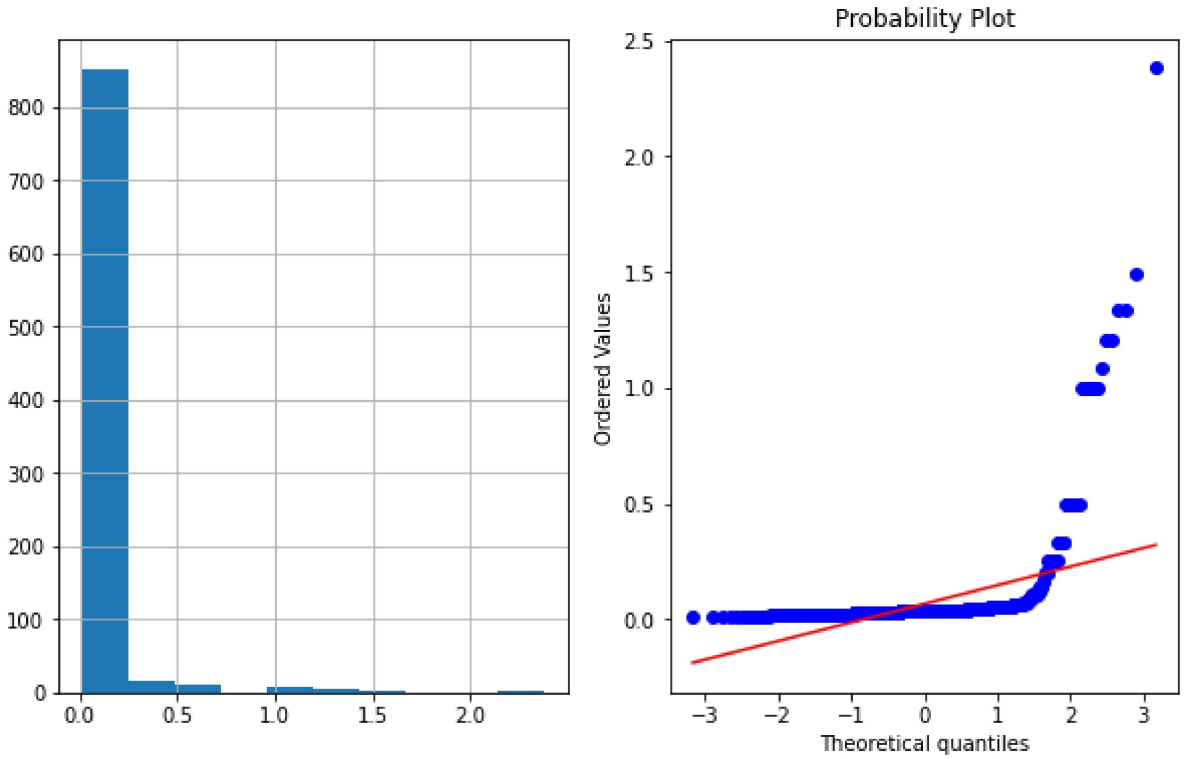
In [30]:
```
import numpy as np
titanic['age_log']=np.log(titanic['age'])
plot_data(titanic,'age_log')
```



Reciprocal Trnasformation

- This transformation is not defined for zero.
- It is a powerful transformation with a radical effect.
- This transformation reverses the order among values of the same sign, so large values become smaller and vice-versa

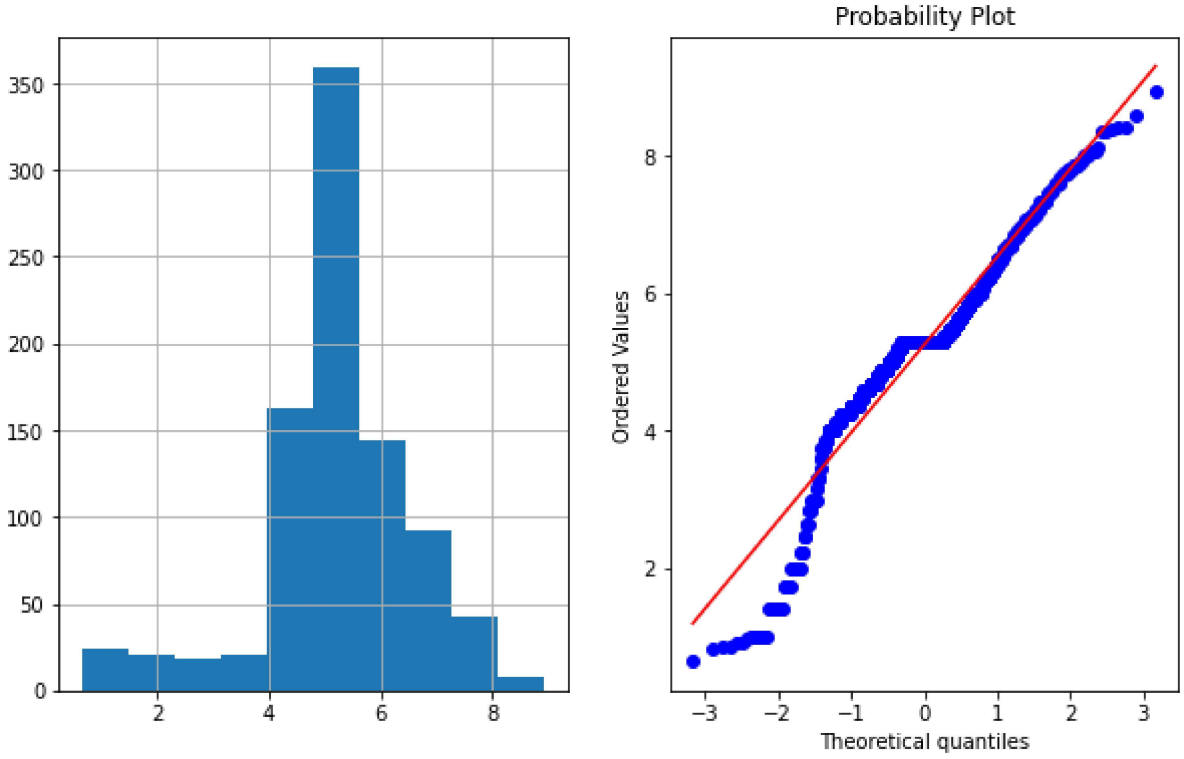
In [31]:
```
titanic['age_reciprocal']=1/titanic.age
plot_data(titanic,'age_reciprocal')
```



Square Root Transformation

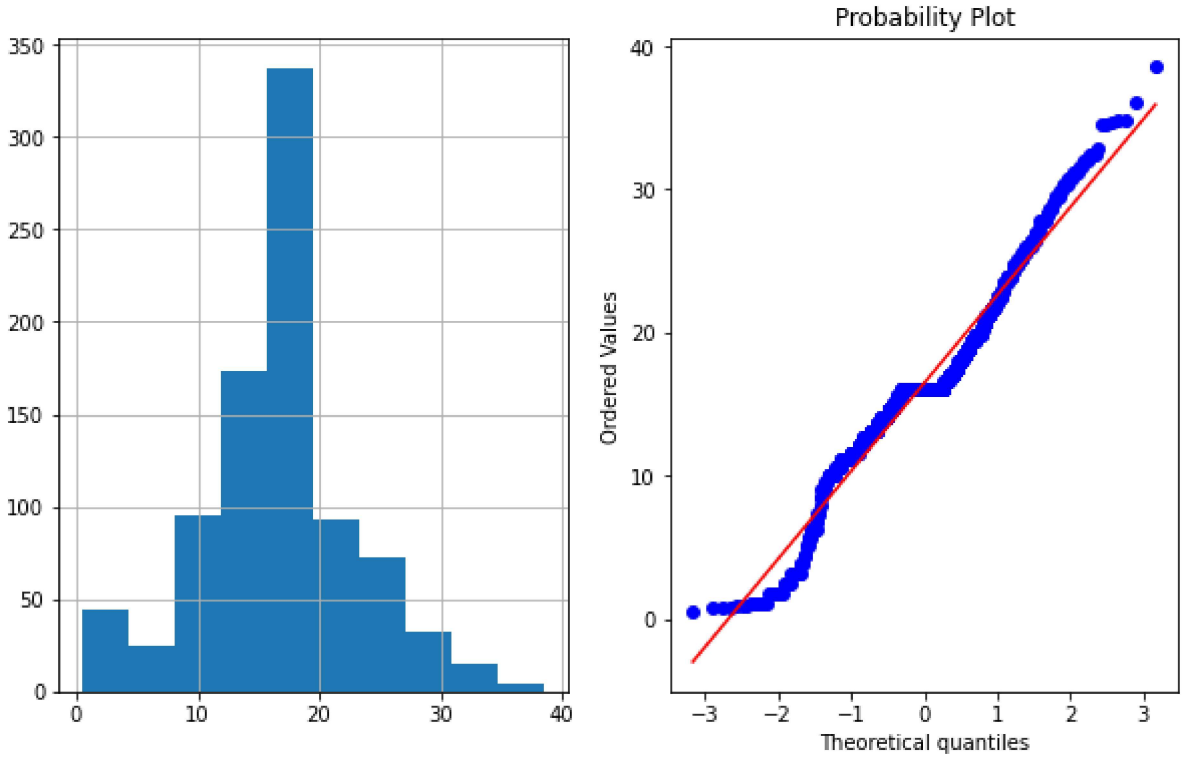
- The square root, x to x^(1/2) = sqrt(x), is a transformation with a moderate effect on distribution shape.
- This transformation is defined only for positive numbers.
- This transformation is weaker than Log Transformation.
- This can be used for reducing the skewness of right-skewed data and also has the advantage that it can be applied to zero values.

In [32]:
```
titanic['age_sqaure']=titanic.age**(1/2)
plot_data(titanic,'age_sqaure')
```



Exponential Transformation

In [33]:
```
titanic['age_exponential']=titanic.age**(1/1.2)
plot_data(titanic,'age_exponential')
```



BoxCOx Transformation

- Many machine learning algorithms prefer or perform better when numerical variables have a Gaussian probability distribution.

- Power transforms are a technique for transforming numerical input or output variables to have a Gaussian or more-Gaussian-like probability distribution.
- How to use the PowerTransform in scikit-learn to use the Box-Cox and Yeo-Johnson transforms when preparing data for predictive modeling.
- It is a power transform that assumes the values of the input variable to which it is applied are strictly positive. That means 0 and negative values are not supported.

The Box-Cox transformation is defined as:

$T(Y)=(Y\ \exp(\lambda)-1)/\lambda$

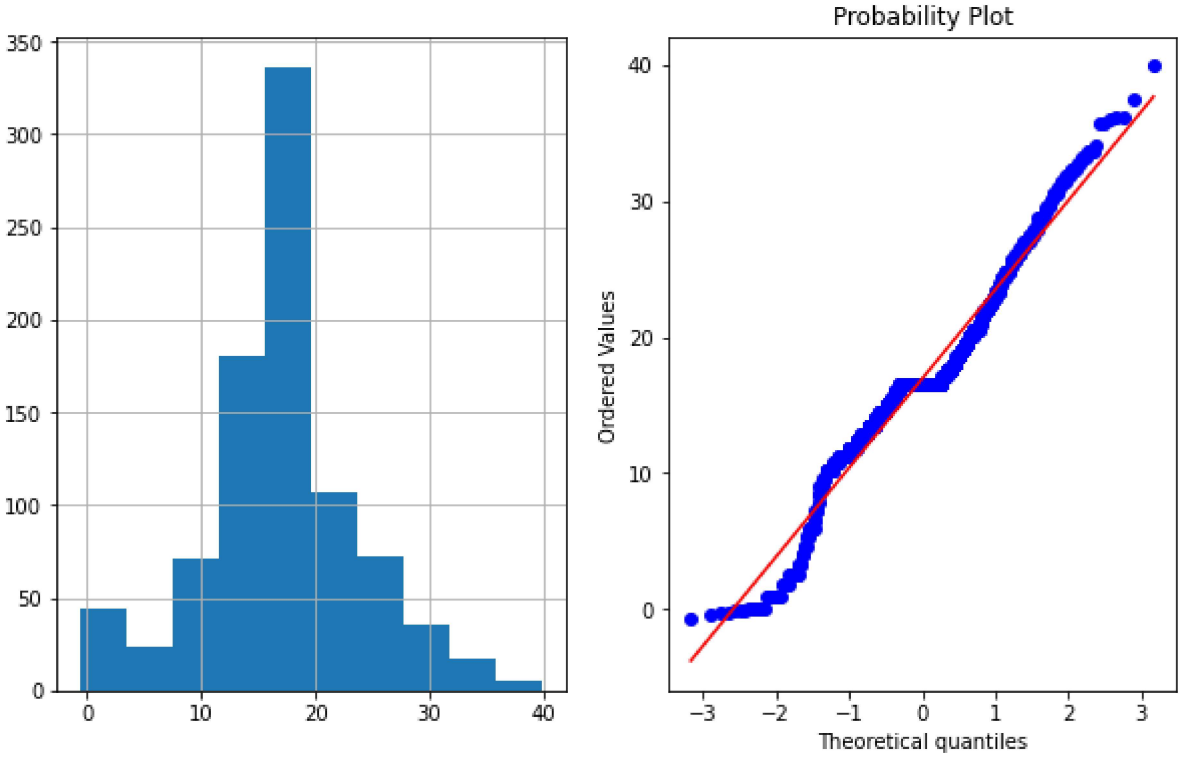
where Y is the response variable and λ is the transformation parameter. λ varies from -5 to 5. In the transformation, all values of λ are considered and the optimal value for a given variable is selected.

In [34]:
```
titanic['age_Boxcox'],parameters=stat.boxcox(titanic['age'])
```

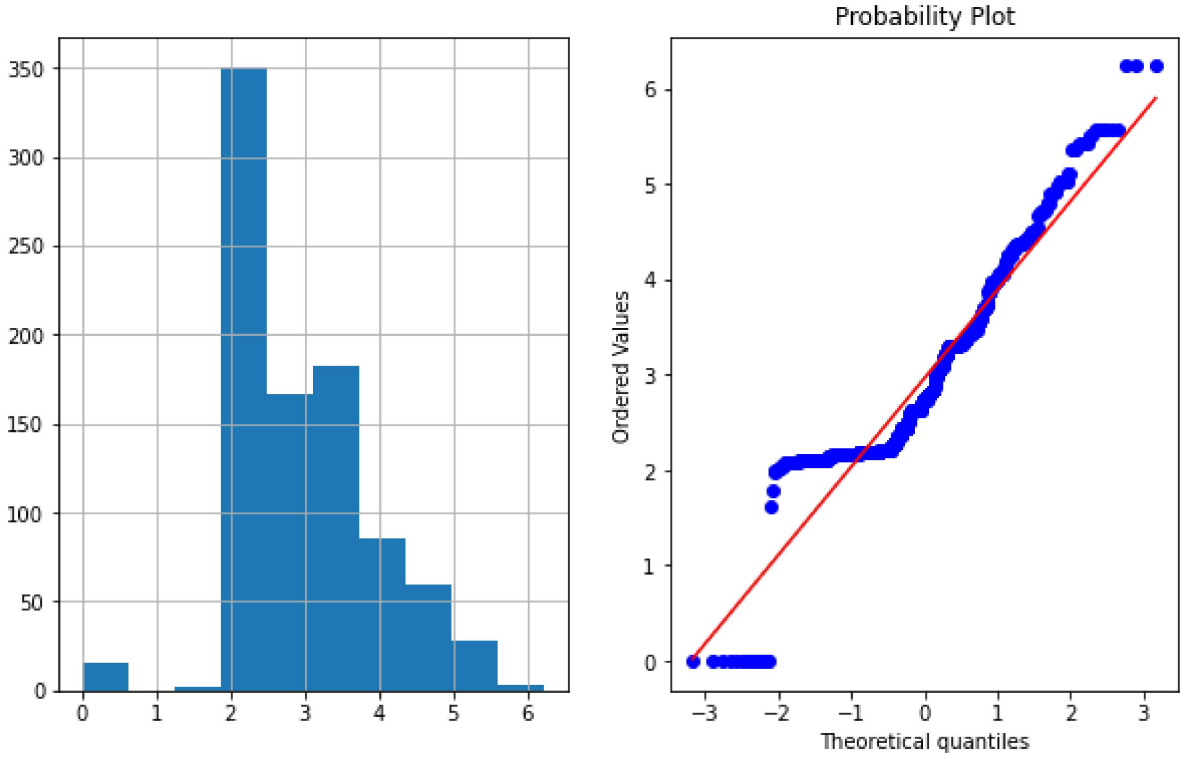
In [35]:
```
print(parameters)
```

```
0.7964531473656952
```

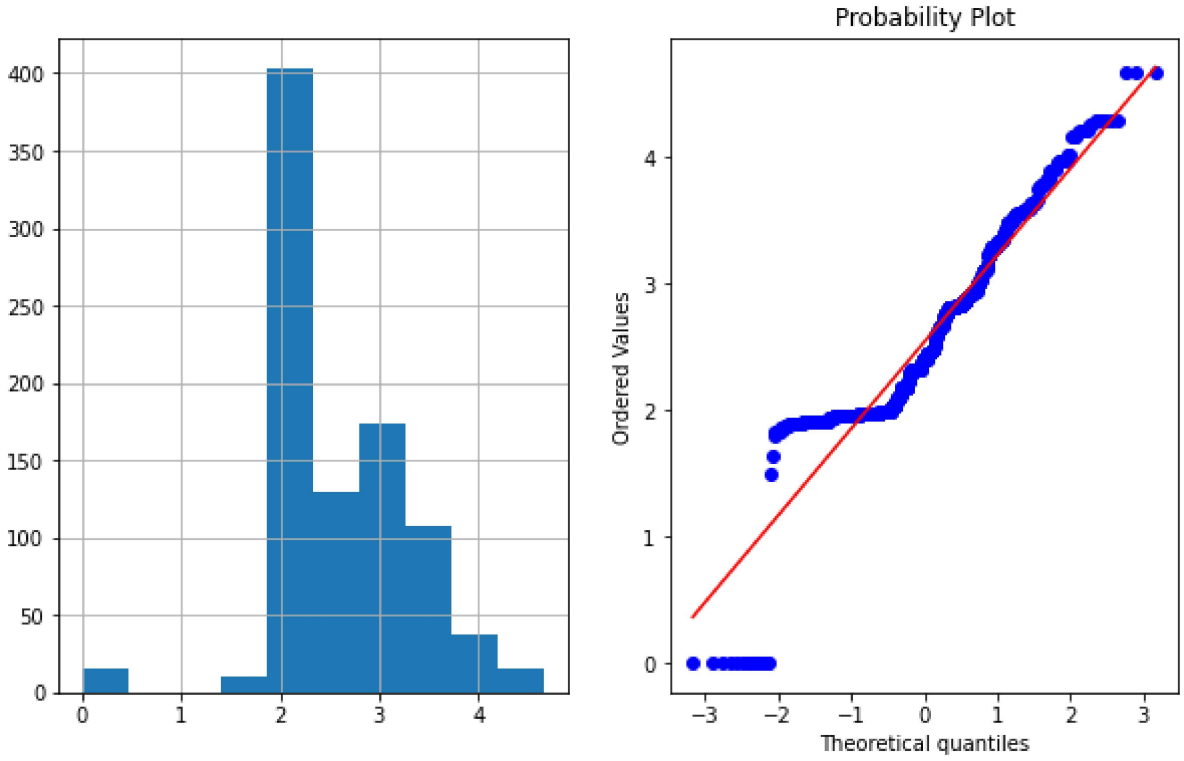
In [36]:
```
plot_data(titanic,'age_Boxcox')
```



In [37]:
```
#### Fare
titanic['fare_log']=np.log1p(titanic['fare'])
plot_data(titanic,'fare_log')
```



In [38]:
```
titanic['fare_Boxcox'],parameters=stat.boxcox(titanic['fare']+1)
plot_data(titanic,'fare_Boxcox')
```



In [ ]:

In [ ]: