

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
%matplotlib inline
```

```
from sklearn.datasets import fetch_openml
housing = fetch_openml(name="house_prices", as_frame=True)
housing.keys()
```

C:\Users\Lenovo\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\datasets_openml.py:968: FutureWarning: The default value of `parser` will change from `liac-arff` to `auto` in 1.4. You can set `parser='auto'` to silence this warning. Therefore, an `ImportError` will be raised from 1.4 if the dataset is dense and pandas is not installed. Note that the pandas parser may return different data types. See the Notes Section in fetch_openml's API doc for details.

```
warn(
dict_keys(['data', 'target', 'frame', 'categories', 'feature_names', 'target_names', 'DESCR', 'details', 'url'])
```

```
import pandas as pd
data = pd.DataFrame(housing.data, columns=housing.feature_names)
data.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold
0	1	60	RL	65.0	8450	Pave	None	Reg	Lvl	AllPub	...	0	0	None	None	None	0	2
1	2	20	RL	80.0	9600	Pave	None	Reg	Lvl	AllPub	...	0	0	None	None	None	0	5
2	3	60	RL	68.0	11250	Pave	None	IR1	Lvl	AllPub	...	0	0	None	None	None	0	9
3	4	70	RL	60.0	9550	Pave	None	IR1	Lvl	AllPub	...	0	0	None	None	None	0	2
4	5	60	RL	84.0	14260	Pave	None	IR1	Lvl	AllPub	...	0	0	None	None	None	0	12

5 rows × 80 columns

```
data.keys()
```

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition'],
      dtype='object')
```

```
data['Price'] = housing.target
data.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	Sale
0	1	60	RL	65.0	8450	Pave	None	Reg	Lvl	AllPub	...	0	None	None	None	0	2	2008	
1	2	20	RL	80.0	9600	Pave	None	Reg	Lvl	AllPub	...	0	None	None	None	0	5	2007	
2	3	60	RL	68.0	11250	Pave	None	IR1	Lvl	AllPub	...	0	None	None	None	0	9	2008	
3	4	70	RL	60.0	9550	Pave	None	IR1	Lvl	AllPub	...	0	None	None	None	0	2	2006	
4	5	60	RL	84.0	14260	Pave	None	IR1	Lvl	AllPub	...	0	None	None	None	0	12	2008	

5 rows × 81 columns

.

```
data.describe()
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	WoodDeckSF	OpenPorchSF
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1452.000000	1460.000000	...	1460.000000	1460.000000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342	1971.267808	1984.865753	103.685262	443.639726	...	94.244521	46.660274
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799	30.202904	20.645407	181.066207	456.098091	...	125.338794	66.256028
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000	1950.000000	0.000000	0.000000	...	0.000000	0.000000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000	1954.000000	1967.000000	0.000000	0.000000	...	0.000000	0.000000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000	1973.000000	1994.000000	0.000000	383.500000	...	0.000000	25.000000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000	2000.000000	2004.000000	166.000000	712.250000	...	168.000000	68.000000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.000000	...	857.000000	547.000000

8 rows × 38 columns

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 1460 entries, 0 to 1459
```

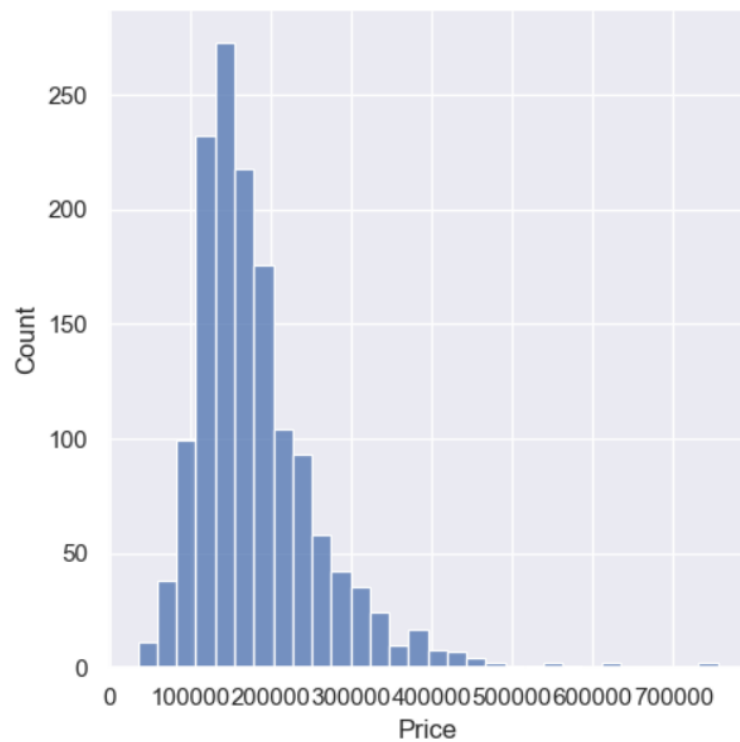
```
Data columns (total 81 columns):
```

#	Column	Non-Null Count	Dtype
0	Id	1460 non-null	int64
1	MSSubClass	1460 non-null	int64
2	MSZoning	1460 non-null	object
3	LotFrontage	1201 non-null	float64
4	LotArea	1460 non-null	int64
5	Street	1460 non-null	object
6	Alley	91 non-null	object
7	LotShape	1460 non-null	object
8	LandContour	1460 non-null	object
9	Utilities	1460 non-null	object
10	LotConfig	1460 non-null	object
11	LandSlope	1460 non-null	object
12	Neighborhood	1460 non-null	object
13	Condition1	1460 non-null	object
14	Condition2	1460 non-null	object
15	BldgType	1460 non-null	object
16	HouseStyle	1460 non-null	object
17	OverallQual	1460 non-null	int64
18	OverallCond	1460 non-null	int64
19	YearBuilt	1460 non-null	int64
20	YearRemodAdd	1460 non-null	int64
21	RoofStyle	1460 non-null	object
22	RoofMatl	1460 non-null	object
23	Exterior1st	1460 non-null	object
24	Exterior2nd	1460 non-null	object
25	MasVnrType	1452 non-null	object
26	MasVnrArea	1452 non-null	float64
27	ExterQual	1460 non-null	object
28	ExterCond	1460 non-null	object
29	Foundation	1460 non-null	object
30	BsmtQual	1423 non-null	object
31	BsmtCond	1423 non-null	object
32	BsmtExposure	1422 non-null	object
33	BsmtFinType1	1423 non-null	object
34	BsmtFinSF1	1460 non-null	int64
35	BsmtFinType2	1422 non-null	object
36	BsmtFinSF2	1460 non-null	int64

37	BsmtUnfSF	1460	non-null	int64
38	TotalBsmtSF	1460	non-null	int64
39	Heating	1460	non-null	object
40	HeatingQC	1460	non-null	object
41	CentralAir	1460	non-null	object
42	Electrical	1459	non-null	object
43	1stFlrSF	1460	non-null	int64
44	2ndFlrSF	1460	non-null	int64
45	LowQualFinSF	1460	non-null	int64
46	GrLivArea	1460	non-null	int64
47	BsmtFullBath	1460	non-null	int64
48	BsmtHalfBath	1460	non-null	int64
49	FullBath	1460	non-null	int64
50	HalfBath	1460	non-null	int64
51	BedroomAbvGr	1460	non-null	int64
52	KitchenAbvGr	1460	non-null	int64
53	KitchenQual	1460	non-null	object
54	TotRmsAbvGrd	1460	non-null	int64
55	Functional	1460	non-null	object
56	Fireplaces	1460	non-null	int64
57	FireplaceQu	770	non-null	object
58	GarageType	1379	non-null	object
58	GarageType	1379	non-null	object
59	GarageYrBlt	1379	non-null	float64
60	GarageFinish	1379	non-null	object
61	GarageCars	1460	non-null	int64
62	GarageArea	1460	non-null	int64
63	GarageQual	1379	non-null	object
64	GarageCond	1379	non-null	object
65	PavedDrive	1460	non-null	object
66	WoodDeckSF	1460	non-null	int64
67	OpenPorchSF	1460	non-null	int64
68	EnclosedPorch	1460	non-null	int64
69	3SsnPorch	1460	non-null	int64
70	ScreenPorch	1460	non-null	int64
71	PoolArea	1460	non-null	int64
72	PoolQC	7	non-null	object
73	Fence	281	non-null	object
74	MiscFeature	54	non-null	object
75	MiscVal	1460	non-null	int64
76	MoSold	1460	non-null	int64
77	YrSold	1460	non-null	int64
78	SaleType	1460	non-null	object
79	SaleCondition	1460	non-null	object
80	Price	1460	non-null	int64

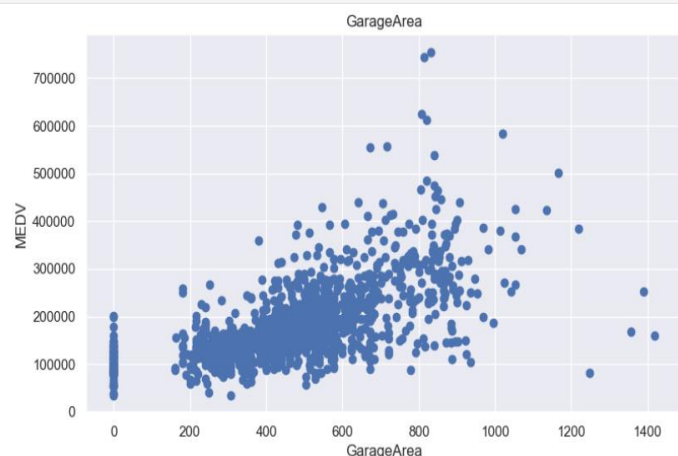
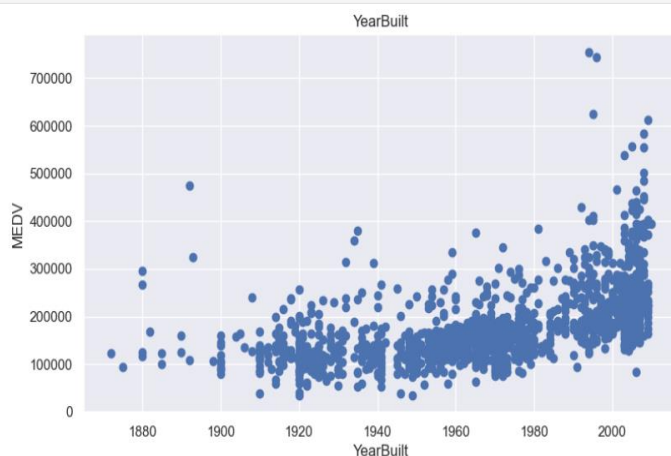
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB

```
sns.set(rc={'figure.figsize':(11.7,8.27)})
sns.displot(data['Price'], bins=30)
plt.show()
```



```
correlation_matrix= data.corr().round(2)
sns.heatmap(data=correlation_matrix,annot=True);
```

```
plt.figure(figsize=(20,5))
features = ['YearBuilt', 'GarageArea']
target=data['Price']
for i,col in enumerate(features):
    plt.subplot(1,len(features),i+1)
    x=data[col]
    y=target
    plt.scatter(x,y,marker='o')
    plt.title(col)
    plt.xlabel(col)
    plt.ylabel('MEDV')
```



```
import numpy as np
x = pd.DataFrame(np.c_[data['YearBuilt'],data['GarageArea']],columns=['YearBuilt', 'GarageArea'])
y = data['Price']
```

```
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=5)
print(X_train.shape,X_test.shape,Y_train.shape,Y_test.shape,sep='\n')
```

```
(1168, 2)
(292, 2)
(1168,)
(292,)
```

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(X_train,Y_train)
```

LinearRegression()

```
from sklearn.metrics import mean_squared_error,r2_score
```

```
y_pred= model.predict(X_test)
rmse=(np.sqrt(mean_squared_error(Y_test,y_pred)))
r2 = r2_score(Y_test,y_pred)
print('Model Preformance',f'RMSE is {rmse}',f'R2 score is {r2}',sep='\n')
```

Model Preformance

RMSE is 59404.92746939876

R2 score is 0.45624933440937054

```
sample_data=[[1990,9000]]
price=model.predict(sample_data)
print(price[0])
```

1700043.8762959081