In [1]:

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

In [2]:

```python
#already data is standardized with z score
df=pd.read_csv("D:\python programs\Scaler classes\cars24-car-price-clean.csv")
```

In [3]:

```python
df
```

Out[3]:

	selling_price	year	km_driven	mileage	engine	max_power	age	make	model	Indi
0	-1.111046	-0.801317	1.195828	0.045745	-1.310754	-1.157780	0.801317	-0.433854	-1.125683	1.2
1	-0.223944	0.450030	-0.737872	-0.140402	-0.537456	-0.360203	-0.450030	-0.327501	-0.333227	1.2
2	-0.915058	-1.426990	0.035608	-0.582501	-0.537456	-0.404885	1.426990	-0.327501	-0.789807	1.2
3	-0.892365	-0.801317	-0.409143	0.329620	-0.921213	-0.693085	0.801317	-0.433854	-0.905265	1.2
4	-0.182683	0.137194	-0.544502	0.760085	0.042999	0.010435	-0.137194	-0.246579	-0.013096	-0.8
...	...	...	...	...	...	...	...	...	...	
19815	-0.017641	0.762867	0.218923	0.950886	-0.215410	-0.694202	-0.762867	0.724475	0.091865	-0.8
19816	0.549692	1.388540	-0.776546	-0.466159	-0.198054	-0.156899	-1.388540	-0.433854	0.112857	-0.8
19817	-0.481822	0.137194	0.170967	0.380810	0.042999	0.120577	-0.137194	-0.240799	-0.448684	-0.8
19818	1.168601	0.450030	72.355997	-0.815185	1.356256	0.935579	-0.450030	0.123346	0.328027	-0.8
19819	1.117025	1.388540	-0.873231	-0.349818	0.041071	0.435139	-1.388540	-0.177218	1.036514	-0.8

19820 rows × 18 columns

In [4]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19820 entries, 0 to 19819
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   selling_price     19820 non-null  float64
 1   year              19820 non-null  float64
 2   km_driven         19820 non-null  float64
 3   mileage           19820 non-null  float64
 4   engine            19820 non-null  float64
 5   max_power         19820 non-null  float64
 6   age               19820 non-null  float64
 7   make              19820 non-null  float64
 8   model             19820 non-null  float64
 9   Individual        19820 non-null  float64
 10  Trustmark Dealer  19820 non-null  float64
 11  Diesel            19820 non-null  float64
 12  Electric          19820 non-null  float64
 13  LPG               19820 non-null  float64
 14  Petrol            19820 non-null  float64
 15  Manual            19820 non-null  float64
 16  5                 19820 non-null  float64
 17  >5                19820 non-null  float64
```

```
dtypes: float64(18)
memory usage: 2.7 MB
```

In [5]:
```
# model:x1
# price:y
# y=f(x)

x=df['model'].values
x
```

Out[5]:
```
array([-1.12568266, -0.3332271 , -0.78980745, ..., -0.4486842 ,
        0.32802721,  1.03651397])
```

In [6]:
```
y=df['selling_price'].values
y
```

Out[6]:
```
array([-1.11104589, -0.22394353, -0.91505816, ..., -0.48182212,
        1.16860087,  1.11702515])
```

In [7]:
```
Model=LinearRegression()
```

In [8]:
```
type(Model)
```

Out[8]:
```
sklearn.linear_model._base.LinearRegression
```

In [9]:
```
x
```

Out[9]:
```
array([-1.12568266, -0.3332271 , -0.78980745, ..., -0.4486842 ,
        0.32802721,  1.03651397])
```

In [10]:
```
y
```

Out[10]:
```
array([-1.11104589, -0.22394353, -0.91505816, ..., -0.48182212,
        1.16860087,  1.11702515])
```

In [11]:
```
print(x.shape)
print(y.shape)
```

```
(19820,)
(19820,)
```

In [12]:
```
x=x.reshape(x.size,1)
y=y.reshape(y.size,1)
```

In [13]:
```
Model.fit(x,y)
```

Out[13]:
```
▾ LinearRegression
LinearRegression()
```

In [14]:
```
y_hat=Model.predict(x)
y_hat
```

Out[14]:
```
array([[-1.08634131],
       [-0.32158118],
       [-0.76220457],
       ...,
       [-0.43300319],
       [ 0.31656303],
       [ 1.00028896]])
```

In [15]:
```
y
```

Out[15]:
```
array([[-1.11104589],
       [-0.22394353],
       [-0.91505816],
       ...,
       [-0.48182212],
       [ 1.16860087],
       [ 1.11702515]])
```

In [16]:

```
Model.intercept_
```

Out[16]:

```
array([-1.18731936e-16])
```

In [17]:

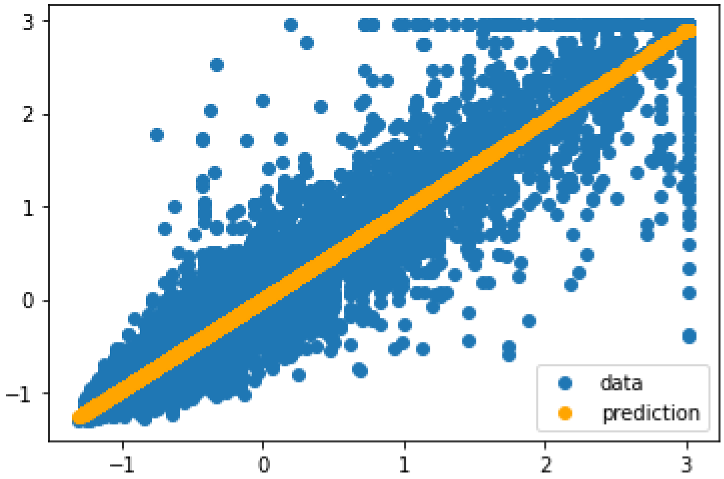
```
Model.coef_
```

Out[17]:

```
array([[0.96505112]])
```

In [18]:

```
# price=0.9650*model-1.1873   written as manually
fig=plt.figure()
plt.scatter(x,y,label='data')
plt.scatter(x,y_hat,color='orange',label='prediction')
plt.legend()
plt.show()
```



In [19]:

```
#R^2 coefficient of determination reliable method to compare two models quickly.
Model.score(x,y)
```

Out[19]:

```
0.9313236629576508
```

In [20]:

```
df.head()
```

Out[20]:

	selling_price	year	km_driven	mileage	engine	max_power	age	make	model	Individu
0	-1.111046	-0.801317	1.195828	0.045745	-1.310754	-1.157780	0.801317	-0.433854	-1.125683	1.24889
1	-0.223944	0.450030	-0.737872	-0.140402	-0.537456	-0.360203	-0.450030	-0.327501	-0.333227	1.24889
2	-0.915058	-1.426990	0.035608	-0.582501	-0.537456	-0.404885	1.426990	-0.327501	-0.789807	1.24889
3	-0.892365	-0.801317	-0.409143	0.329620	-0.921213	-0.693085	0.801317	-0.433854	-0.905265	1.24889
4	-0.182683	0.137194	-0.544502	0.760085	0.042999	0.010435	-0.137194	-0.246579	-0.013096	-0.80071

In [21]:

```
y=df['selling_price']
```

In [22]:

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

In [23]:

```
x.shape
```

```
Out[23]:  (19820, 17)
```

```
In [24]:  y.shape
```

```
Out[24]:  (19820,)
```

```
In [25]:  x=x.to_numpy()
          y=y.to_numpy()
```

```
In [26]:  y=y.reshape(y.size,1)
```

```
In [27]:  Model.fit(x,y)
```

```
Out[27]:  ▾ LinearRegression
          LinearRegression()
```

```
In [28]:  y_hat=Model.predict(x)
          y_hat
```

```
Out[28]:  array([[-1.18348944],
                 [-0.29402792],
                 [-0.89529669],
                 ...,
                 [-0.32373367],
                 [-0.86281723],
                 [ 1.04516761]])
```

```
In [29]:  print(Model.coef_)
```

```
[[ 2.63965047e+10 -1.82342529e-02 -4.86898422e-02  3.13931704e-02
   2.73611546e-02  2.63965047e+10  6.07826114e-02  7.94148445e-01
  -1.69569254e-02 -3.47542763e-03  1.40804052e-02  1.27513111e-02
   2.83169746e-03 -2.25827694e-02 -1.29337311e-02 -2.46414095e-02
  -3.26581001e-02]]
```

```
In [30]:  print(Model.intercept_)
```

```
[3.55622928e-05]
```

```
In [31]:  Model.score(x, y)
```

```
Out[31]:  0.9421886817589384
```

```
In [32]:
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [32], in <cell line: 1>()
----> 1 pyppeteer-install

NameError: name 'pyppeteer' is not defined
```

```
In [ ]:
```