```
# what is function transformer in machine learning.
```

```
# the function transformer is a tool in scikit-learn,
  # a popular python library for machine learning,
 # that allows you to apply a specified function to the input data .
 #  the function transformer can be useful for
  # performing custom transformations of input data in a machine learning pipeline.
```

```
from sklearn.preprocessing import FunctionTransformer
import numpy as np

# create a dataset
x = np.array([[1, 2], [3, 4]])

# define the transformation function.
log_transformer = FunctionTransformer(np.log1p)

# apply the transformation to the dataset.
x_transformer= log_transformer.transform(x)

# print the transformed dataset.
print(x_transformer)
```

```
[[0.69314718 1.09861229]
 [1.38629436 1.60943791]]
```

```
# 1.custom feature engineering                                          manually define

from sklearn.preprocessing import FunctionTransformer
import numpy as np

# create a dataset
x = np.array([[1, 2], [3, 4]])

# define a custom feature engineering function
def squ(x):
  return np.hstack((x,x**2))

# create a functiontransformer to apply the custom function.
custom_transformer = FunctionTransformer(squ)

# apply the transformer to the input data.
x_transformer = custom_transformer.transform(x)

# view the transformed data.
print(x_transformer)
```

```
[[ 1  2  1  4]
 [ 3  4  9 16]]
```

```
from sklearn.preprocessing import FunctionTransformer
import numpy as np

# create a dataset
x = np.array([[1, 2], [3, 4]])

# define a custom scaling function
def my_scaling(x):
  return x/ np.max(x)

# create a functiontransformer to apply the custom function
custom_transformer = FunctionTransformer(my_scaling)

# apply the transformer to the input data
x_transformed = custom_transformer.transform(x)

# view the transformed data
print(x_transformed)
```

```
[[0.25 0.5 ]
 [0.75 1.  ]]
```

```
from sklearn.preprocessing import FunctionTransformer
import numpy as np
```

```
# create a dataset with missing values
x = np.array([[1, 2], [3, np.nan]])

# define a custom cleaning function
def my_cleaning(x):
  x[np.isnan(x)] = 0
  return x

# create a functiontransformer to apply the custom function
custom_transformer = FunctionTransformer(my_cleaning)

# apply the transformer to the input data
x_transformed = custom_transformer.transform(x)

# view the transformed data
print(x_transformed)
```

```
[[1. 2.]
 [3. 0.]]
```

```
import pandas as pd
import numpy as np
```

```
df= pd.read_csv("/content/placement - placement dd.csv")
df.head(2)
```

	cgpa	resume_score	placed
0	8.14	6.52	1
1	6.17	5.17	0

```
x= df.drop(columns=["placed"])
y= df["placed"]
```

```
from sklearn.preprocessing import FunctionTransformer
```

Double-click (or enter) to edit

```
log_transformer = FunctionTransformer(np.log1p)

# apply the transformation to the dataset
x_transformed = log_transformer.transform(x)
```

```
x_transformed
```

	cgpa	resume_score
0	2.212660	2.017566
1	1.969906	1.819699
2	2.226783	2.288486
3	2.064328	2.112635
4	2.142416	2.116256
...	...	...
95	1.991976	1.998774
96	2.222459	2.170196
97	2.034706	2.172476
98	2.212660	1.891605
99	1.958685	2.029463

100 rows × 2 columns

```
df=pd.read_csv("/content/insurance - insurance.csv")
df.head(2)
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.90	0	yes	southwest	16884.9240
1	18	male	33.77	1	no	southeast	1725.5523

```
from sklearn.preprocessing import LabelEncoder
```

```
lb= LabelEncoder()
```

```
x= df.drop(columns=["region"])
y= df["region"]
```

```
x['sex']= lb.fit_transform(x['sex'])
x['smoker']= lb.fit_transform(x['smoker'])
```

```
log_transformer = FunctionTransformer(np.log1p)

x_transformed = log_transformer.transform(x)
```

```
x_transformed
```

	age	sex	bmi	children	smoker	charges
0	2.995732	0.000000	3.363842	0.000000	0.693147	9.734236
1	2.944439	0.693147	3.548755	0.693147	0.000000	7.453882
2	3.367296	0.693147	3.526361	1.386294	0.000000	8.400763
3	3.526361	0.693147	3.165686	0.000000	0.000000	9.998137
4	3.496508	0.693147	3.397189	0.000000	0.000000	8.260455
...	...	...	...	...	...	...
1333	3.931826	0.693147	3.464798	1.386294	0.000000	9.268755
1334	2.944439	0.000000	3.494080	0.000000	0.000000	7.699381
1335	2.944439	0.000000	3.633631	0.000000	0.000000	7.396847
1336	3.091042	0.000000	3.288402	0.000000	0.000000	7.605365
1337	4.127134	0.000000	3.403528	0.000000	0.693147	10.279948

1338 rows × 6 columns

```
import pandas as pd
import numpy as np
```

```
df=pd.read_csv("/content/cars - cars.......csv")
df.head(2)
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC BSII	1992	50000	100000	Petrol	Individual	Manual	Fourth & Above Owner
1	Maruti Gypsy E MG410W ST	1995	95000	100000	Petrol	Individual	Manual	Second Owner

```
x = df.drop(columns=['owner'])
y = df[['owner']]
```

```
from sklearn.preprocessing import LabelEncoder
```

```
lb=LabelEncoder()
```

```
x['fuel']= lb.fit_transform(x['fuel'])
x['seller_type']= lb.fit_transform(x['seller_type'])
x['transmission']= lb.fit_transform(x['transmission'])
x['name']= lb.fit_transform(x['name'])
```

```
from sklearn.preprocessing import FunctionTransformer
```

```
log_transformer = FunctionTransformer(np.log1p)

# apply the transformation to the dataset
x_transformed = log_transformer.transform(x)
```

```
x_transformed
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission
0	6.655440	7.597396	10.819798	11.512935	1.609438	0.693147	0.693147
1	6.804615	7.598900	11.461643	11.512935	1.609438	0.693147	0.693147
2	6.473891	7.599401	12.429220	10.463132	0.693147	0.693147	0.693147
3	6.476972	7.599401	12.206078	11.002117	0.693147	0.693147	0.693147
4	6.473891	7.599902	11.918397	11.695255	0.693147	0.693147	0.693147
...	...	...	...	...	...	...	...
4335	6.210600	7.611348	13.864302	7.003974	1.609438	0.693147	0.693147
4336	6.077642	7.611348	13.208543	8.517393	1.609438	0.693147	0.693147
4337	5.416100	7.611348	13.180634	10.714440	0.693147	0.000000	0.693147
4338	7.098376	7.611348	12.962197	6.908755	0.693147	0.693147	0.693147
4339	5.590987	7.611348	13.327752	6.908755	1.609438	0.693147	0.693147

4340 rows × 7 columns

```
import numpy as np
import pandas as pd
```

```
df=pd.read_csv('/content/dsjob - dsjob1.csv')
df.head(2)
```

	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_university	education_level	major_di
0	32403	city_41	0.827	Male	Has relevent experience	Full time course	Graduate	
1	9858	city_103	0.920	Female	Has relevent experience	no_enrollment	Graduate	

```
x=df.drop(columns=['relevent_experience'])
y=df['relevent_experience']
```

```
from sklearn.preprocessing import LabelEncoder
```

```
lb=LabelEncoder()
```

```
x['city']=lb.fit_transform(x['city'])
x['gender']=lb.fit_transform(x['gender'])
x['company_type']=lb.fit_transform(x['company_type'])
x['major_discipline']=lb.fit_transform(x['major_discipline'])
x['education_level']=lb.fit_transform(x['education_level'])
x['enrolled_university']=lb.fit_transform(x['enrolled_university'])
```

```
from sklearn.preprocessing import FunctionTransformer
```

```
log_transformer = FunctionTransformer(np.log1p)

# apply the transformation to the dataset
x_transformed = log_transformer.transform(x)
```

```
x_transformed
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission
0	6.655440	7.597396	10.819798	11.512935	1.609438	0.693147	0.693147
1	6.804615	7.598900	11.461643	11.512935	1.609438	0.693147	0.693147
2	6.473891	7.599401	12.429220	10.463132	0.693147	0.693147	0.693147
3	6.476972	7.599401	12.206078	11.002117	0.693147	0.693147	0.693147
4	6.473891	7.599902	11.918397	11.695255	0.693147	0.693147	0.693147
...	...	...	...	...	...	...	...
4335	6.210600	7.611348	13.864302	7.003974	1.609438	0.693147	0.693147
4336	6.077642	7.611348	13.208543	8.517393	1.609438	0.693147	0.693147
4337	5.416100	7.611348	13.180634	10.714440	0.693147	0.000000	0.693147
4338	7.098376	7.611348	12.962197	6.908755	0.693147	0.693147	0.693147
4339	5.590987	7.611348	13.327752	6.908755	1.609438	0.693147	0.693147

4340 rows × 7 columns