In [1]:
```
import pandas as pd
import numpy as np
from sklearn import linear_model
import matplotlib.pyplot as plt
```

In [3]:
```
df = pd.read_csv('Home_price.csv')
df
```

Out[3]:

	area	Price
0	2600	550000
1	3000	565000
2	3200	610000
3	3600	680000
4	4000	725000

In [25]:
```
X = df.drop('Price',axis='columns')
X.shape
```

Out[25]:
```
(5, 1)
```

In [32]:
```
X
```

Out[32]:
```
0    2600
1    3000
2    3200
3    3600
4    4000
Name: area, dtype: int64
```

In [26]:
```
Y = df.Price
Y.shape
```

Out[26]:
```
(5,)
```

In [33]:
```
Y
```

Out[33]:
```
0    550000
1    565000
2    610000
3    680000
4    725000
Name: Price, dtype: int64
```

In [27]:
```
# Create linear regression object
reg = linear_model.LinearRegression()
reg.fit(new_df,price)
```

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

In [28]:
```
reg.predict([[3300]])
```

Out[28]:
```
array([628715.75342466])
```

In [10]:
```
#if you try like this it will give you errors
```

In [29]:
```
X=df['area']
X.shape
```

Out[29]:
```
(5,)
```

In [30]:
```
X
```

Out[30]:
```
0    2600
1    3000
2    3200
3    3600
4    4000
Name: area, dtype: int64
```

In [24]:
```
Y=df['Price']
Y.shape
```

Out[24]:
```
(5,)
```

In [31]:
```
Y
```

Out[31]:
```
0    550000
1    565000
2    610000
3    680000
4    725000
Name: Price, dtype: int64
```

In [ ]:
```

```

In [34]:
```
reg = linear_model.LinearRegression()
reg.fit(X,Y)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\SIKAND~1\AppData\Local\Temp/ipykernel_13396/2329011747.py in <module>
      1 reg = linear_model.LinearRegression()
----> 2 reg.fit(X,Y)

~\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in fit(self, X, y,
 sample_weight)
    516         accept_sparse = False if self.positive else ['csr', 'csc', 'co
o']
    517
--> 518         X, y = self._validate_data(X, y, accept_sparse=accept_sparse,
    519                                    y_numeric=True, multi_output=True)
    520

~\Anaconda3\lib\site-packages\sklearn\base.py in _validate_data(self, X, y, res
et, validate_separately, **check_params)
    431                 y = check_array(y, **check_y_params)
    432             else:
--> 433                 X, y = check_X_y(X, y, **check_params)
    434             out = X, y
    435

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **k
wargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_X_y(X, y, ac
cept_sparse, accept_large_sparse, dtype, order, copy, force_all_finite, ensure_
2d, allow_nd, multi_output, ensure_min_samples, ensure_min_features, y_numeric,
estimator)
    869         raise ValueError("y cannot be None")
    870
--> 871     X = check_array(X, accept_sparse=accept_sparse,
    872                     accept_large_sparse=accept_large_sparse,
    873                     dtype=dtype, order=order, copy=copy,

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **k
wargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_array(array,
accept_sparse, accept_large_sparse, dtype, order, copy, force_all_finite, ensur
e_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator)
    692             # If input is 1D raise error
    693             if array.ndim == 1:
```

```
--> 694                 raise ValueError(
    695                     "Expected 2D array, got 1D array instead:\narray=
{}.\n"
    696                     "Reshape your data either using array.reshape(-1,
 1) if "

ValueError: Expected 2D array, got 1D array instead:
array=[2600 3000 3200 3600 4000].
Reshape your data either using array.reshape(-1, 1) if your data has a single f
eature or array.reshape(1, -1) if it contains a single sample.
```

In [ ]: