exercise-no-3

September 25, 2024

[ ]:

```
import pandas as pd
df=pd.read_csv("/content/Encoding Data (1).csv")
df
```

[ ]:

```
   id bin_1 bin_2  nom_0 ord_2
0   0     F     N    Red   Hot
1   1     F     Y   Blue  Warm
2   2     F     N   Blue  Cold
3   3     F     N  Green  Warm
4   4     T     N    Red  Cold
5   5     T     N  Green   Hot
6   6     F     N    Red  Cold
7   7     T     N    Red  Cold
8   8     F     N   Blue  Warm
9   9     F     Y    Red   Hot
```

[3]:

```
from sklearn.preprocessing import LabelEncoder,OrdinalEncoder
import pandas as pd
df=pd.read_csv("/content/Encoding Data (1).csv")
pm=["Hot","Warm","Cold"]
e1=OrdinalEncoder(categories=[pm])
e1.fit_transform(df[["ord_2"]])
```

[3]:

```
array([[0.],
       [1.],
       [2.],
       [1.],
       [2.],
       [0.],
       [2.],
       [2.],
       [1.],
       [0.]])
```

[4]:

```
df["bo2"]=e1.fit_transform(df[["ord_2"]])
df
```

```
[4]:    id bin_1 bin_2  nom_0 ord_2  bo2
     0   0     F     N    Red   Hot  0.0
     1   1     F     Y   Blue  Warm  1.0
     2   2     F     N   Blue  Cold  2.0
     3   3     F     N  Green  Warm  1.0
     4   4     T     N    Red  Cold  2.0
     5   5     T     N  Green   Hot  0.0
     6   6     F     N    Red  Cold  2.0
     7   7     T     N    Red  Cold  2.0
     8   8     F     N   Blue  Warm  1.0
     9   9     F     Y    Red   Hot  0.0
```

```
[5]: le=LabelEncoder()
     dfc=df.copy()
     dfc["ord_2"]=le.fit_transform(dfc["ord_2"])
     dfc
```

```
[5]:    id bin_1 bin_2  nom_0  ord_2  bo2
     0   0     F     N    Red      1  0.0
     1   1     F     Y   Blue      2  1.0
     2   2     F     N   Blue      0  2.0
     3   3     F     N  Green      2  1.0
     4   4     T     N    Red      0  2.0
     5   5     T     N  Green      1  0.0
     6   6     F     N    Red      0  2.0
     7   7     T     N    Red      0  2.0
     8   8     F     N   Blue      2  1.0
     9   9     F     Y    Red      1  0.0
```

```
[10]: from sklearn.preprocessing import OneHotEncoder
      ohe = OneHotEncoder(sparse_output=False)
      df2=df.copy()
      enc=pd.DataFrame(ohe.fit_transform(df[["nom_0"]]))
      df2=pd.concat([df2,enc],axis=1)
      df2
```

```
[10]:    id bin_1 bin_2  nom_0 ord_2  bo2    0    1    2
      0   0     F     N    Red   Hot  0.0  0.0  0.0  1.0
      1   1     F     Y   Blue  Warm  1.0  1.0  0.0  0.0
      2   2     F     N   Blue  Cold  2.0  1.0  0.0  0.0
      3   3     F     N  Green  Warm  1.0  0.0  1.0  0.0
      4   4     T     N    Red  Cold  2.0  0.0  0.0  1.0
      5   5     T     N  Green   Hot  0.0  0.0  1.0  0.0
      6   6     F     N    Red  Cold  2.0  0.0  0.0  1.0
      7   7     T     N    Red  Cold  2.0  0.0  0.0  1.0
      8   8     F     N   Blue  Warm  1.0  1.0  0.0  0.0
      9   9     F     Y    Red   Hot  0.0  0.0  0.0  1.0
```

```
[11]: pd.get_dummies(df2,columns=["nom_0"])
```

```
[11]:    id bin_1 bin_2 ord_2  bo2    0    1    2  nom_0_Blue  nom_0_Green  \
      0   0     F     N   Hot  0.0  0.0  0.0  1.0       False        False
      1   1     F     Y  Warm  1.0  1.0  0.0  0.0        True        False
      2   2     F     N  Cold  2.0  1.0  0.0  0.0        True        False
      3   3     F     N  Warm  1.0  0.0  1.0  0.0       False         True
      4   4     T     N  Cold  2.0  0.0  0.0  1.0       False        False
      5   5     T     N   Hot  0.0  0.0  1.0  0.0       False         True
      6   6     F     N  Cold  2.0  0.0  0.0  1.0       False        False
      7   7     T     N  Cold  2.0  0.0  0.0  1.0       False        False
      8   8     F     N  Warm  1.0  1.0  0.0  0.0        True        False
      9   9     F     Y   Hot  0.0  0.0  0.0  1.0       False        False

         nom_0_Red
      0       True
      1      False
      2      False
      3      False
      4       True
      5      False
      6       True
      7       True
      8      False
      9       True
```

```
[12]: pip install category_encoders
```

```
Collecting category_encoders
  Downloading category_encoders-2.6.3-py2.py3-none-any.whl.metadata (8.0 kB)
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (1.26.4)
Requirement already satisfied: scikit-learn>=0.20.0 in
/usr/local/lib/python3.10/dist-packages (from category_encoders) (1.5.2)
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (1.13.1)
Requirement already satisfied: statsmodels>=0.9.0 in
/usr/local/lib/python3.10/dist-packages (from category_encoders) (0.14.3)
Requirement already satisfied: pandas>=1.0.5 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (2.1.4)
Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (0.5.6)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas>=1.0.5->category_encoders)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas>=1.0.5->category_encoders) (2024.2)
```

```
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas>=1.0.5->category_encoders) (2024.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages
(from patsy>=0.5.1->category_encoders) (1.16.0)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn>=0.20.0->category_encoders) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-
learn>=0.20.0->category_encoders) (3.5.0)
Requirement already satisfied: packaging>=21.3 in
/usr/local/lib/python3.10/dist-packages (from
statsmodels>=0.9.0->category_encoders) (24.1)
Downloading category_encoders-2.6.3-py2.py3-none-any.whl (81 kB)
                         81.9/81.9 kB
2.0 MB/s eta 0:00:00
Installing collected packages: category_encoders
Successfully installed category_encoders-2.6.3
```

[19]:

```
from category_encoders import BinaryEncoder
be=BinaryEncoder()
df4=pd.read_csv("/content/data.csv")
dfb=be.fit_transform(df4['Ord_2'])
df3=pd.concat([df4,dfb],axis=1)
df3
```

[19]:

```
   id bin_1 bin_2       City     Ord_1        Ord_2  Target  Ord_2_0  Ord_2_1  \
0   0     F     N      Delhi       Hot  High School       0        0        0
1   1     F     Y  Bangalore      Warm      Masters       1        0        1
2   2     M     N     Mumbai  Very Hot      Diploma       1        0        1
3   3     M     Y    Chennai      Cold    Bachelors       0        1        0
4   4     M     Y      Delhi      Cold    Bachelors       1        1        0
5   5     F     N      Delhi  Very Hot      Masters       0        0        1
6   6     M     N    Chennai      Warm          PhD       1        1        0
7   7     F     N    Chennai       Hot  High School       1        0        0
8   8     M     N      Delhi  Very Hot  High School       0        0        0
9   9     F     Y      Delhi      Warm          PhD       0        1        0

   Ord_2_2
0        1
1        0
2        1
3        0
4        0
5        0
6        1
7        1
8        1
```

```
9         1
```

[23]:
```
from category_encoders import TargetEncoder
te=TargetEncoder()
cc=df4.copy()
new=te.fit_transform(X=cc["City"],y=cc["Target"])
cc=pd.concat([cc,new],axis=1)
cc
```

[23]:
```
   id bin_1 bin_2       City     Ord_1       Ord_2  Target      City
0   0     F     N      Delhi       Hot  High School       0  0.445272
1   1     F     Y  Bangalore      Warm     Masters       1  0.565054
2   2     M     N     Mumbai  Very Hot     Diploma       1  0.565054
3   3     M     Y    Chennai      Cold   Bachelors       0  0.525744
4   4     M     Y      Delhi      Cold   Bachelors       1  0.445272
5   5     F     N      Delhi  Very Hot     Masters       0  0.445272
6   6     M     N    Chennai      Warm         PhD       1  0.525744
7   7     F     N    Chennai       Hot  High School       1  0.525744
8   8     M     N      Delhi  Very Hot  High School       0  0.445272
9   9     F     Y      Delhi      Warm         PhD       0  0.445272
```

[23]:
```
   id bin_1 bin_2       City     Ord_1       Ord_2  Target      City
0   0     F     N      Delhi       Hot  High School       0  0.445272
1   1     F     Y  Bangalore      Warm     Masters       1  0.565054
2   2     M     N     Mumbai  Very Hot     Diploma       1  0.565054
3   3     M     Y    Chennai      Cold   Bachelors       0  0.525744
4   4     M     Y      Delhi      Cold   Bachelors       1  0.445272
5   5     F     N      Delhi  Very Hot     Masters       0  0.445272
6   6     M     N    Chennai      Warm         PhD       1  0.525744
7   7     F     N    Chennai       Hot  High School       1  0.525744
8   8     M     N      Delhi  Very Hot  High School       0  0.445272
9   9     F     Y      Delhi      Warm         PhD       0  0.445272
```

[7]:
```
import pandas as pd
from scipy import stats
import numpy as np
df5=pd.read_csv("/content/Data_to_Transform.csv")
df5
```

[7]:
```
      Moderate Positive Skew  Highly Positive Skew  Moderate Negative Skew  \
0                   0.899990              2.895074               11.180748
1                   1.113554              2.962385               10.842938
2                   1.156830              2.966378               10.817934
3                   1.264131              3.000324               10.764570
4                   1.323914              3.012109               10.753117
…                          …                     …                       …
9995               14.749050             16.289513               -2.980821
```

```
9996              14.854474             16.396252              -3.147526
9997              15.262103             17.102991              -3.517256
9998              15.269983             17.628467              -4.689833
9999              16.204517             18.052331              -6.335679

      Highly Negative Skew
0                 9.027485
1                 9.009762
2                 9.006134
3                 9.000125
4                 8.981296
…                      …
9995             -3.254882
9996             -3.772332
9997             -4.717950
9998             -5.670496
9999             -7.036091

[10000 rows x 4 columns]
```

[26]:

```
df5.skew()
```

[26]:

```
Moderate Positive Skew    0.656308
Highly Positive Skew      1.271249
Moderate Negative Skew   -0.690244
Highly Negative Skew     -1.201891
dtype: float64
```

[8]:

```
np.log(df5["Highly Positive Skew"])
```

[8]:

```
0       1.063011
1       1.085995
2       1.087342
3       1.098720
4       1.102640
          …
9995    2.790522
9996    2.797053
9997    2.839253
9998    2.869515
9999    2.893275
Name: Highly Positive Skew, Length: 10000, dtype: float64
```

[9]:

```
np.reciprocal(df5["Moderate Positive Skew"])
```

[9]:

```
0       1.111123
1       0.898026
```

```
2       0.864431
3       0.791057
4       0.755336
          …
9995    0.067801
9996    0.067320
9997    0.065522
9998    0.065488
9999    0.061711
Name: Moderate Positive Skew, Length: 10000, dtype: float64
```

```
[10]: np.sqrt(df5["Highly Negative Skew"])
```

```
/usr/local/lib/python3.10/dist-packages/pandas/core/arraylike.py:396:
RuntimeWarning: invalid value encountered in sqrt
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

```
[10]: 0       3.004577
      1       3.001627
      2       3.001022
      3       3.000021
      4       2.996881
                …
      9995         NaN
      9996         NaN
      9997         NaN
      9998         NaN
      9999         NaN
      Name: Highly Negative Skew, Length: 10000, dtype: float64
```

```
[11]: np.square(df5["Highly Positive Skew"])
```

```
[11]: 0         8.381452
      1         8.775724
      2         8.799396
      3         9.001942
      4         9.072800
                   …
      9995    265.348230
      9996    268.837091
      9997    292.512290
      9998    310.762852
      9999    325.886637
      Name: Highly Positive Skew, Length: 10000, dtype: float64
```

```
[14]: df5["Highly positive skew_boxcox"],parameters=stats.boxcox(df5["Highly Positive␣
        ↪Skew"])
```

```
df5
```

```
[14]:       Moderate Positive Skew  Highly Positive Skew  Moderate Negative Skew  \
      0                   0.899990              2.895074               11.180748
      1                   1.113554              2.962385               10.842938
      2                   1.156830              2.966378               10.817934
      3                   1.264131              3.000324               10.764570
      4                   1.323914              3.012109               10.753117
      …                        …                   …                     …
      9995               14.749050             16.289513               -2.980821
      9996               14.854474             16.396252               -3.147526
      9997               15.262103             17.102991               -3.517256
      9998               15.269983             17.628467               -4.689833
      9999               16.204517             18.052331               -6.335679

            Highly Negative Skew  Highly positive skew_boxcox
      0                 9.027485                     0.812909
      1                 9.009762                     0.825921
      2                 9.006134                     0.826679
      3                 9.000125                     0.833058
      4                 8.981296                     0.835247
      …                      …                            …
      9995             -3.254882                     1.457701
      9996             -3.772332                     1.459189
      9997             -4.717950                     1.468681
      9998             -5.670496                     1.475357
      9999             -7.036091                     1.480525

      [10000 rows x 5 columns]
```

```
[15]: df5.skew()
```

```
[15]: Moderate Positive Skew         0.656308
      Highly Positive Skew           1.271249
      Moderate Negative Skew        -0.690244
      Highly Negative Skew          -1.201891
      Highly positive skew_boxcox    0.023089
      dtype: float64
```

```
[18]: df5["Highly Negative Skew_yoejhonson"],parameters=stats.yeojohnson(df5["Highly␣
       ↪Negative Skew"])
      df5
```

```
[18]:       Moderate Positive Skew  Highly Positive Skew  Moderate Negative Skew  \
      0                   0.899990              2.895074               11.180748
      1                   1.113554              2.962385               10.842938
      2                   1.156830              2.966378               10.817934
```

```
3                   1.264131              3.000324               10.764570
4                   1.323914              3.012109               10.753117
…                      …                   …                     …
9995               14.749050             16.289513               -2.980821
9996               14.854474             16.396252               -3.147526
9997               15.262103             17.102991               -3.517256
9998               15.269983             17.628467               -4.689833
9999               16.204517             18.052331               -6.335679

      Highly Negative Skew  Highly positive skew_boxcox  \
0                 9.027485                     0.812909
1                 9.009762                     0.825921
2                 9.006134                     0.826679
3                 9.000125                     0.833058
4                 8.981296                     0.835247
…                     …                          …
9995             -3.254882                     1.457701
9996             -3.772332                     1.459189
9997             -4.717950                     1.468681
9998             -5.670496                     1.475357
9999             -7.036091                     1.480525

      Highly Negative Skew_yoejhonson
0                           51.081488
1                           50.898043
2                           50.860532
3                           50.798434
4                           50.604086
…                               …
9995                        -1.433326
9996                        -1.545673
9997                        -1.722267
9998                        -1.872430
9999                        -2.053503

[10000 rows x 6 columns]
```

[20]:

```
df5.skew()
```

[20]:

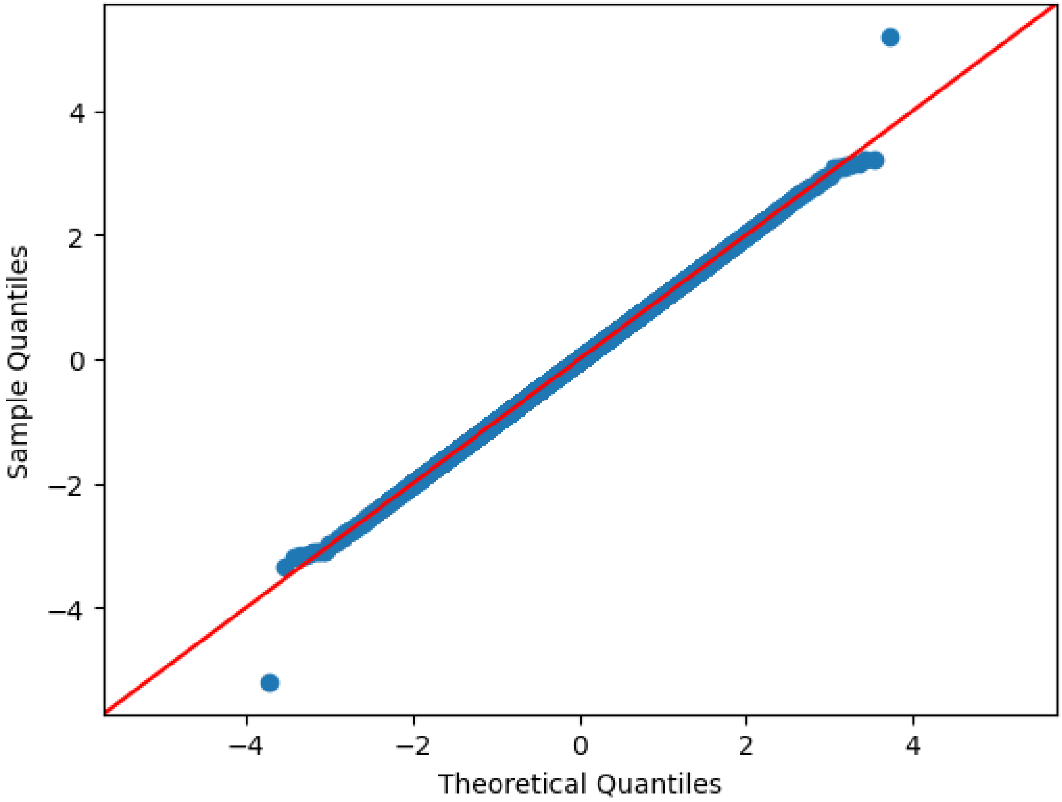
```
Moderate Positive Skew             0.656308
Highly Positive Skew               1.271249
Moderate Negative Skew            -0.690244
Highly Negative Skew              -1.201891
Highly positive skew_boxcox        0.023089
Highly Negative Skew_yoejhonson   -0.274676
dtype: float64
```

[22]:
```
from sklearn.preprocessing import QuantileTransformer
qt=QuantileTransformer(output_distribution="normal")
df5["Moderate Negative Skew"]=qt.fit_transform(df5[["Moderate Negative Skew"]])
df5
```

[22]:
```
      Moderate Positive Skew  Highly Positive Skew  Moderate Negative Skew  \
0                   0.899990              2.895074                5.199338
1                   1.113554              2.962385                3.227288
2                   1.156830              2.966378                3.206801
3                   1.264131              3.000324                3.167111
4                   1.323914              3.012109                3.159208
…                      …                   …                     …
9995               14.749050             16.289513               -3.147619
9996               14.854474             16.396252               -3.162489
9997               15.262103             17.102991               -3.198205
9998               15.269983             17.628467               -3.350199
9999               16.204517             18.052331               -5.199338

      Highly Negative Skew  Highly positive skew_boxcox  \
0                 9.027485                     0.812909
1                 9.009762                     0.825921
2                 9.006134                     0.826679
3                 9.000125                     0.833058
4                 8.981296                     0.835247
…                    …                          …
9995             -3.254882                     1.457701
9996             -3.772332                     1.459189
9997             -4.717950                     1.468681
9998             -5.670496                     1.475357
9999             -7.036091                     1.480525

      Highly Negative Skew_yoejhonson
0                           51.081488
1                           50.898043
2                           50.860532
3                           50.798434
4                           50.604086
…                               …
9995                        -1.433326
9996                        -1.545673
9997                        -1.722267
9998                        -1.872430
9999                        -2.053503

[10000 rows x 6 columns]
```

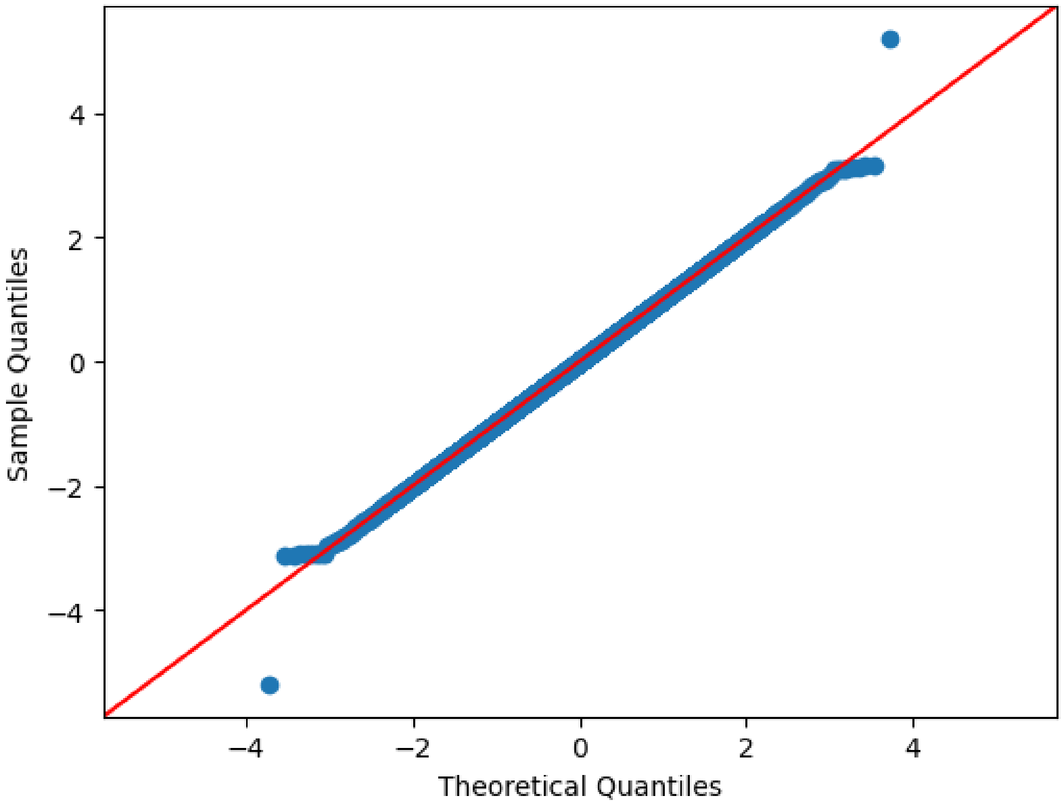
[23]:

```
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm
sm.qqplot(df5["Moderate Negative Skew"],line='45')
plt.show()
```



[28]:

```
df5["Highly Negative Skew"]=qt.fit_transform(df5[["Highly Negative Skew"]])
sm.qqplot(df5["Highly Negative Skew"],line='45')
plt.show()
```



```
[29]: df6=pd.read_csv("/content/titanic_dataset.csv")
      df6
```

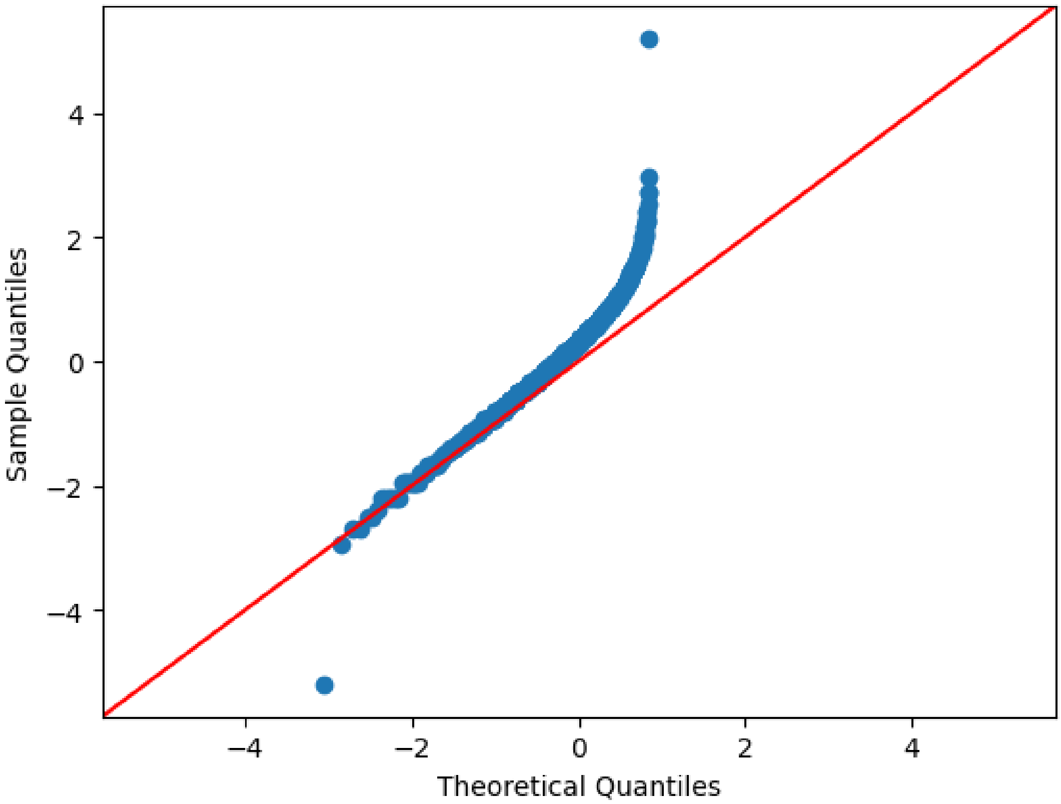
```
[29]:      PassengerId  Survived  Pclass  \
      0              1         0       3
      1              2         1       1
      2              3         1       3
      3              4         1       1
      4              5         0       3
      ..           …         …     …
      886          887         0       2
      887          888         1       1
      888          889         0       3
      889          890         1       1
      890          891         0       3

                                                        Name     Sex   Age  SibSp  \
      0                                  Braund, Mr. Owen Harris    male  22.0      1
      1    Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
      2                                   Heikkinen, Miss. Laina  female  26.0      0
      3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
```

```
4                             Allen, Mr. William Henry    male  35.0      0
..                                                 …     …   …    …
886                              Montvila, Rev. Juozas    male  27.0      0
887                       Graham, Miss. Margaret Edith  female  19.0      0
888           Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                              Behr, Mr. Karl Howell    male  26.0      0
890                                Dooley, Mr. Patrick    male  32.0      0

     Parch             Ticket     Fare Cabin Embarked
0        0          A/5 21171   7.2500   NaN        S
1        0           PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0             113803  53.1000  C123        S
4        0             373450   8.0500   NaN        S
..     …               …      …   …      …
886      0             211536  13.0000   NaN        S
887      0             112053  30.0000   B42        S
888      2         W./C. 6607  23.4500   NaN        S
889      0             111369  30.0000  C148        C
890      0             370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

[30]:

```
df6["Age"]=qt.fit_transform(df6[["Age"]])
sm.qqplot(df6["Age"],line='45')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/preprocessing/_data.py:2785:
UserWarning: n_quantiles (1000) is greater than the total number of samples
(891). n_quantiles is set to n_samples.
  warnings.warn(
```


4
2
0
−2
−4
Sample Quantiles
−4
−2
0
2
4
Theoretical Quantiles