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

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

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	GT86	2016	16000	Manual	24089	Petrol	265	36.2	2.0
1	GT86	2017	15995	Manual	18615	Petrol	145	36.2	2.0

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

model	count
Yaris	2122
Aygo	1961
Auris	712
C-HR	479
RAV4	473
Corolla	267
Prius	232
Avensis	115
Verso	114
Hilux	86
GT86	73
Land Cruiser	51
PROACE VERSO	15
Supra	12
Camry	11
IQ	8
Urban Cruiser	4
Verso-S	3

dtype: int64

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

transmission	count
Manual	3826
Automatic	2657
Semi-Auto	254
Other	1

dtype: int64

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

fuelType	count
Petrol	4087
Hybrid	2043
Diesel	503
Other	105

dtype: int64

```
df.drop(columns=['fuelType'])
```

	model	year	price	transmission	mileage	tax	mpg	engineSize
0	GT86	2016	16000	Manual	24089	265	36.2	2.0
1	GT86	2017	15995	Manual	18615	145	36.2	2.0
2	GT86	2015	13998	Manual	27469	265	36.2	2.0
3	GT86	2017	18998	Manual	14736	150	36.2	2.0
4	GT86	2017	17498	Manual	36284	145	36.2	2.0
...	...	...	...	...	...	...	...	...
6733	IQ	2011	5500	Automatic	30000	20	58.9	1.0
6734	Urban Cruiser	2011	4985	Manual	36154	125	50.4	1.3
6735	Urban Cruiser	2012	4995	Manual	46000	125	57.6	1.4
6736	Urban Cruiser	2011	3995	Manual	60700	125	50.4	1.3
6737	Urban Cruiser	2011	4495	Manual	45128	125	50.4	1.3

6738 rows × 8 columns

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

```
x.head(2)
```

	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	2016	16000	Manual	24089	Petrol	265	36.2	2.0
1	2017	15995	Manual	18615	Petrol	145	36.2	2.0

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

	0
model	0
year	0
price	0
transmission	0
mileage	0
fuelType	0
tax	0
mpg	0
engineSize	0

dtype: int64

```
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=42)
```

```
np.round(x_train.describe(),2)
```

	year	price	mileage	tax	mpg	engineSize
count	5390.00	5390.00	5390.00	5390.0	5390.00	5390.00
mean	2016.75	12467.88	22820.11	94.7	63.17	1.47
std	2.18	6298.40	19104.20	73.9	16.16	0.44
min	1998.00	850.00	2.00	0.0	2.80	0.00
25%	2016.00	8250.00	9500.00	0.0	55.40	1.00
50%	2017.00	10703.50	18489.00	135.0	62.80	1.50
75%	2018.00	14995.00	31057.00	145.0	69.00	1.80
max	2020.00	59995.00	174419.00	565.0	235.00	4.50

```
from sklearn.preprocessing import StandardScaler
```

```
sc = StandardScaler()
```

```
# Apply StandardScaler
x_train_sc = sc.fit_transform(x_train)
```

```
x_train_new=pd.DataFrame(x_train_sc,columns=x_train_cols)
```

```
np.round(x_train_new.describe(),2)
```

	year	price	mileage	tax	mpg	engineSize	transmission_Manual	transmission_Other	transmission_Semi-Auto	f
count	5390.00	5390.00	5390.00	5390.00	5390.00	5390.00	5390.00	5390.00	5390.00	
mean	-0.00	0.00	0.00	-0.00	0.00	-0.00	-0.00	0.00	-0.00	
std	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	
min	-8.58	-1.84	-1.19	-1.28	-3.74	-3.36	-1.16	-0.01	-0.19	
25%	-0.34	-0.67	-0.70	-1.28	-0.48	-1.07	-1.16	-0.01	-0.19	
50%	0.12	-0.28	-0.23	0.55	-0.02	0.07	0.86	-0.01	-0.19	
75%	0.57	0.40	0.43	0.68	0.36	0.76	0.86	-0.01	-0.19	
max	1.49	7.55	7.94	6.36	10.64	6.94	0.86	73.41	5.22	

```
# normalization---->
```

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

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

```
df['model']=df['model'].map({'Camry':1,'Corolla':2,'Yaris':3})
df['mileage']=df['mileage'].map({'24kmpl':1,'27kmpl':2,'32kmpl':3})
```

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

```
df=df.drop(columns=['fuelType'])
```

```
df=df.drop(columns=['transmission'])
```

```
df.head(2)
```

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

```
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=42)
```

```
                                                                    random_state 42)
```

```
np.round(x_train.describe(),2)
```

```
from sklearn.preprocessing import MinMaxScaler
```

```
mn=MinMaxScaler
```

```
x_train_mn=mn.fit_transform(x_train)
```

```
x_train_new=pd.DataFrame(x_train_mn,columns=x_train.columns)
```

```
np.round(x_train_new.describe(),2)
```