모델 개선 및 교차 검증

학습 목표

- 새로운 피처를 만들고, 이를 통해 평가 검증해 본다.

- 데이콘 대회 : https://dacon.io/competitions/official/235745/overview/description (https://dacon.io/competitions/official/235745/overview/description)
- 오류 관련 링크 : https://dacon.io/competitions/official/235745/talkboard/403708?page=1&dtype=recent (https://dacon.io/competitions/official/235745/talkboard/403708?page=1&dtype=recent)

In [59]:

```
### 한글 폰트 설정
import matplotlib
from matplotlib import font_manager, rc
import matplotlib.pyplot as plt
import platform
import numpy as np

path = "C:/Windows/Fonts/malgun.ttf"
if platform.system() == "Windows":
    font_name = font_manager.FontProperties(fname=path).get_name()
    rc('font', family=font_name)
elif platform.system()=="Darwin":
    rc('font', family='AppleGothic')
else:
    print("Unknown System")

matplotlib.rcParams['axes.unicode_minus'] = False

%matplotlib inline
```

In [60]:

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

In [61]:

```
import pandas as pd

train = pd.read_csv("../../data/parking_demand/train_df_errno.csv")
test = pd.read_csv("../../data/parking_demand/test_df.csv")
sub = pd.read_csv("../../data/parking_demand/sample_submission.csv")
age = pd.read_csv("../../data/parking_demand/age_gender_info.csv")

train.shape, test.shape, sub.shape, age.shape
```

Out[61]:

```
((2896, 15), (1008, 14), (150, 2), (16, 23))
```

In [62]:

```
train.columns
```

Out[62]:

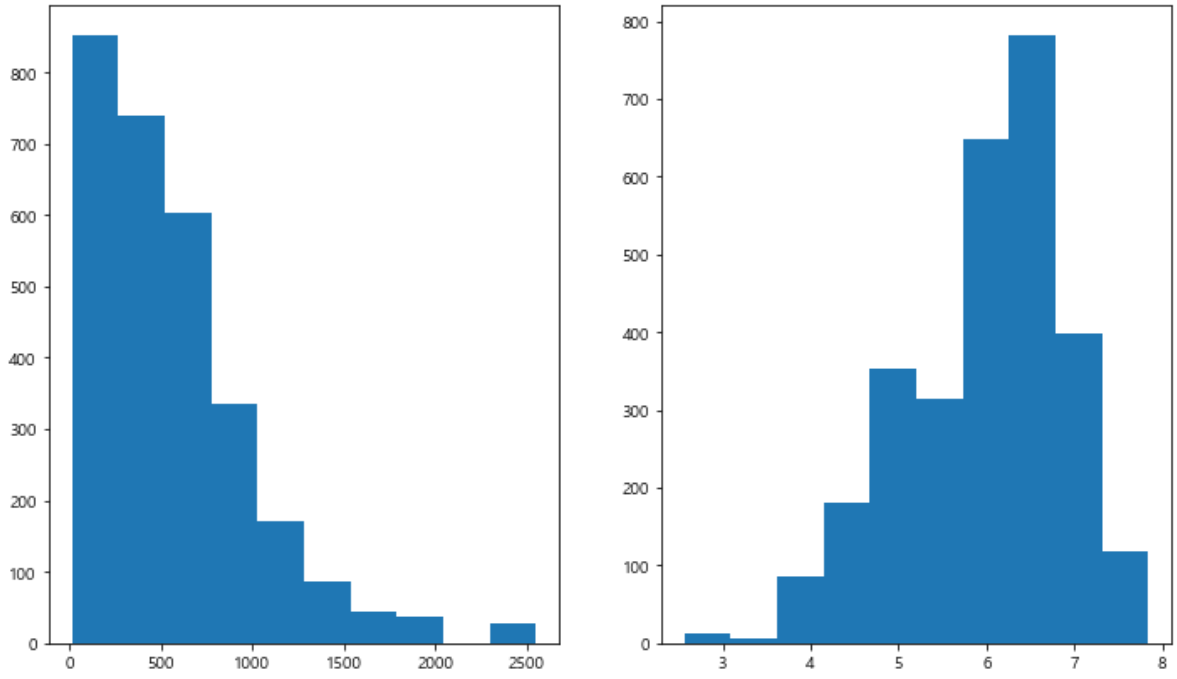
```
Index(['단지코드', '총세대수', '임대건물구분', '지역', '공급유형', '전용면적', '전용
면적별세대수', '공가수',
       '자격유형', '임대보증금', '임대료', '10분내지하철수', '10분내버스정류장수',
'단지내주차면수', '등록차량수'],
      dtype='object')
```

In [63]:

```
fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(12,7))

ax1.hist(train['등록차량수'])
ax2.hist(np.log(train['등록차량수']))
```

Out[63]:

```
(array([ 12.,   6.,  85., 180., 352., 314., 649., 781., 399., 118.]),
 array([2.56494936, 3.09283929, 3.62072921, 4.14861914, 4.67650907,
        5.204399  , 5.73228893, 6.26017885, 6.78806878, 7.31595871,
        7.84384864]),
 <BarContainer object of 10 artists>)
```



In [64]:

```
all_df = pd.concat([train, test], join='inner')
all_df
```

Out[64]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
0	C2515	545	아파트	경상남도	국민임대	33.48	276	17.0	A	9216000	82940	0.0	3.0	624.0
1	C2515	545	아파트	경상남도	국민임대	39.60	60	17.0	A	12672000	107130	0.0	3.0	624.0
2	C2515	545	아파트	경상남도	국민임대	39.60	20	17.0	A	12672000	107130	0.0	3.0	624.0
3	C2515	545	아파트	경상남도	국민임대	46.90	38	17.0	A	18433000	149760	0.0	3.0	624.0
4	C2515	545	아파트	경상남도	국민임대	46.90	19	17.0	A	18433000	149760	0.0	3.0	624.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1003	C1267	675	아파트	경상남도	행복주택	36.77	126	38.0	L	-	-	0.0	1.0	467.0
1004	C2189	382	아파트	전라북도	국민임대	29.19	96	45.0	H	6872000	106400	0.0	2.0	300.0
1005	C2189	382	아파트	전라북도	국민임대	29.19	20	45.0	H	6872000	106400	0.0	2.0	300.0
1006	C2189	382	아파트	전라북도	국민임대	39.45	202	45.0	H	13410000	144600	0.0	2.0	300.0

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
1007	C2189	382	아파트	전라북도	국민임대	46.23	60	45.0	H	18689000	166500	0.0	2.0	300.0

3904 rows × 14 columns

In [65]:

```
all_df.isnull().sum()
```

Out[65]:

```
단지코드            0
총세대수            0
임대건물구분          0
지역              0
공급유형            0
전용면적            0
전용면적별세대수        0
공가수             0
자격유형            2
임대보증금         749
임대료           749
10분내지하철수      249
10분내버스정류장수     4
단지내주차면수        0
dtype: int64
```

In [66]:

```
all_df.loc[all_df['자격유형'].isnull()]
```

Out[66]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
196	C2411	962	아파트	경상남도	국민임대	46.90	240	25.0	NaN	71950000	37470	0.0	2.0	840.0
258	C2253	1161	아파트	강원도	영구임대	26.37	745	0.0	NaN	2249000	44770	0.0	2.0	173.0

In [67]:

```
all_df.loc[ 196, "자격유형"] = 'A'
all_df.loc[ 258, "자격유형"] = 'C'
```

In [68]:

```
mapping = { 'A':1, 'B':2, 'C':3, 'D':4, 'E':5,
            'F':6, 'G':7, 'H':8, 'I':9, 'J':10,
            'K':11, 'L':12, 'M':13, 'N':14, 'O':15  }

all_df['자격유형'] =all_df['자격유형'].map(mapping).astype(int)
```

In [69]:

```
all_df.head()
```

Out[69]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
0	C2515	545	아파트	경상남도	국민임대	33.48	276	17.0	1	9216000	82940	0.0	3.0	624.0
1	C2515	545	아파트	경상남도	국민임대	39.60	60	17.0	1	12672000	107130	0.0	3.0	624.0
2	C2515	545	아파트	경상남도	국민임대	39.60	20	17.0	1	12672000	107130	0.0	3.0	624.0
3	C2515	545	아파트	경상남도	국민임대	46.90	38	17.0	1	18433000	149760	0.0	3.0	624.0
4	C2515	545	아파트	경상남도	국민임대	46.90	19	17.0	1	18433000	149760	0.0	3.0	624.0

In [70]:

```
all_df.isnull().sum()
```

Out[70]:

```
단지코드              0
총세대수              0
임대건물구분            0
지역                0
공급유형              0
전용면적              0
전용면적별세대수          0
공가수               0
자격유형              0
임대보증금           749
임대료             749
10분내지하철수        249
10분내버스정류장수       4
단지내주차면수          0
dtype: int64
```

In [71]:

```
all_df.corr()['10분내버스정류장수']
```

Out[71]:

```
총세대수          -0.002576
전용면적           0.002303
전용면적별세대수       0.040635
공가수            0.038906
자격유형          -0.014581
10분내지하철수       0.058901
10분내버스정류장수     1.000000
단지내주차면수        0.097617
Name: 10분내버스정류장수, dtype: float64
```

In [72]:

```
grouped = train.groupby(['임대건물구분', '지역'])
group1 = grouped.get_group( ('아파트', '경상남도')  )
group1['10분내버스정류장수'].mean()
```

Out[72]:

```
3.996268656716418
```

In [73]:

```
grouped = train.groupby(['임대건물구분', '지역'])
group1 = grouped.get_group( ('아파트', '경상남도')  )
val = group1['10분내버스정류장수'].mean()
val
```

Out[73]:

```
3.996268656716418
```

In [74]:

```
# 데이터 확인 후,
all_df.loc[ all_df['10분내버스정류장수'].isnull(), "10분내버스정류장수"] = val
```

In [75]:

```
all_df.loc[ all_df['10분내버스정류장수'].isnull(), :]
```

Out[75]:

단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수

In [76]:

```
all_df.isnull().sum()
```

Out[76]:

```
단지코드            0
총세대수            0
임대건물구분          0
지역              0
공급유형            0
전용면적            0
전용면적별세대수        0
공가수             0
자격유형            0
임대보증금         749
임대료           749
10분내지하철수      249
10분내버스정류장수      0
단지내주차면수         0
dtype: int64
```

In [77]:

```
all_df.shape
```

Out[77]:

```
(3904, 14)
```

In [78]:

```
all_df.head()
```

Out[78]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
0	C2515	545	아파트	경상남도	국민임대	33.48	276	17.0	1	9216000	82940	0.0	3.0	624.0
1	C2515	545	아파트	경상남도	국민임대	39.60	60	17.0	1	12672000	107130	0.0	3.0	624.0
2	C2515	545	아파트	경상남도	국민임대	39.60	20	17.0	1	12672000	107130	0.0	3.0	624.0
3	C2515	545	아파트	경상남도	국민임대	46.90	38	17.0	1	18433000	149760	0.0	3.0	624.0
4	C2515	545	아파트	경상남도	국민임대	46.90	19	17.0	1	18433000	149760	0.0	3.0	624.0

In [79]:

```
gubun1 = {'아파트':1, '상가':2}
gubun2 = {'경상남도':1, '대전광역시':2, '경기도':3, '전라북도':4,
          '강원도':5, '광주광역시':6, '충청남도':7, '부산광역시':8,
          '제주특별자치도':9, '울산광역시':10, '충청북도':11, '전라남도':12,
          '경상북도':13, '대구광역시':14, '서울특별시':15, '세종특별자치시':16}

gubun3 = {'국민임대':1, '공공임대(50년)':2, '영구임대':3, '임대상가':4,
          '공공임대(10년)':5, '공공임대(분납)':6, '장기전세':7, '공공분양':8,
          '행복주택':9, '공공임대(5년)':10}

all_df['임대건물구분_lbl'] = all_df['임대건물구분'].map(gubun1)
all_df['지역_lbl'] = all_df['지역'].map(gubun2)
all_df['공급유형_lbl'] = all_df['공급유형'].map(gubun3)

all_df
```

Out[79]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	_l
0	C2515	545	아파트	경상남도	국민임대	33.48	276	17.0	1	9216000	82940	0.0	3.0	624.0	1	
1	C2515	545	아파트	경상남도	국민임대	39.60	60	17.0	1	12672000	107130	0.0	3.0	624.0	1	
2	C2515	545	아파트	경상남도	국민임대	39.60	20	17.0	1	12672000	107130	0.0	3.0	624.0	1	
3	C2515	545	아파트	경상남도	국민임대	46.90	38	17.0	1	18433000	149760	0.0	3.0	624.0	1	
4	C2515	545	아파트	경상남도	국민임대	46.90	19	17.0	1	18433000	149760	0.0	3.0	624.0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1003	C1267	675	아파트	경상남도	행복주택	36.77	126	38.0	12	-	-	0.0	1.0	467.0	1	

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	_l
1004	C2189	382	아파트	전라북도	국민임대	29.19	96	45.0	8	6872000	106400	0.0	2.0	300.0	1	
1005	C2189	382	아파트	전라북도	국민임대	29.19	20	45.0	8	6872000	106400	0.0	2.0	300.0	1	
1006	C2189	382	아파트	전라북도	국민임대	39.45	202	45.0	8	13410000	144600	0.0	2.0	300.0	1	
1007	C2189	382	아파트	전라북도	국민임대	46.23	60	45.0	8	18689000	166500	0.0	2.0	300.0	1	

3904 rows × 17 columns

feature 연산

In [80]:

```
all_df['단지코드'] = all_df['단지코드'].astype("category")
all_df['단지코드_lbl'] = all_df['단지코드'].cat.codes

### 전용면적을 구간화하기
all_df['전용면적별세대수'] = all_df['전용면적별세대수'].astype('float32')

# 전용면적, 공가수, 단지내주차면수
all_df['qcut_총세대수'] = pd.qcut(all_df['총세대수'], 5, labels=False)
all_df.head(10)
```

Out[80]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	_l
0	C2515	545	아파트	경상남도	국민임대	33.48	276.0	17.0	1	9216000	82940	0.0	3.0	624.0	1	
1	C2515	545	아파트	경상남도	국민임대	39.60	60.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	
2	C2515	545	아파트	경상남도	국민임대	39.60	20.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	
3	C2515	545	아파트	경상남도	국민임대	46.90	38.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	
4	C2515	545	아파트	경상남도	국민임대	46.90	19.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	
5	C2515	545	아파트	경상남도	국민임대	51.97	106.0	17.0	1	23042000	190090	0.0	3.0	624.0	1	
6	C2515	545	아파트	경상남도	국민임대	51.97	26.0	17.0	1	23042000	190090	0.0	3.0	624.0	1	
7	C1407	1216	아파트	대전광역시	국민임대	30.95	288.0	13.0	1	15620000	127350	1.0	1.0	1285.0	1	

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	_l
8	C1407	1216	아파트	대전광역시	국민임대	30.99	68.0	13.0	1	15620000	127350	1.0	1.0	1285.0	1	
9	C1407	1216	아파트	대전광역시	국민임대	30.99	34.0	13.0	1	15620000	127350	1.0	1.0	1285.0	1	

전용면적별 세대의 합계와 총세대수가 일치하지 않는 오류

- 차이가 14세대 이하인 48개 단지 - ['C1925', 'C1312', 'C2013', 'C1424', 'C2520', 'C2319', 'C1850', 'C1068', 'C2644', 'C2156', C2453', 'C1910', 'C2139', 'C2508', 'C1695', 'C2556', 'C2362', 'C2568', 'C2245', 'C2549', 'C1584', 'C2298', 'C2225', 'C1218', 'C1970', C1732', 'C2433', 'C1894', 'C1156', 'C2142', 'C2186', 'C2411', 'C1812', 'C1030', 'C1749', 'C1349', 'C2043', 'C1229', 'C2363', 'C1414', C2174', 'C2404', 'C1683', 'C1038', 'C2456', 'C1266', 'C1267', 'C2189']
- 차이가 94~452세대인 10개 단지(크기순) - ['C1490', 'C2497', 'C2620', 'C1344', 'C1024', 'C2470', 'C1206', 'C1740', 'C2405', 'C1804']

In [81]:

```
group1 = ['C1925', 'C1312', 'C2013', 'C1424', 'C2520', 'C2319', 'C1850', 'C1068', 'C2644', 'C2156',
          'C2453', 'C1910', 'C2139', 'C2508', 'C1695', 'C2556', 'C2362', 'C2568', 'C2245', 'C2549',
          'C1584', 'C2298', 'C2225', 'C1218', 'C1970', 'C1732', 'C2433', 'C1894', 'C1156', 'C2142',
          'C2186', 'C2411', 'C1812', 'C1030', 'C1749', 'C1349', 'C2043', 'C1229', 'C2363', 'C1414',
          'C2174', 'C2404', 'C1683', 'C1038', 'C2456', 'C1266', 'C1267', 'C2189' ]

for one1 in group1:
    all_df.loc[ all_df['단지코드'] == one1, "단지코드_Type" ] = 1
```

In [82]:

```
all_df['단지코드_Type'].unique()
```

Out[82]:

```
array([nan,  1.])
```

실습 1. 차이가 94~452세대인 10개단지 처리하기

In [83]:

```
all_df.loc[  all_df['단지코드_Type'].isna(), "단지코드_Type"] = 3
all_df['단지코드_Type'].unique()
```

Out[83]:

```
array([3., 1.])
```

In [84]:

```
all_df_last = all_df.drop(['임대건물구분', '지역', '공급유형'] , axis=1)
all_df_last
```

Out[84]:

	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지역_lbl	공급유형_lbl	_
0	C2515	545	33.48	276.0	17.0	1	9216000	82940	0.0	3.0	624.0	1	1	1	4
1	C2515	545	39.60	60.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	4
2	C2515	545	39.60	20.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	4
3	C2515	545	46.90	38.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	4
4	C2515	545	46.90	19.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1003	C1267	675	36.77	126.0	38.0	12	-	-	0.0	1.0	467.0	1	1	9	
1004	C2189	382	29.19	96.0	45.0	8	6872000	106400	0.0	2.0	300.0	1	4	1	3
1005	C2189	382	29.19	20.0	45.0	8	6872000	106400	0.0	2.0	300.0	1	4	1	3
1006	C2189	382	39.45	202.0	45.0	8	13410000	144600	0.0	2.0	300.0	1	4	1	3
1007	C2189	382	46.23	60.0	45.0	8	18689000	166500	0.0	2.0	300.0	1	4	1	3

3904 rows × 17 columns

In [85]:

```
all_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3904 entries, 0 to 1007
Data columns (total 20 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   단지코드        3904 non-null   category
 1   총세대수        3904 non-null   int64
 2   임대건물구분      3904 non-null   object
 3   지역          3904 non-null   object
 4   공급유형        3904 non-null   object
 5   전용면적        3904 non-null   float64
 6   전용면적별세대수    3904 non-null   float32
 7   공가수         3904 non-null   float64
 8   자격유형        3904 non-null   int32
 9   임대보증금       3155 non-null   object
 10  임대료         3155 non-null   object
 11  10분내지하철수    3655 non-null   float64
 12  10분내버스정류장수  3904 non-null   float64
 13  단지내주차면수     3904 non-null   float64
 14  임대건물구분_lbl  3904 non-null   int64
 15  지역_lbl      3904 non-null   int64
 16  공급유형_lbl    3904 non-null   int64
 17  단지코드_lbl    3904 non-null   int16
 18  qcut_총세대수   3904 non-null   int64
 19  단지코드_Type   3904 non-null   float64
dtypes: category(1), float32(1), float64(6), int16(1), int32(1), int64(5), object(5)
memory usage: 668.6+ KB
```

In [86]:

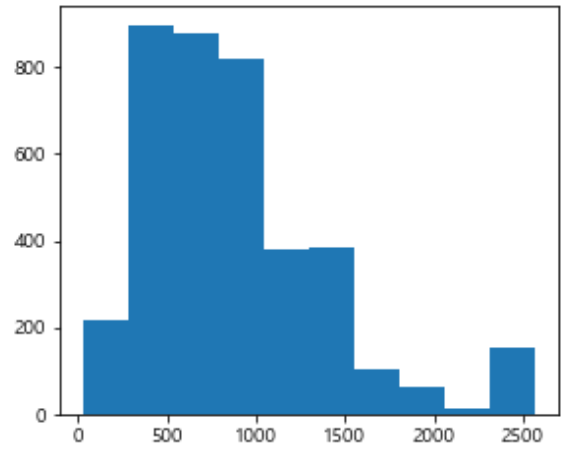
```
# '총세대수' : 0.333440, '단지내주차면수' : 0.861338, 임대건물구분_lbl : -0.449130
# 전용면적    0.112717, 전용면적별세대수   0.250513, 공가수          0.118910,
# 자격유형        -0.154034, 공급유형_lbl     -0.137277
```

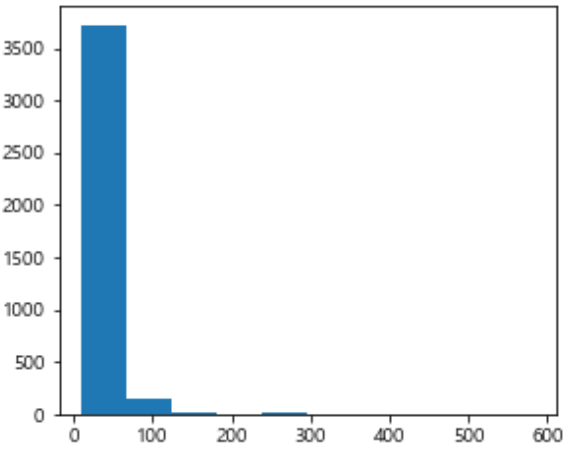
In [87]:

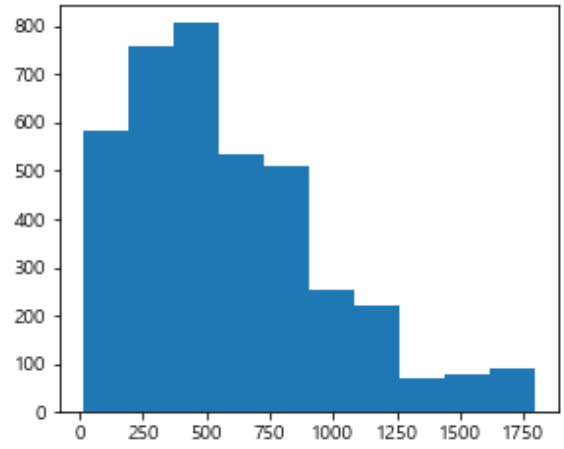
```
fig, ax = plt.subplots(nrows=3, ncols=2, figsize=(10,13))

ax[0][0].hist(all_df_last['총세대수'])
ax[0][1].hist(all_df_last['전용면적'])
ax[1][0].hist(all_df_last['단지내주차면수'])
ax[1][1].hist(all_df_last['10분내지하철수'])
ax[2][0].hist(all_df_last['10분내버스정류장수'])
ax[2][1].hist(all_df_last['전용면적별세대수'])
```

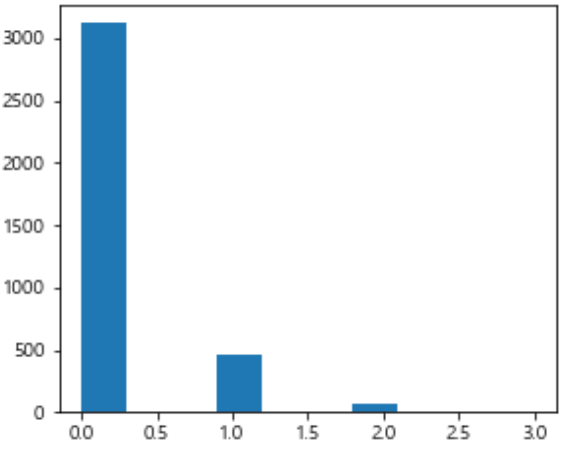
Out[87]:

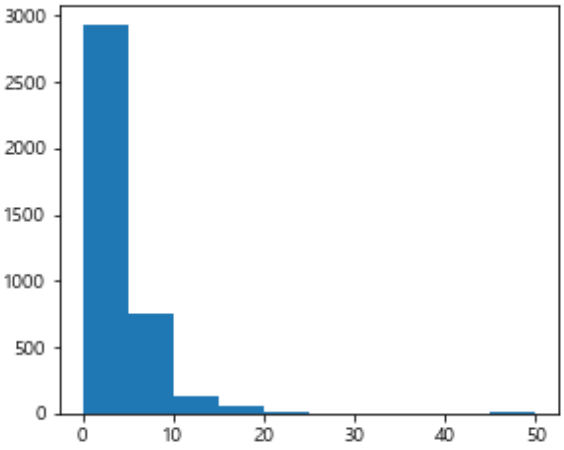
```
(array([3.266e+03, 4.760e+02, 1.060e+02, 4.500e+01, 4.000e+00, 2.000e+00,
        1.000e+00, 3.000e+00, 0.000e+00, 1.000e+00]),
 array([1.0000e+00, 1.8740e+02, 3.7380e+02, 5.6020e+02, 7.4660e+02,
        9.3300e+02, 1.1194e+03, 1.3058e+03, 1.4922e+03, 1.6786e+03,
        1.8650e+03], dtype=float32),
 <BarContainer object of 10 artists>)
```

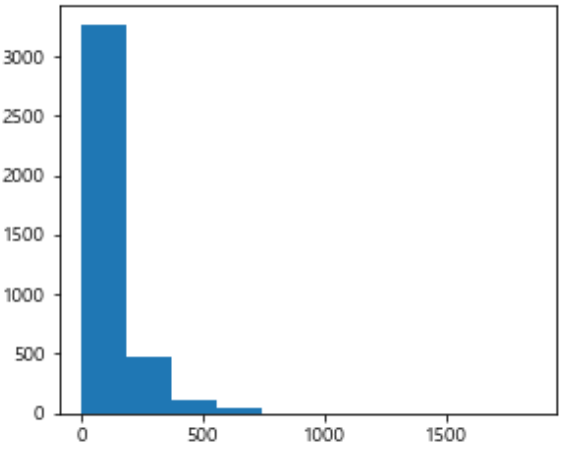

0
200
400
600
800
0
500
1000
1500
2000
2500


0
500
1000
1500
2000
2500
3000
3500
0
100
200
300
400
500
600


0
100
200
300
400
500
600
700
800
0
250
500
750
1000
1250
1500
1750


0
500
1000
1500
2000
2500
3000
0.0
0.5
1.0
1.5
2.0
2.5
3.0


0
500
1000
1500
2000
2500
3000
0
10
20
30
40
50


0
500
1000
1500
2000
2500
3000
0
500
1000
1500

In [88]:

```
train_df = all_df_last.iloc[0:2896,:]
test_df = all_df_last.iloc[2896:,:]

train_df.shape, test_df.shape

train_df = pd.concat([train_df, train['등록차량수'] ], axis=1)
train_df
```

Out[88]:

	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지역_lbl	공급유형_lbl	_
0	C2515	545	33.48	276.0	17.0	1	9216000	82940	0.0	3.0	624.0	1	1	1	4
1	C2515	545	39.60	60.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	4
2	C2515	545	39.60	20.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	4
3	C2515	545	46.90	38.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	4
4	C2515	545	46.90	19.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2891	C2532	239	49.20	19.0	7.0	1	11346000	116090	0.0	1.0	166.0	1	5	1	5
2892	C2532	239	51.08	34.0	7.0	1	14005000	142310	0.0	1.0	166.0	1	5	1	5
2893	C2532	239	51.73	34.0	7.0	1	14005000	142310	0.0	1.0	166.0	1	5	1	5
2894	C2532	239	51.96	114.0	7.0	1	14005000	142310	0.0	1.0	166.0	1	5	1	5
2895	C2532	239	54.95	19.0	7.0	1	14830000	151030	0.0	1.0	166.0	1	5	1	5

2896 rows × 18 columns

In [89]:

```
train_df['log_등록차량수'] = np.log1p(train_df['등록차량수'])
```

In [90]:

```
from sklearn.model_selection import train_test_split

print("등록차량수 상관계수 : ", train_df.corr()['등록차량수'])
print()
print("log_등록차량수 상관계수 : ", train_df.corr()['log_등록차량수'])
```

```
등록차량수 상관계수 :  총세대수          0.333440
전용면적           0.112717
전용면적별세대수       0.250513
공가수            0.118910
자격유형           -0.154034
10분내지하철수      -0.107308
10분내버스정류장수     0.104200
단지내주차면수        0.861338
임대건물구분_lbl    -0.449130
지역_lbl         0.060674
공급유형_lbl      -0.137277
단지코드_lbl      -0.062077
qcut_총세대수      0.401309
단지코드_Type      0.133207
등록차량수          1.000000
log_등록차량수      0.881679
Name: 등록차량수, dtype: float64

log_등록차량수 상관계수 :  총세대수          0.228964
전용면적           0.111644
전용면적별세대수       0.274772
공가수            0.200793
자격유형           -0.238579
10분내지하철수      -0.125054
10분내버스정류장수     0.080122
단지내주차면수        0.805096
임대건물구분_lbl    -0.596565
지역_lbl         0.108532
공급유형_lbl      -0.291515
단지코드_lbl      -0.065896
qcut_총세대수      0.303550
단지코드_Type      0.109606
등록차량수          0.881679
log_등록차량수      1.000000
Name: log_등록차량수, dtype: float64
```

In [91]:

```
pd.set_option('display.max_columns',500)
```

In [92]:

```
train_df.head()
```

Out[92]:

	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지역_lbl	공급유형_lbl	단지코드_lbl
0	C2515	545	33.48	276.0	17.0	1	9216000	82940	0.0	3.0	624.0	1	1	1	492
1	C2515	545	39.60	60.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	492
2	C2515	545	39.60	20.0	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	492
3	C2515	545	46.90	38.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	492
4	C2515	545	46.90	19.0	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	492

In [102]:

```
sel = [ '총세대수', '전용면적', '공가수', '단지내주차면수',
       'qcut_총세대수', '자격유형', '전용면적별세대수', '10분내버스정류장수',
       '임대건물구분_lbl', '공급유형_lbl', '지역_lbl', '단지코드_lbl',
       '단지코드_Type']

lable_name = 'log_등록차량수'
X = train_df[sel]
y = train_df[lable_name]
test_X = test_df[sel]

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.1,
                                                    random_state=0)
```

In [103]:

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
#import xgboost as xgb
#import lightgbm as lgb
```

In [104]:

```
model = RandomForestRegressor(n_jobs=-1)
model.fit(X_train, y_train)
pred = model.predict(X_test)

print("학습(score) :", model.score(X_train, y_train) ) # 결정계수
print("테스트(score) :", model.score(X_test, y_test) ) # 결정계수
```

```
학습(score) : 0.9990904288073044
테스트(score) : 0.9983978478845238
```

In [105]:

```
model = GradientBoostingRegressor()
model.fit(X_train, y_train)
pred = model.predict(X_test)

print("학습(score) :", model.score(X_train, y_train) ) # 결정계수
print("테스트(score) :", model.score(X_test, y_test) ) # 결정계수
```

```
학습(score) : 0.9596969868857229
테스트(score) : 0.9589475185389966
```

In [106]:

```
import time
```

In [117]:

```
now_time = time.time()

model_RF = RandomForestRegressor(n_estimators = 1000,
                                 random_state=0, n_jobs=-1)
model_RF.fit(X_train, y_train)
score = cross_val_score(model_RF, X_train, y_train,
                        cv=5, scoring="neg_mean_absolute_error") # neg_mean_squared_error
m_score = np.abs(score.mean())
print("RandomForestRegressor Score : {}".format(m_score))  # 점수

pro_time = time.time() - now_time
print(pro_time)  # 걸린 시간
```

```
RandomForestRegressor Score : 0.03266266039479642
15.470895290374756
```

In [118]:

```
hyperparameters = {'boosting_type': 'gbdt',
                   'colsample_bytree': 0.7250136792694301,
                   'is_unbalance': False,
                   'learning_rate': 0.013,
                   'min_child_samples': 20,
                   'num_leaves': 56,
                   'subsample': 0.5233384321711397,
                   'n_estimators': 1000}
```

In [109]:

```
import lightgbm as lgb

ow_time = time.time()

m_lgbm1 = lgb.LGBMRegressor(**hyperparameters)
m_lgbm1.fit(X_train, y_train)
score = cross_val_score(m_lgbm1, X_train, y_train,
                        cv=5, scoring="neg_mean_absolute_error")

m_score = np.abs(score.mean())  # 절대값
pro_time = time.time() - now_time

print(pro_time)  # 걸린 시간
print("LightGBM Score : {}".format(m_score))  # 점수
```

```
51.80305099487305
LightGBM Score : 0.044679934497532305
```

```
    hyperparameters = {'boosting_type': 'gbdt',
                       'colsample_bytree': 0.7250136792694301,
                       'is_unbalance': False,
                       'learning_rate': 0.05,
                       'min_child_samples': 20,
                       'num_leaves': 56,
                       'subsample': 0.5233384321711397,
                       'n_estimators': 1000}
```

In [110]:

```
# sel = [ '총세대수', '전용면적', '전용면적별세대수', '공가수',
#        '자격유형', '단지내주차면수',
#        '임대건물구분_lbl', '공급유형_lbl', '지역_lbl']
```

In [111]:

```
hyperparameters = {'boosting_type': 'gbdt',
                   'colsample_bytree': 0.7250136792694301,
                   'is_unbalance': False,
                   'learning_rate': 0.013,
                   'min_child_samples': 20,
                   'num_leaves': 56,
                   'subsample': 0.5233384321711397,
                   'n_estimators': 1000}

model_last = lgb.LGBMRegressor(**hyperparameters)
model_last.fit(X_train, y_train)
pred = model_last.predict(test_X)
pred[0:10]
```

Out[111]:

```
array([6.56643283, 6.53774823, 6.55184919, 6.55200065, 6.54788285,
       6.54807384, 6.50607099, 6.53878309, 7.09102681, 7.07892433])
```

In [112]:

```
test_df['등록차량수'] = np.expm1(pred)
test_df['단지별차량수평균'] = test_df.groupby("단지코드")['등록차량수'].transform(np.mean)
test_new = test_df.drop_duplicates(['단지코드'], keep='first').reset_index()
test_new
```

```
<ipython-input-112-4240d87ed58b>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  test_df['등록차량수'] = np.expm1(pred)
<ipython-input-112-4240d87ed58b>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  test_df['단지별차량수평균'] = test_df.groupby("단지코드")['등록차량수'].transform
(np.mean)
```

Out[112]:

	index	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지_lb
0	0	C1072	754	39.79	116.0	14.0	8	22830000	189840	0.0	2.0	683.0	1	
1	8	C1128	1354	39.79	368.0	9.0	8	22830000	189840	0.0	3.0	1216.0	1	
2	17	C1456	619	33.40	82.0	18.0	1	19706000	156200	0.0	16.0	547.0	1	
3	26	C1840	593	39.57	253.0	7.0	1	14418000	108130	0.0	3.0	543.0	1	
4	30	C1332	1297	39.99	282.0	11.0	8	28598000	203050	0.0	2.0	1112.0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
142	982	C2456	349	26.44	24.0	17.0	8	6992000	117000	0.0	4.0	270.0	1	
143	986	C1266	596	26.94	164.0	35.0	8	8084000	149910	0.0	1.0	593.0	1	1
144	991	C2152	120	24.83	66.0	9.0	3	-	-	0.0	1.0	40.0	1	
145	993	C1267	675	24.87	28.0	38.0	8	6882000	104370	0.0	1.0	467.0	1	
146	1004	C2189	382	29.19	96.0	45.0	8	6872000	106400	0.0	2.0	300.0	1	

147 rows × 20 columns

In [113]:

```
add_dat = {'code':['C2675', 'C2335', 'C1327'],
           'num':['0', '0', '0']}
add_df = pd.DataFrame(add_dat)
```

In [114]:

```
sub_df = test_new[ ['단지코드', '단지별차량수평균']]
sub_df.columns = ['code', 'num']
sub_df = pd.concat([sub_df, add_df]).reset_index()
sub_df = sub_df.drop(['index'], axis=1)
sub_df
```

Out[114]:

	code	num
0	C1072	693.88
1	C1128	1205.46
2	C1456	622.815
3	C1840	573.222
4	C1332	1141.6
...	...	...
145	C1267	395.787
146	C2189	192.549
147	C2675	0
148	C2335	0
149	C1327	0

150 rows × 2 columns

In [115]:

```
sub_df.to_csv('sixth_lgbm_0720.csv', index=False)
sub_df.head()
```

Out[115]:

	code	num
0	C1072	693.88
1	C1128	1205.46
2	C1456	622.815
3	C1840	573.222
4	C1332	1141.6

In [116]:

```
import os
os.listdir(os.getcwd())
```

Out[116]:

```
['.ipynb_checkpoints',
 '06_model_val_01.ipynb',
 '07_feature_engineering.html',
 '07_feature_engineering_01.html',
 '07_feature_engineering_01.ipynb',
 '07_feature_engineering_01.pdf',
 '07_feature_engineering_02.ipynb',
 'sixth_lgbm_0720.csv']
```

In [ ]:

```

```