```
# columnstransformer
```

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

Add text cell

```
df=pd.read_csv('/content/supply_chain - supply_chain.d.csv')
df.head(2)
```

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	Order quantities	...	Location	Lea tim
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7	96	...	Mumbai	2
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30	37	...	Mumbai	2

2 rows × 24 columns

```
df['SKU'].value_counts()
```

SKU	count
SKU0	1
SKU1	1
SKU2	1
SKU3	1
SKU4	1
...	...
SKU95	1
SKU96	1
SKU97	1
SKU98	1
SKU99	1

100 rows × 1 columns

dtype: int64

```
df.isnull().sum()
```

	0
Product type	0
SKU	0
Price	Add text cell
Availability	0
Number of products sold	0
Revenue generated	0
Customer demographics	0
Stock levels	0
Lead times	0
Order quantities	0
Shipping times	0
Shipping carriers	0
Shipping costs	0
Supplier name	0
Location	0
Lead time	0
Production volumes	0
Manufacturing lead time	0
Manufacturing costs	0
Inspection results	0
Defect rates	0
Transportation modes	0
Routes	0
Costs	0

dtype: int64

```
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import OrdinalEncoder
```

```
from sklearn.compose import ColumnTransformer
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
```

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

```
x.head(5)
```

	Product type	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	Order quantities	Shipping times	...	Location
2	haircare	11.319683	34	8	9577.749626	Unknown	1	10	88	2	...	Mumbai
4	skincare	4.805496	26	871	2686.505152	Non-binary	5	5	56	8	...	Delhi

5 rows × 23 columns

```
si=SimpleImputer(strategy='mean')                    # that is not use this data not missing values
x_train_price=si.fit_transform(x_train[['Price']])
```

Add text cell

```
x_test_price=si.fit_transform(x_test[['Price']])
x_train_price.shape
```

```
(80, 1)
```

```
oe=OrdinalEncoder(categories=[['haircare','skincare','cosmetics']])
x_train_Product_type=oe.fit_transform(x_train[['Product type']])
x_test_Product_type=oe.fit_transform(x_test[['Product type']])
x_train_Product_type.shape
```

```
(80, 1)
```

```
one=OneHotEncoder(sparse_output=False,drop='first')
x_train_Gender=one.fit_transform(x_train[['Price','Customer demographics','Location','Inspection results','Transportation m
x_test_Gender=one.fit_transform(x_test[['Price','Customer demographics','Location','Inspection results','Transportation mod
x_train_Gender.shape
```

```
(80, 93)
```

```
x_train_Price=x_train.drop(columns=['Price','Customer demographics','Location','Inspection results','Transportation modes',
x_test_Price=x_train.drop(columns=['Price','Customer demographics','Location','Inspection results','Transportation modes',
```

```
x_train_price.shape
```

```
(80, 1)
```

```
x_train_transformed = np.concatenate((x_train_price, x_train_Product_type, x_train_Gender, x_train_Price.values), axis=1)
```

```
x_train_transformed.shape
```

```
(80, 112)
```

```
from sklearn.compose import ColumnTransformer
```

```
transformer=ColumnTransformer(transformers=[
    ('tnf1',SimpleImputer(strategy='mean'),['Price']),
    ('tnf2',OrdinalEncoder(categories=[['haircare','skincare','cosmetics']]),['Product type']),
    ('tnf3',OneHotEncoder(sparse_output=False,drop='first'),['Customer demographics','Location','Inspection results','Trans
], remainder='passthrough')
```

```
transformer.fit_transform(x_train).shape
```

```
(80, 32)
```