랜덤 포레스트 모델 구축하기

학습 목표

- 라벨 인코딩을 수행한다.
- 데이터 EDA 및 시각화를 통해 데이터를 이해하고 기본 모델을 만들어본다.
- 모델을 제출해 본다.(랜덤 포레스트 등)

- 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)

01. 데이터 불러오기 및 확인

In [1]:

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

In [2]:

```
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[2]:

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

In [3]:

```
train.columns
```

Out[3]:

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

In [4]:

```
train.columns = ['단지코드', '총세대수', '임대건물구분', '지역', '공급유형', '전용면적', '전용면적별
       '자격유형', '임대보증금', '임대료', '10분내지하철수',
       '10분내버스정류장수', '단지내주차면수', '등록차량수']

test.columns = ['단지코드', '총세대수', '임대건물구분', '지역', '공급유형', '전용면적', '전용면적별
       '자격유형', '임대보증금', '임대료', '10분내지하철수',
       '10분내버스정류장수', '단지내주차면수']
```

02. 결측치를 처리(1)

데이터 결합

In [5]: 결측치를 하나로 묶어서 하기 위해서. 병합.(공통된 열)

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

Out[5]:

	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 [6]:

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

Out[6]:

```
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
```

자격유형(test) 결측치 처리

In [7]:

```
all_df['지역'].unique()
```

Out[7]:

```
array(['경상남도', '대전광역시', '경기도', '전라북도', '강원도', '광주광역시', '충청
남도', '부산광역시',
       '제주특별자치도', '울산광역시', '충청북도', '전라남도', '경상북도', '대구광역
시', '서울특별시',
       '세종특별자치시'], dtype=object)
```

In [8]:

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

Out[8]:

	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 [9]:

```
grouped = all_df.groupby(['단지코드', '임대건물구분', '지역','공급유형'])
group1 = grouped.get_group( ('C2411', '아파트', '경상남도', '국민임대')  )
group1
```

Out[9]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
193	C2411	962	아파트	경상남도	국민임대	39.43	56	25.0	A	11992000	100720	0.0	2.0	840.0
194	C2411	962	아파트	경상남도	국민임대	39.72	336	25.0	A	11992000	100720	0.0	2.0	840.0
195	C2411	962	아파트	경상남도	국민임대	39.82	179	25.0	A	11992000	100720	0.0	2.0	840.0
196	C2411	962	아파트	경상남도	국민임대	46.90	240	25.0	NaN	71950000	37470	0.0	2.0	840.0
197	C2411	962	아파트	경상남도	국민임대	51.93	150	25.0	A	21586000	171480	0.0	2.0	840.0

In [10]:

```
group2 = grouped.get_group( ('C2253', '아파트', '강원도', '영구임대')  )
group2
```

Out[10]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
258	C2253	1161	아파트	강원도	영구임대	26.37	745	0.0	NaN	2249000	44770	0.0	2.0	173.0
259	C2253	1161	아파트	강원도	영구임대	31.32	239	0.0	C	3731000	83020	0.0	2.0	173.0
260	C2253	1161	아파트	강원도	영구임대	31.32	149	0.0	C	3731000	83020	0.0	2.0	173.0

In [11]:

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

In [12]:

```
print(all_df.자격유형.unique())
```

```
['A' 'B' 'C' 'D' 'E' 'F' 'G' 'H' 'I' 'J' 'K' 'L' 'M' 'N' 'O']
```

In [13]:

```
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 [14]:

```
print(all_df.공급유형.unique())
```

```
['국민임대' '공공임대(50년)' '영구임대' '임대상가' '공공임대(10년)' '공공임대(분납)'
'장기전세' '공공분양'
 '행복주택' '공공임대(5년)']
```

In [15]:

```
all_df.columns
```

Out[15]:

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

10분내버스정류장수 (tr) 결측치 처리,

In [16]:

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

Out[16]:

```
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 [17]:

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

Out[17]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수
2293	N2431	1047	아파트	경상남도	공공임대(10년)	74.97	80	15.0	1	46000000	456000	NaN	NaN	1066.0
2294	N2431	1047	아파트	경상남도	공공임대(10년)	84.95	124	15.0	1	57000000	462000	NaN	NaN	1066.0
2295	N2431	1047	아파트	경상남도	공공임대(10년)	84.96	289	15.0	1	57000000	462000	NaN	NaN	1066.0
2296	N2431	1047	아파트	경상남도	공공임대(10년)	84.98	82	15.0	1	57000000	462000	NaN	NaN	1066.0

In [18]:

```
all_df['임대건물구분'].unique()
```

Out[18]:

```
array(['아파트', '상가'], dtype=object)
```

In [19]: 100행이상 보이게 하겠다.

```
pd.set_option("display.max_rows", 100)
pd.get_option("display.max_rows")
```

Out[19]:

100

In [20]:

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

Out[20]:

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	등록차량수
0	C2515	545	아파트	경상남도	국민임대	33.48	276	17.0	A	9216000	82940	0.0	3.0	624.0	205.0
1	C2515	545	아파	경상남	국민임	39.60	60	17.0	A	12672000	107130	0.0	3.0	624.0	205.0

In [21]:

```
# 데이터 확인 후, 임의 처리 4
all_df.loc[ all_df['10분내버스정류장수'].isnull(), "10분내버스정류장수"] = 4
```

In [22]:

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

Out[22]:

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

In [23]:

```
all_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3904 entries, 0 to 1007
Data columns (total 14 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   단지코드        3904 non-null   object
 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   int64
 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
dtypes: float64(5), int32(1), int64(2), object(6)
memory usage: 522.2+ KB
```

In [24]:

```
all_df.head()
```

Out[24]:

	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

라벨 인코딩

In [29]:

```
for c in all_df.columns:
    print(all_df[c].unique())
```

```
['C2515', 'C1407', 'C1945', 'C1470', 'C1898', ..., 'C2456', 'C1266', 'C2152', 'C126
7', 'C2189']
Length: 561
Categories (561, object): ['C2515', 'C1407', 'C1945', 'C1470', ..., 'C1266', 'C215
2', 'C1267', 'C2189']
[ 545 1216  755  696  566 1722  624  361  754  240  688  623  601  405
  518  619  875  560  595  625  880 1396  970  606 1507  639  965  946
  785 2424  809  388 1013  453  861  806  903 1116 1486 1957  802 1527
 1005 1988 1350 2428 1755  840  390  451  410  779  693  753  498 1533
  711  420  590  657  495  460  338 1144  521 1003  306  697  213  481
  468 1364  800  830  775  261  474   72 1473  996  870  678  632  961
 1232  676 1300  998  493 1117  307  501  896  458  290  409  586 1084
 1174  270 1308  355  384  853  492  901  815  312  571  594  944  635
  962  822 1129 1479  330  386  456  642  302  757  705 1072  375 1018
  341  416  708  662 1002  462  781  496  630  512  534  762  890  494
  550  383  882  615  470  477 1260  773 1124  324 1497  531  389  712
  561  816  898  581  851  136  225 1243  866  720  280  232  602  522
  457  808  356  283  467  506  243  525  445  514  374  378  277  856
  646  511  285  256  466  314  382  212  872  508  558  758  812  504
  631  332  747  372  434  538  718  818 1035  431  648 1668  443 1215
 1647 1017  589  937  553  424  499  200  603  700  107  318  656  992
  533 1006  478 1339  600  291  679 1065 1037  715  790 1099 1128  304
  690  749  680  376  783 1029  919  368   70  922  622 1366 1122  752
  886  552  850  432  960  874 1106  772  483 1028  326 1047  532  791
 1401  344  935 1444  452 1105  354  268  984 1546  640 1460  396 1022
 1126  924  340 1684  794 1454  316  580  198  556  540  938 2568 1058
 1088 1696  404 1934  295 1550 1618  363  636  659  902 1256 1438 2334
  528  867  542  526  947  480  908  450 1020  616  820  311  100  847
  203  303 1080  442  178 1509  412   26  202 1227 1206 1077  464  214
  266 1004  954  362 2200   90   40  239 1354  593 1297 1974 1349  353
  548 1505  427 2572  371 1245 1400  638 1021 1161  709  584 1301  207
  620  627  408  663  980  419 1442  449  377  891  349 1138  262  801
  385  557 1044  503  327  828 1008  977  321  618  948  981  643  976
  860 1267  359  244  320  284 1175  879  335  745 1414 1385 1147  964
 1060  826  687  488  728  736 1331  737 1191  296  765 1390 1278  190
  990  963  626  906  565  482 1054  900  674  150   75  395  848  465
  469  864  873  328  554  596  120  675]
['아파트' '상가']
['경상남도' '대전광역시' '경기도' '전라북도' '강원도' '광주광역시' '충청남도' '부산
광역시' '제주특별자치도'
 '울산광역시' '충청북도' '전라남도' '경상북도' '대구광역시' '서울특별시' '세종특별자
치시']
['국민임대' '공공임대(50년)' '영구임대' '임대상가' '공공임대(10년)' '공공임대(분납)'
'장기전세' '공공분양'
 '행복주택' '공공임대(5년)']
[ 33.48  39.6   46.9   51.97  30.95  30.99  41.11  41.39  41.58  46.36
  51.24  39.72  51.93  59.88  36.55  36.62  39.62  46.73  46.81  46.95
  51.78  51.95  33.38  39.46  46.66  46.87  46.88  51.96  39.63  46.96
  59.89  51.88  51.49  59.64  37.67  42.9   49.44  26.37  31.32  32.1
  72.16  39.98  41.55  12.62  17.4   22.89  23.13  23.25  27.75  28.19
  34.8   42.35  42.4   49.37  55.17  55.5   36.44  45.09  45.32  26.34
  30.48  19.69  27.12  32.54  36.43  46.89  49.99  51.14 317.17  36.02
  36.38  45.88  39.99  49.95  51.51  51.59  51.9   59.91  59.94  59.99
  52.74  31.84  63.68 137.49  39.69  51.86  51.91  59.57  40.32  38.
  37.26  37.41  37.49  37.95  38.04  39.33  54.51  54.61  54.91  75.98
```

109.11	583.4	39.3	126.65	19.	14.1	19.31	21.19	22.95	23.4
27.23	31.85	32.29	35.13	36.47	50.08	240.22	39.65	46.47	51.84
51.92	54.6	49.69	39.74	49.62	16.	14.17	18.9	20.52	22.97
25.88	27.3	29.62	30.62	35.91	44.63	46.8	36.65	36.98	46.86
46.98	36.	35.28	18.98	19.36	21.46	22.83	27.55	29.17	31.92
32.6	36.57	72.26	30.	21.85	21.94	21.98	22.24	23.35	25.98
26.53	28.38	28.45	33.39	33.51	36.76	52.5	401.5	39.41	39.43
15.	19.25	23.88	23.91	24.38	27.5	28.88	248.56	59.63	16.57
18.38	20.9	26.25	28.5	33.15	33.31	40.39	407.97	61.89	404.65
39.	39.39	48.9	49.72	36.8	45.48	45.98	39.54	46.79	51.77
36.2	49.5	49.57	19.15	82.92	36.63	39.51	46.18	46.71	39.57
33.4	36.7	36.73	36.74	46.74	46.83	51.73	36.52	46.94	51.72
35.1	43.23	26.47	33.96	36.96	36.99	46.4	33.35	59.23	59.73
46.54	36.92	39.89	36.66	36.94	46.03	46.46	59.48	59.69	36.37
46.72	51.44	51.79	39.48	33.8	39.9	46.7	46.75	59.61	36.16
46.61	51.75	26.95	36.78	36.87	46.48	51.94	46.69	39.82	39.49
46.59	51.66	29.91	39.96	46.56	46.77	39.77	46.22	46.76	39.85
51.89	36.19	36.23	36.3	39.84	49.51	36.45	36.72	46.21	59.85
59.92	59.95	36.22	39.15	39.73	59.83	39.68	46.19	51.67	51.63
59.65	46.93	51.6	59.96	51.81	59.29	51.85	36.79	51.64	51.98
29.96	46.06	50.83	46.45	36.97	46.97	51.99	51.87	59.93	46.85
59.47	39.59	36.91	46.57	36.61	46.29	49.7	59.58	46.53	46.82
45.53	39.42	39.45	39.93	59.78	59.84	47.4	46.91	53.26	36.93
36.64	33.7	51.61	46.11	39.5	49.63	33.95	39.8	46.99	39.88
41.99	39.56	39.66	45.79	59.86	51.71	46.84	36.5	36.9	51.26
51.82	33.52	59.59	59.87	33.82	33.91	39.52	39.91	39.94	51.76
51.83	39.25	29.95	36.81	46.92	26.9	55.88	33.55	46.43	51.55
26.91	55.7	55.74	26.88	46.42	59.4	74.97	84.95	84.97	84.99
46.49	36.77	51.35	33.76	46.25	46.51	26.12	39.64	39.86	51.3
39.83	75.56	84.64	84.98	23.32	29.33	29.89	36.84	46.5	23.93
26.67	36.75	51.25	36.05	36.08	46.01	46.05	21.78	29.43	26.99
39.97	36.88	51.69	51.32	26.68	33.11	29.72	36.89	26.66	36.56
21.63	51.56	33.42	46.34	21.81	26.44	29.85	29.9	51.68	33.92
41.64	41.96	74.42	74.92	84.53	84.86	74.89	84.83	84.92	84.96
59.97	74.9	59.9	74.91	84.72	84.94	74.8	84.05	84.39	46.52
36.03	23.76	41.83	41.94	41.97	75.84	75.99	26.69	36.86	36.95
26.97	51.5	29.79	36.83	29.99	29.66	36.59	59.72	36.85	46.78
21.97	26.98	51.57	29.68	51.29	59.67	67.86	67.88	74.93	74.94
36.71	59.68	59.8	59.98	33.83	46.63	29.81	36.42	36.48	26.89
49.81	26.7	26.86	36.68	26.81	66.85	66.94	66.96	74.73	74.98
29.13	29.8	26.92	33.97	74.85	29.71	46.41	46.17	36.06	36.07
46.39	55.61	21.72	26.79	36.53	24.95	74.72	26.85	29.92	21.95
24.76	24.77	37.59	59.45	37.98	21.56	26.62	26.8	33.88	24.72
24.79	26.83	37.7	26.04	26.07	26.24	42.57	42.65	43.07	24.97
23.86	23.89	43.9	24.86	24.96	84.78	36.69	23.48	37.64	45.71
24.98	16.99	26.54	26.58	44.78	44.96	31.65	24.74	21.54	21.9
26.27	57.53	57.87	26.73	37.43	37.6	84.9	51.05	51.08	59.07
49.76	74.6	74.7	84.67	74.55	74.65	84.74	16.91	74.84	84.88
21.43	23.54	26.6	36.17	16.84	29.76	44.83	21.86	16.85	26.65
36.46	33.09	46.65	24.43	24.71	26.94	33.94	16.75	26.4	26.49
36.14	16.67	26.11	26.26	26.82	37.01	37.17	46.16	24.75	29.26
46.27	16.8	16.97	26.72	36.4	74.96	37.29	84.91	21.96	21.99
16.87	16.71	16.34	16.9	26.78	16.92	26.61	26.64	16.02	16.27
16.29	26.17	26.19	59.31	59.36	59.66	21.88	39.78	16.76	21.84
44.97	16.89	16.64	26.51	26.96	37.04	27.82	37.77	29.34	37.78
16.95	44.56	26.52	29.53	37.44	26.42	24.83	49.2	54.95	39.79
51.46	59.46	33.61	33.72	51.48	39.7	39.87	49.94	47.41	9.96
13.07	13.59	14.25	253.71	13.77	22.91	23.79	24.19	28.69	28.93
39.37	39.12	32.76	42.48	13.02	18.54	19.08	22.28	27.57	32.21
32.46	34.86	35.76	36.51	59.25	46.67	39.75	59.17	46.37	51.16
59.42	45.8	46.55	49.91	59.37	59.44	33.13	41.05	51.45	29.41

```
   36.39  55.85  33.24  55.83  29.73  51.52  59.24  47.    39.09  49.11
   74.95  46.64  49.86  84.84  33.46  26.32  46.38  29.75  21.93  74.82
   84.69  84.89  84.76  21.92  75.9   16.69  36.31  44.16  44.29  17.23
   26.59  33.53  24.59  24.66  16.39  16.42  26.41  36.32  59.18  59.19
   59.51  20.59  22.59  26.93  19.32  34.15  38.2   38.25  38.28  44.02
   16.78  26.75  29.39  43.33  24.54  16.94  26.56  26.76  37.79  37.45
   46.33  33.84  24.87  24.99  22.86  29.19  46.23]
 [ 276   60   20   38   19  106   26  288   68   34  148   74   70  170
   62  120  207   96  160   52  228  196  246  230   28   13  194   15
   86  275  126  168  219   98  144   76   75  232   83  190   43   46
   23   78   30   94   97   10   40  517  179   58  178  360  150   89
  354  180  298    1  313   92  225  143  294  149  557  192   90  237
  267  268  119   35  589  118  890  141    9    4  960  157  588  450
  204  420   69   63  251  171  324 1865  478  191  358  258  277  326
  200  130  252   44   88   91  198   99  447  596  745 1490  357  445
  382  239 1074  886  464 1192 1470  220  454  166  210  435  566  187
  142  223  446  606  336  136   61   55  110   84   39  156  195   36
   71  304  600  240  264   65  104  468  128  174  280  414  108   45
   80   54  216   22   12   51  102  310   42  134   24    8  188  202
   82  259   87  176  117   72   14  516   41  116  124  140  139  203
   73  340   56   79  245   95   18  684   25  262  369  152  135  496
  184  121  122  115  244  100  186   16   48  215  131  132  254  165
  289  172  189   33  105   27  291  510  177  260  145  300  432    2
   85   50  159  398  208  234  480  151  419   64  123  236  224  422
   31   66  255  129  284  221  377  133   81  287  167  213  217  372
  470  125  443  401  233  238   67  162  512  209    6  112  114  111
  250  147  257  161    5  292   29  442  212  253   59  603   32    3
  127  218  279   37  231  330  406  155  153  263  242  302  347  113
   53  103  379   49  164   57  315  154  175    7   11  453  486  286
  227  222  109   47  314  322  388  270  282  318  296  416  205  241
  299  201  261  396  305  295  146  339  308   17  332  448  410  488
  415  107  197  271   93  634  274  182  762  342  438  214  604  540
  317  137  269  350  343  138  408  366  272  536  580  556  574  532
  572  266  628  620  158  193  375  515  368  498  616  728  472  590
  430  344  625  726  508  660  469  290  656  548  370  456  395  376
   21  385  341  672  256  390  199  404  471  491  645  229 1341  894
  573  462  345  293  306  492  312  424  384  235  169  428  740  427
  473  348  346  564  163  367  320  658  582  440   77  528  460  434
  206  297  400  349  338]
[17. 13.  6. 14.  9. 10. 15.  8.  0. 19. 12.  2.  3.  4. 26.  1. 34.  7.
 21. 11. 25. 29. 23. 32.  5. 16. 20. 28. 22. 18. 24. 27. 30. 36. 33. 31.
 39. 37. 35. 43. 40. 38. 42. 46. 49. 47. 41. 55. 45.]
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15]
['9216000' '12672000' '18433000' ... '6872000' '13410000' '18689000']
['82940' '107130' '149760' ... '106400' '144600' '166500']
[ 0.  1. nan  2.  3.]
[ 3.  1.  2.  6. 10.  5.  4.  7. 12. 14.  8.  0. 20. 11. 16. 15. 19. 13.
 50. 18.]
[ 624. 1285.  734.  645.  517. 1483.  634.  288.  530.  168.  631.  222.
  117.  296.  527.   97.  616.  154.  509.  736.  277.  420.  548.  407.
  162.  287.  986.  150.  487.  264.  226.  178.  107.  612.  804.  200.
  262.  405.  351.  375.  240.  166.  217. 1043.  756.  270.  594.  109.
  120.  190.   88.  486.  682.  402. 1296.  131.  393.  376.  155.  235.
  950.  153.  192.  246.  128.  164.   65.  477. 1505.  671.  802.  738.
  198.  374.   54. 1299.  823.  763.  579.  478.  928. 1181.  658. 1082.
 1240.  421.  985.  219.  900.  521.  399.  205.  316.  472.  837. 1026.
 1119.  354.  384.  853.  895.  330.  347.  608.  573.  668.  574.  224.
  483.  480.  958.  508.  873.  827. 1137. 1488.  259.  299.  567.  660.
  502.  894.  454.  862.  788.  638.  546.  775.  325.  663.  428.  637.
  362.  426.  465.  403.  661.  882.  665.  358.  492.  385.  834.  755.
  413.  425.  941.  334.  335.  779. 1129.  448. 1100.  291.  352.  572.
```

```
 822.  909.  771.  717.  136. 1016.  692.  317.  581.  280.  195.  715.
 525.  537.  728.  641.  761.  550.  227.  436.  214.  422.  404.  281.
 253.  265.  777.  583.  199.  329.  241.  306.  163.  733.  370.  596.
 442.  449.  564.  445.  258.  817.  360.  698.  640.  911.  319.  366.
 438.  812. 1050.  367.  600.  903. 1756. 1105. 1741.  387.  707.  879.
 515.  434.  770.  772. 1039.  584.  467.  409.  506.  592.   27.  248.
 845.  871.  484.  915.  629. 1590.  400.  528.  257.  615.  936.  582.
 673. 1154.  321.  951.  230.  652.  694.  759.  877.  380.  978. 1061.
 988.  285.   50.  731.  551. 1192. 1182.  680.  809.  523.  948.  349.
 876.  722.  752.  700.  437.  973.  268. 1066.  944. 1636.  344.  939.
1275.  461. 1110.  297.  865.  535. 1176.  549. 1534.  237. 1202.  983.
 320.  750. 1689.  676.  880.  239.  378.  490.  500. 1117. 1798.  956.
 333.  773. 1648. 1670.  167. 1319. 1382.  309.  635.  878. 1500. 1713.
 543.  897.  542. 1201.  727. 1306.  338. 1001.  318.  423.  719.  350.
 232.  183.   57.  496.  397.  249. 1142.  533.  995. 1055.  114.   13.
 141.  857.  408.  842. 1239.  108.  188.  389.  345.  664. 1493. 1570.
  66.   25.   30.  683. 1216.  547. 1112. 1696. 1098.  470.  418.  210.
 491.  187.  840.  308.  447.  185.  173.  571.  593.  499. 1101.  152.
 498.  841.  458.  516.  785. 1225.  562.  382.  805. 1141.  202.  611.
 159.  893.  274.  585. 1035.  507.  741.  545.  831.  989.  392.  646.
 711.  884. 1280.  254.  211.  236.  213.  379.  286. 1024.  278.  705.
1307. 1139.  602. 1222.  818.  925.  746.  372.  690.  803.  704. 1204.
 632. 1052.  451.  910.  980. 1014.  142.  701.  685. 1161.  566.  906.
 824.  339.  633.  674.   29.  433.  675.  322.  603.  610.  667.   53.
 250.   40.  300.]
```

In [31]: 라벨 인코딩 수행.

```
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, '세종특별자
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[31]:

	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	
1004	C2189	382	아파트	전라북도	국민임대	29.19	96	45.0	8	6872000	106400	0.0	2.0	300.0	1	

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

In [32]:

```
all_df.단지코드.unique()
```

Out[32]:

```
['C2515', 'C1407', 'C1945', 'C1470', 'C1898', ..., 'C2456', 'C1266', 'C2152', 'C126
7', 'C2189']
Length: 561
Categories (561, object): ['C2515', 'C1407', 'C1945', 'C1470', ..., 'C1266', 'C215
2', 'C1267', 'C2189']
```

In [33]:

```
all_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3904 entries, 0 to 1007
Data columns (total 17 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   int64   
 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   
dtypes: category(1), float64(5), int32(1), int64(5), object(5)
memory usage: 615.3+ KB
```

In [34]:

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

In [35]:

단지코드도 라벨 인코딩

```
all_df['단지코드_lbl'] = all_df['단지코드'].cat.codes
all_df
```

Out[35]:

	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	
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	

	단지코드	총세대수	임대건물구분	지역	공급유형	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	_l
1007	C2189	382	아파트	전라북도	국민임대	46.23	60	45.0	8	18689000	166500	0.0	2.0	300.0	1	

3904 rows × 18 columns

In [36]:

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

Out[36]:

	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지역_lbl	공급유형_lbl	단지코드_lb
0	C2515	545	33.48	276	17.0	1	9216000	82940	0.0	3.0	624.0	1	1	1	49
1	C2515	545	39.60	60	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	49
2	C2515	545	39.60	20	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	49
3	C2515	545	46.90	38	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	49
4	C2515	545	46.90	19	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	49
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
1003	C1267	675	36.77	126	38.0	12	-	-	0.0	1.0	467.0	1	1	9	8
1004	C2189	382	29.19	96	45.0	8	6872000	106400	0.0	2.0	300.0	1	4	1	38
1005	C2189	382	29.19	20	45.0	8	6872000	106400	0.0	2.0	300.0	1	4	1	38
1006	C2189	382	39.45	202	45.0	8	13410000	144600	0.0	2.0	300.0	1	4	1	38
1007	C2189	382	46.23	60	45.0	8	18689000	166500	0.0	2.0	300.0	1	4	1	38

3904 rows × 15 columns

In [37]:

```
for c in all_df_last.columns:
    print(all_df_last[c].unique())
```

```
['C2515', 'C1407', 'C1945', 'C1470', 'C1898', ..., 'C2456', 'C1266', 'C2152', 'C126
7', 'C2189']
Length: 561
Categories (561, object): ['C2515', 'C1407', 'C1945', 'C1470', ..., 'C1266', 'C215
2', 'C1267', 'C2189']
[ 545 1216  755  696  566 1722  624  361  754  240  688  623  601  405
  518  619  875  560  595  625  880 1396  970  606 1507  639  965  946
  785 2424  809  388 1013  453  861  806  903 1116 1486 1957  802 1527
 1005 1988 1350 2428 1755  840  390  451  410  779  693  753  498 1533
  711  420  590  657  495  460  338 1144  521 1003  306  697  213  481
  468 1364  800  830  775  261  474   72 1473  996  870  678  632  961
 1232  676 1300  998  493 1117  307  501  896  458  290  409  586 1084
 1174  270 1308  355  384  853  492  901  815  312  571  594  944  635
  962  822 1129 1479  330  386  456  642  302  757  705 1072  375 1018
  341  416  708  662 1002  462  781  496  630  512  534  762  890  494
  550  383  882  615  470  477 1260  773 1124  324 1497  531  389  712
  561  816  898  581  851  136  225 1243  866  720  280  232  602  522
  457  808  356  283  467  506  243  525  445  514  374  378  277  856
  646  511  285  256  466  314  382  212  872  508  558  758  812  504
```

In [38]:

```
train.shape, test.shape
```

Out[38]:

```
((2896, 15), (1008, 14))
```

In [39]:

```
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
```

결측치 처리 후, 다시 원래로 데이터 분리

Out[39]:

	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지역_lbl	공급유형_lbl	단지코드_lb
0	C2515	545	33.48	276	17.0	1	9216000	82940	0.0	3.0	624.0	1	1	1	49
1	C2515	545	39.60	60	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	49
2	C2515	545	39.60	20	17.0	1	12672000	107130	0.0	3.0	624.0	1	1	1	49
3	C2515	545	46.90	38	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	49
4	C2515	545	46.90	19	17.0	1	18433000	149760	0.0	3.0	624.0	1	1	1	49
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
2891	C2532	239	49.20	19	7.0	1	11346000	116090	0.0	1.0	166.0	1	5	1	50
2892	C2532	239	51.08	34	7.0	1	14005000	142310	0.0	1.0	166.0	1	5	1	50
2893	C2532	239	51.73	34	7.0	1	14005000	142310	0.0	1.0	166.0	1	5	1	50
2894	C2532	239	51.96	114	7.0	1	14005000	142310	0.0	1.0	166.0	1	5	1	50
2895	C2532	239	54.95	19	7.0	1	14830000	151030	0.0	1.0	166.0	1	5	1	50

2896 rows × 16 columns

In [40]:

```
train_df.columns
```

Out[40]:

```
Index(['단지코드', '총세대수', '전용면적', '전용면적별세대수', '공가수', '자격유형',
'임대보증금', '임대료',
       '10분내지하철수', '10분내버스정류장수', '단지내주차면수', '임대건물구분_lbl',
'지역_lbl', '공급유형_lbl',
       '단지코드_lbl', '등록차량수'],
      dtype='object')
```

In [41]:

```
from sklearn.model_selection import train_test_split
```

In [42]:

```
sel = [ '총세대수', '전용면적', '전용면적별세대수', '공가수', '자격유형', '10분내버스정류장수',
       '단지내주차면수', '임대건물구분_lbl', '지역_lbl', '공급유형_lbl',
       '단지코드_lbl']   변수 선택. 데이터 나누기

X = train_df[sel]
y = train_df['등록차량수']
test_X = test_df[sel]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                    random_state=0)
```

In [43]:

```
from sklearn.linear_model import LinearRegression
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

In [44]:

```
model = LinearRegression()   회귀 모델.
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) : 0.7957809911189807
테스트(score) : 0.7988974846133388
```

In [45]: MAE, MSE, RMSE

```
mae_val = np.mean( abs( y_test - pred ) )
print( mae_val )
mse_val = np.mean( (y_test - pred) **2 )
print( mae_val )
rmse_val = mse_val ** 0.5
print( rmse_val )
```

```
142.33218947810167
142.33218947810167
201.661938023843
```

In [46]: 랜덤포레스트 모델

```
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) : 0.9985135924575252
테스트(score) : 0.9935152117632033
```

In [47]: MAE, MSE, RMSE

```
mae_val = np.mean( abs( y_test - pred ) )
print( mae_val )
mse_val = np.mean( (y_test - pred) **2 )
print( mae_val )
rmse_val = mse_val ** 0.5
print( rmse_val )
```

```
16.22971231300347
16.22971231300347
36.212881581076395
```

In [48]:

```
model = RandomForestRegressor(n_jobs=-1)
model.fit(X_train, y_train)
pred = model.predict(test_X)
pred[0:10]
```

Out[48]:

```
array([ 678.32,  722.15,  680.15,  680.15,  676.93,  676.93,  691.27,
        688.1 , 1494.3 , 1496.67])
```

In [49]: 제출을 위해서 단지별 차량수 평균을 추가하기.

```
test_df['등록차량수'] = 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-49-3cc158f3592c>: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'] = pred
<ipython-input-49-3cc158f3592c>: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[49]:

	index	단지코드	총세대수	전용면적	전용면적별세대수	공가수	자격유형	임대보증금	임대료	10분내지하철수	10분내버스정류장수	단지내주차면수	임대건물구분_lbl	지역_lbl
0	0	C1072	754	39.79	116	14.0	8	22830000	189840	0.0	2.0	683.0	1	3
1	8	C1128	1354	39.79	368	9.0	8	22830000	189840	0.0	3.0	1216.0	1	3
2	17	C1456	619	33.40	82	18.0	1	19706000	156200	0.0	16.0	547.0	1	8
3	26	C1840	593	39.57	253	7.0	1	14418000	108130	0.0	3.0	543.0	1	4
4	30	C1332	1297	39.99	282	11.0	8	28598000	203050	0.0	2.0	1112.0	1	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
142	982	C2456	349	26.44	24	17.0	8	6992000	117000	0.0	4.0	270.0	1	9
143	986	C1266	596	26.94	164	35.0	8	8084000	149910	0.0	1.0	593.0	1	11
144	991	C2152	120	24.83	66	9.0	3	-	-	0.0	1.0	40.0	1	5
145	993	C1267	675	24.87	28	38.0	8	6882000	104370	0.0	1.0	467.0	1	1
146	1004	C2189	382	29.19	96	45.0	8	6872000	106400	0.0	2.0	300.0	1	4

147 rows × 18 columns

In [50]: 테스트에 빠진 코드는 전부 0으로 처리하기.

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

Out[50]:

	code	num
0	C2675	0
1	C2335	0
2	C1327	0

In [51]: 이를 csv파일 만들기

```
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[51]:

	code	num
0	C1072	686.75
1	C1128	1493.12
2	C1456	603.13
3	C1840	587.66
4	C1332	1020.37
...	...	...
145	C1267	425.798
146	C2189	296.075
147	C2675	0
148	C2335	0
149	C1327	0

150 rows × 2 columns

In [52]:

```
sub_df.to_csv('third_rf_0714.csv', index=False)
sub_df.head()
```

Out[52]:

	code	num
0	C1072	686.75
1	C1128	1493.12
2	C1456	603.13
3	C1840	587.66
4	C1332	1020.37

In [53]:

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

Out[53]:

```
['.git',
 '.ipynb_checkpoints',
 '01_competition_firstmodel.html',
 '01_competition_firstmodel.ipynb',
 '01_competition_firstmodel.pdf',
 '01_대회_첫모델만들기.md',
 '02_second_data.md',
 '02_second_datapreprocessing.html',
 '02_second_datapreprocessing.ipynb',
 '02_second_datapreprocessing.pdf',
 '03_second_linear_model-Copy2.ipynb',
 '03_second_linear_model.html',
 '03_second_linear_model.ipynb',
 '03_second_linear_model.pdf',
 '03_second_linear_ridge_lasso.ipynb',
 '04_second_rf_model.ipynb',
 '05_second_etc_model.ipynb',
 'baseline_0712.csv',
 'fourth_rf_0714.csv',
 'README.md',
 'second_rf_0712.csv',
 'test_df.csv',
 'third_rf_0714.csv',
 'train_df.csv',
 'train_df_errno.csv',
 'Untitled.ipynb']
```

점수 : 138.65787