In [295]:
```
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import mglearn
%matplotlib inline
import seaborn as sns
import platform
from matplotlib import font_manager , rc

if platform.system() == 'Darwin':
  rc('font' , family = 'AppleGothic')
elif platform.system() == 'Windows':
  path = 'C:/Windows/Fonts/malgun.ttf'
  font_name = font_manager.FontProperties(fname = path).get_name()
  rc('font' , family = font_name)
else:
  print('모름')
plt.rcParams['axes.unicode_minus'] = False
import warnings
warnings.filterwarnings('ignore')
from sklearn.metrics import accuracy_score , precision_score , recall_score , r
```
executed in 21ms, finished 16:47:26 2023-11-01

In [296]:
```
path = 'C:/k_digital/machine/source/house-prices-advanced-regression-techniques'
```
executed in 14ms, finished 16:47:26 2023-11-01

In [297]:
```
house = pd.read_csv(path + '/train.csv')
test = pd.read_csv(path + '/test.csv')
pred = pd.read_csv(path + '/sample_submission.csv')
```
executed in 45ms, finished 16:47:26 2023-11-01

In [298]:
```
house
```

executed in 29ms, finished 16:47:26 2023-11-01

Out[298]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandC
0	1	60	RL	65.0	8450	Pave	NaN	Reg	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	
...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	

1460 rows × 81 columns

In [299]:
```
house.info()
```
executed in 30ms, finished 16:47:26 2023-11-01

```
<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
 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  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

In [300]:
```
house.MSZoning.unique()
```
executed in 13ms, finished 16:47:26 2023-11-01

Out[300]:
```
array(['RL', 'RM', 'C (all)', 'FV', 'RH'], dtype=object)
```

In [301]:
```
house.corr()[house.corr()['SalePrice']>0.3]['SalePrice'].index
```
executed in 30ms, finished 16:47:26 2023-11-01

Out[301]:
```
Index(['LotFrontage', 'OverallQual', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea',
       'BsmtFinSF1', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'GrLivArea',
       'FullBath', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'GarageCars',
       'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'SalePrice'],
      dtype='object')
```

In [302]:
```
house[['LotFrontage', 'OverallQual', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea',
       'BsmtFinSF1', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'GrLivArea',
       'FullBath', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'GarageCars',
       'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'SalePrice']]
```
executed in 33ms, finished 16:47:26 2023-11-01

Out[302]:

	LotFrontage	OverallQual	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	Totall
0	65.0	7	2003	2003	196.0	706	
1	80.0	6	1976	1976	0.0	978	
2	68.0	7	2001	2002	162.0	486	
3	60.0	7	1915	1970	0.0	216	
4	84.0	8	2000	2000	350.0	655	
...	...	...	...	...	...	...	
1455	62.0	6	1999	2000	0.0	0	
1456	85.0	6	1978	1988	119.0	790	
1457	66.0	7	1941	2006	0.0	275	
1458	68.0	5	1950	1996	0.0	49	
1459	75.0	5	1965	1965	0.0	830	

1460 rows × 19 columns

In [303]:
```
house.drop('Id' , axis = 1 , inplace = True)
```
executed in 11ms, finished 16:47:26 2023-11-01

In [304]:
```
house.describe().columns
```
executed in 56ms, finished 16:47:26 2023-11-01

Out[304]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

LowQualFinSF -> 저품질 마감된 구역이니까 , 이건 음수로 바꾸자

In [305]:
```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
```
executed in 13ms, finished 16:47:26 2023-11-01

In [306]:
```
mm = MinMaxScaler()
```
executed in 14ms, finished 16:47:26 2023-11-01

In [307]:
```
house1 = house[['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCo
        'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
        'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
        'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
        'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
        'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
        'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
        'MoSold', 'YrSold', 'SalePrice']]
```
executed in 14ms, finished 16:47:26 2023-11-01

In [308]:
```
house1.isna().sum()
```
executed in 14ms, finished 16:47:26 2023-11-01

Out[308]:
```
MSSubClass         0
LotFrontage      259
LotArea            0
OverallQual        0
OverallCond        0
YearBuilt          0
YearRemodAdd       0
MasVnrArea         8
BsmtFinSF1         0
BsmtFinSF2         0
BsmtUnfSF          0
TotalBsmtSF        0
1stFlrSF           0
2ndFlrSF           0
LowQualFinSF       0
GrLivArea          0
BsmtFullBath       0
BsmtHalfBath       0
FullBath           0
HalfBath           0
BedroomAbvGr       0
KitchenAbvGr       0
TotRmsAbvGrd       0
Fireplaces         0
GarageYrBlt       81
GarageCars         0
GarageArea         0
WoodDeckSF         0
OpenPorchSF        0
EnclosedPorch      0
3SsnPorch          0
ScreenPorch        0
PoolArea           0
MiscVal            0
MoSold             0
YrSold             0
SalePrice          0
dtype: int64
```

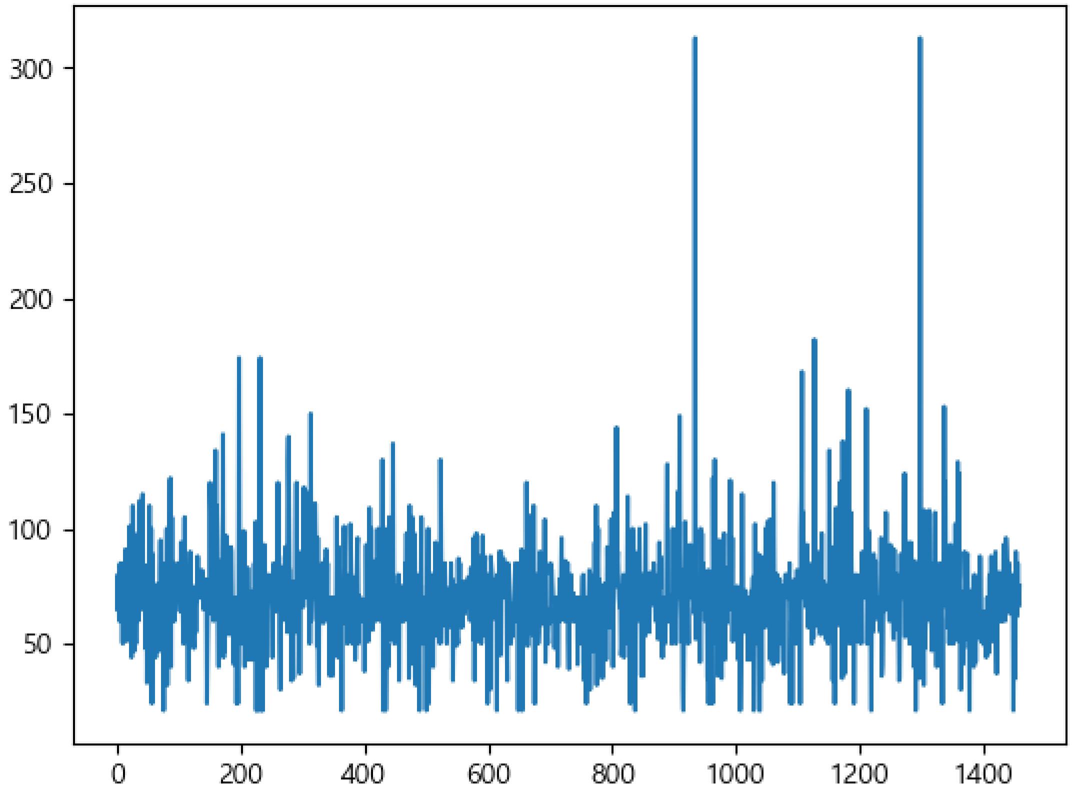
In [309]:
```
house1.LotFrontage.mean()
test.LotFrontage.mean()
```
executed in 15ms, finished 16:47:26 2023-11-01

Out[309]:
```
68.58035714285714
```

In [310]:
```
house1.LotFrontage = house1.LotFrontage.fillna(70)
test.LotFrontage = test.LotFrontage.fillna(69)
```
executed in 14ms, finished 16:47:26 2023-11-01

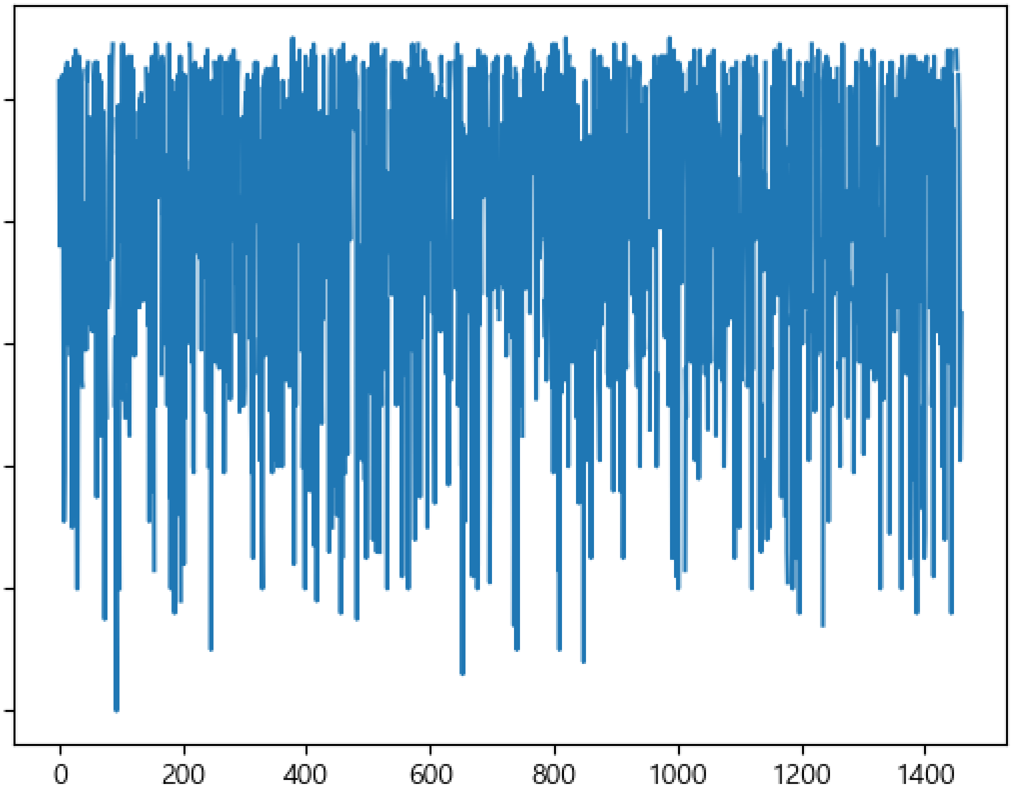
In [311]:
```
plt.plot(house1.LotFrontage)
```
executed in 140ms, finished 16:47:26 2023-11-01

Out[311]:
```
[<matplotlib.lines.Line2D at 0x252a461b6a0>]
```



In [312]:
```
plt.plot(house1.GarageYrBlt)
```
executed in 153ms, finished 16:47:26 2023-11-01

Out[312]:
```
[<matplotlib.lines.Line2D at 0x252a48eb250>]
```



In [313]:
```
test = test[['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond
         'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
         'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
         'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
         'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
         'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
         'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
         'MoSold', 'YrSold']]
```
executed in 15ms, finished 16:47:26 2023-11-01

In [314]:
```
house1.GarageYrBlt = house1.GarageYrBlt.fillna(house1.GarageYrBlt.mean())
test.GarageYrBlt = test.GarageYrBlt.fillna(test.GarageYrBlt.mean())
```
executed in 13ms, finished 16:47:26 2023-11-01

In [315]:
```
house1.MasVnrArea = house1.MasVnrArea.fillna(0)
test.MasVnrArea = test.MasVnrArea.fillna(0)
```
executed in 14ms, finished 16:47:26 2023-11-01

In [316]:
```
test = test.fillna(0)
```
executed in 13ms, finished 16:47:26 2023-11-01

In [ ]:
```

```

In [ ]:

In [317]:
```
target
```
executed in 15ms, finished 16:47:26 2023-11-01

Out[317]:
```
0       13.247694
1       13.109011
2       13.317167
3       12.849398
4       13.429216
          ...
1455    13.072541
1456    13.254863
1457    13.493130
1458    12.864462
1459    12.901583
Name: SalePrice, Length: 1460, dtype: float64
```

In [349]:
```
house1.LowQualFinSF = -house1.LowQualFinSF
```
executed in 13ms, finished 16:49:27 2023-11-01

In [350]:
```
data = house1.iloc[:,:-1]
target = house1.iloc[:,-1]
```
executed in 12ms, finished 16:49:29 2023-11-01

In [351]:
```
data.iloc[:,14] = -data.iloc[:,14]
```
executed in 5ms, finished 16:49:29 2023-11-01

In [352]:
```
mm.fit(data)
```
executed in 24ms, finished 16:49:29 2023-11-01

Out[352]:
```
▾ MinMaxScaler
MinMaxScaler()
```

In [ ]:

In [353]:
```
data_mm = mm.transform(data)
```
executed in 16ms, finished 16:49:30 2023-11-01

In [354]:
```
data_mm
```
executed in 17ms, finished 16:49:30 2023-11-01

Out[354]:
```
array([[0.23529412, 0.15068493, 0.0334198 , ..., 0.        , 0.09090909,
        0.5       ],
       [0.        , 0.20205479, 0.03879502, ..., 0.        , 0.36363636,
        0.25      ],
       [0.23529412, 0.1609589 , 0.04650728, ..., 0.        , 0.72727273,
        0.5       ],
       ...,
       [0.29411765, 0.15410959, 0.03618687, ..., 0.16129032, 0.36363636,
        1.        ],
       [0.        , 0.1609589 , 0.03934189, ..., 0.        , 0.27272727,
        1.        ],
       [0.        , 0.18493151, 0.04037019, ..., 0.        , 0.45454545,
        0.5       ]])
```

In [355]:
```
house1.columns
```
executed in 12ms, finished 16:49:34 2023-11-01

Out[355]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

In [356]:
```
data_mm = pd.DataFrame(data = data_mm , columns = house1.columns[:-1])
```
executed in 16ms, finished 16:49:35 2023-11-01

In [357]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
```
executed in 18ms, finished 16:49:35 2023-11-01

In [358]:
```
lr.fit(data_mm , target)
```
executed in 1.84s, finished 16:49:37 2023-11-01

Out[358]:
```
▾ LogisticRegression
LogisticRegression()
```

In [359]:
```
pred.SalePrice = lr.predict(test_mm)
```
executed in 14ms, finished 16:49:37 2023-11-01

In [360]:
```
target.shape
```
executed in 13ms, finished 16:49:37 2023-11-01

Out[360]:
```
(1460,)
```

In [361]:
```
data.shape
```
executed in 13ms, finished 16:49:37 2023-11-01

Out[361]:
```
(1460, 36)
```

In [362]:
```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
```
executed in 8ms, finished 16:49:43 2023-11-01

In [363]:
```
target.shape
```
executed in 5ms, finished 16:49:43 2023-11-01

Out[363]:
```
(1460,)
```

In [364]:
```
target.shape
```
executed in 19ms, finished 16:49:43 2023-11-01

Out[364]:
```
(1460,)
```

In [365]:
```
data = house1.iloc[:,:-1]
target = house1.iloc[:,-1]
```
executed in 7ms, finished 16:49:43 2023-11-01

In [366]:
```
rf.fit(data,target)
```
executed in 6.04s, finished 16:49:50 2023-11-01

Out[366]:
```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [367]:
```
data_mm
```
executed in 29ms, finished 16:49:50 2023-11-01

Out[367]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemod
0	0.235294	0.150685	0.033420	0.666667	0.500	0.949275	0.88
1	0.000000	0.202055	0.038795	0.555556	0.875	0.753623	0.43
2	0.235294	0.160959	0.046507	0.666667	0.500	0.934783	0.86
3	0.294118	0.133562	0.038561	0.666667	0.500	0.311594	0.33
4	0.235294	0.215753	0.060576	0.777778	0.500	0.927536	0.83
...	...	...	...	...	...	...	
1455	0.235294	0.140411	0.030929	0.555556	0.500	0.920290	0.83
1456	0.000000	0.219178	0.055505	0.555556	0.625	0.768116	0.63
1457	0.294118	0.154110	0.036187	0.666667	1.000	0.500000	0.93
1458	0.000000	0.160959	0.039342	0.444444	0.625	0.565217	0.76
1459	0.000000	0.184932	0.040370	0.444444	0.625	0.673913	0.25

1460 rows × 36 columns

In [368]:
```
house1.shape , test.shape
```
executed in 14ms, finished 16:49:50 2023-11-01

Out[368]:
```
((1460, 37), (1459, 36))
```

In [369]:
```
data.shape
```
executed in 13ms, finished 16:49:50 2023-11-01

Out[369]:
```
(1460, 36)
```

In [370]:
```
test.shape
```
executed in 14ms, finished 16:49:50 2023-11-01

Out[370]:
```
(1459, 36)
```

In [371]:
```
test_scaled = mm.transform(test)
```
executed in 9ms, finished 16:49:52 2023-11-01

In [372]:
```
test_mm = pd.DataFrame(data = test_scaled , columns = house1.columns[:-1])
```
executed in 15ms, finished 16:49:53 2023-11-01

In [373]:
```
pred.to_csv('house.csv' , index = False)
```
executed in 9ms, finished 16:49:53 2023-11-01

In [ ]:

In [342]:
```
# 문자열을 수치형 데이터로 바꿔주는 함수
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```
executed in 14ms, finished 16:47:35 2023-11-01

In [ ]: