Multi Campus
2021.08.20

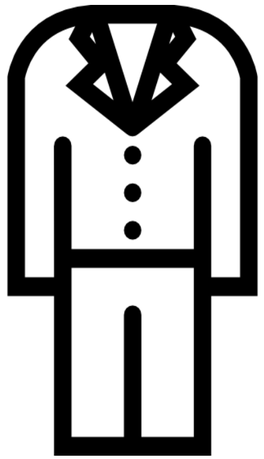
머신러닝을 이용한 서울 아파트 실거래가 예측

3조 SUITS

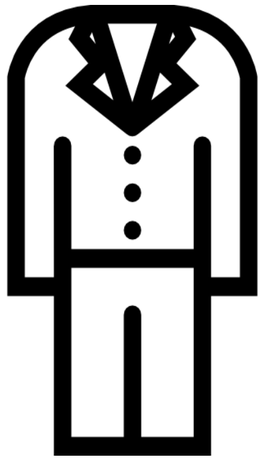
목 차

팀원 소개

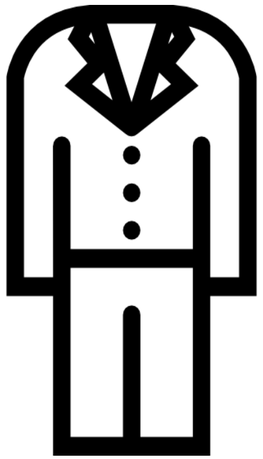
TEAM MEMBERS



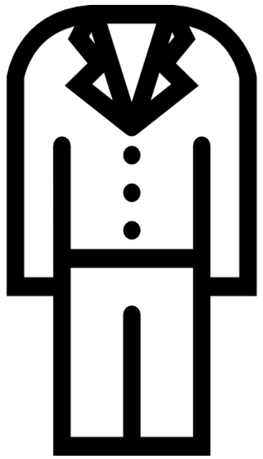
신다롬



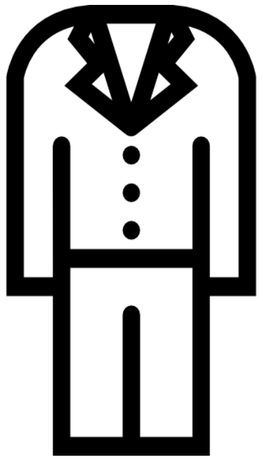
채길호



이민성



이우재



김보아

배경 및 목표

한국경제_2021.02.17

정부의 가계부채 노력에도
불구하고 시중에
떠도는 엄청난 현금성 자산

데일리안_2021.08.16

대표적인 주택거래 통계들이
역대 최고치를 갱신

한겨레 _2021.01.19

전국 청약 경쟁률이
최고치를 기록

데이터 목록

수집 데이터		출처
국토 교통부 실거래가 공개 시스템	거래가격 거래일자 아파트명 면적 층수 위치 (구, 동)	https://rt.molit.go.kr/
전국 병원 리스트 (상급병원/종합병원 데이터만 활용)	요양기관 명 종별코드 명 시도코드 명 주소	https://www.data.go.kr/data/15051059/fileData.do
서울 지하철 행정동 정보	역 명 행정동(법정동) 명	http://www.seoulmetro.co.kr/

데이터 설명

헤도닉 가격모형

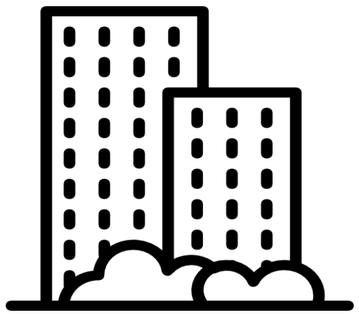
1. 각 주택별 환경질, 학군, 주택 크기 등 주택의 속성 등 변수 확보
2. 각 개인의 주택선택과 이들의 경제, 사회인구학적 변수 조사
3. 각 주택의 가격을 주택 특성 및 개인 특성 등에 회귀분석하여 헤도닉 가격함수 추정하고, 추정된 헤도닉 가격함수로부터 잠재한계가격 도출

$$P_i = P(Z_i) = P(z_{1i}, z_{2i}, z_{3i}, \cdots, z_{ni})$$

- 종속변수 : 주택가격
- 독립변수 : 주택의 속성
  - ✓ 주택의 구조적 특성: 평수, 층수, 방 수, 화장실 수, 노후연수, 정원, 주차장 존재 여부 등
  - ✓ 주변 환경 특성: 학군, 공원, 중심지 접근성, 주요 대중교통시설 접근성 등
  - ✓ 이웃의 사회인구학적 특성: 소득, 인종 등

입지적 특성 변수

위치(행정구, 행정동)
동별 지하철의 개수
동별 병원의 유무(0,1)



물리적 특성 변수

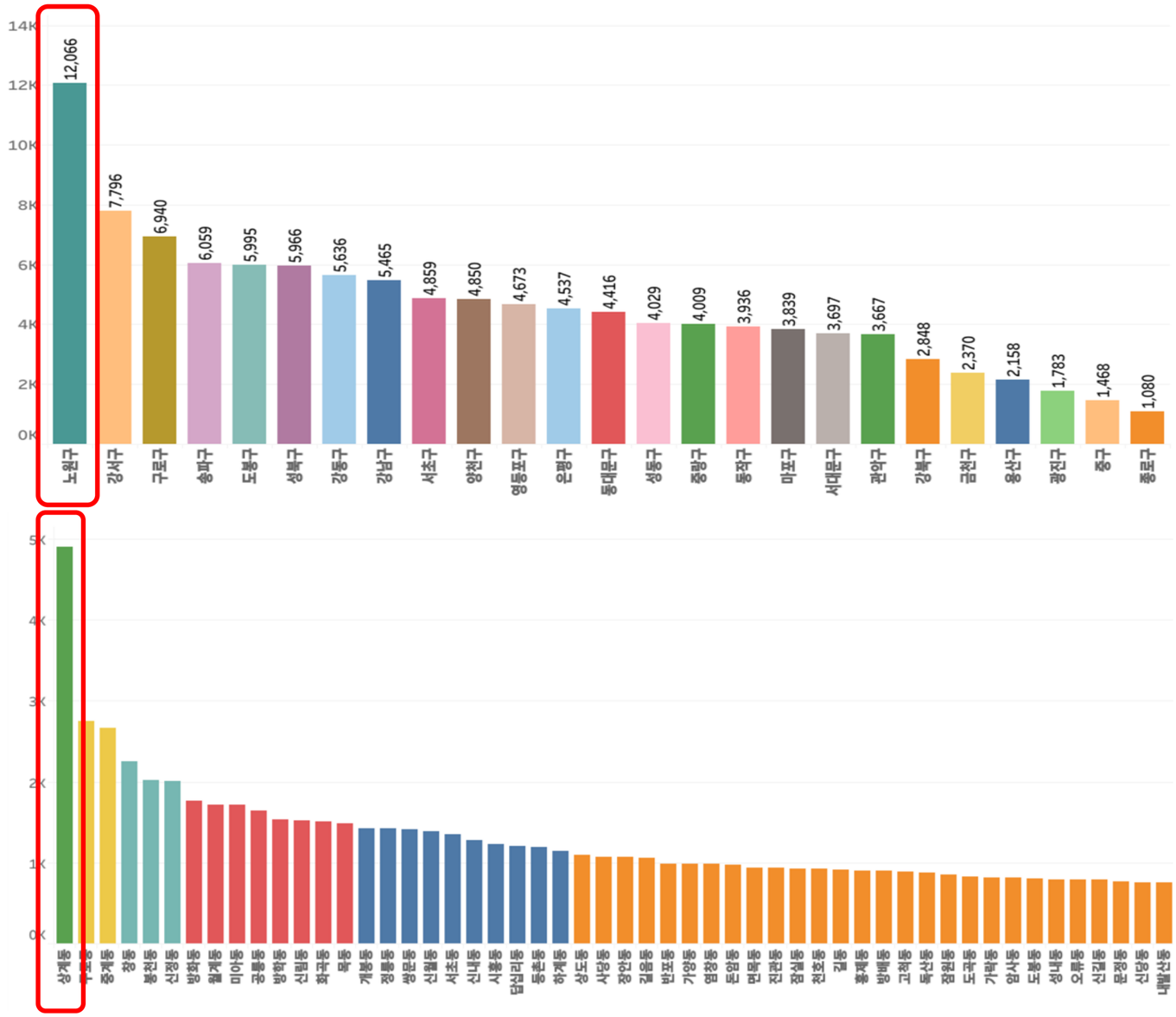
전용면적
층
건축 년도
연식
아파트명

입지적 특성 변수 →
물리적 특성 변수 →
머신러닝 모델 →
예측 실거래가

EDA

대출 규제를 강화한 6·17 대책 시행 전 규제를 피한 내 집 마련에 나선 실수요자의 가세로 인해 2020년 6월 거래가 급증

EDA



특정 구, 특정 동에 따라

거래량에 차이가 나타남을 확인

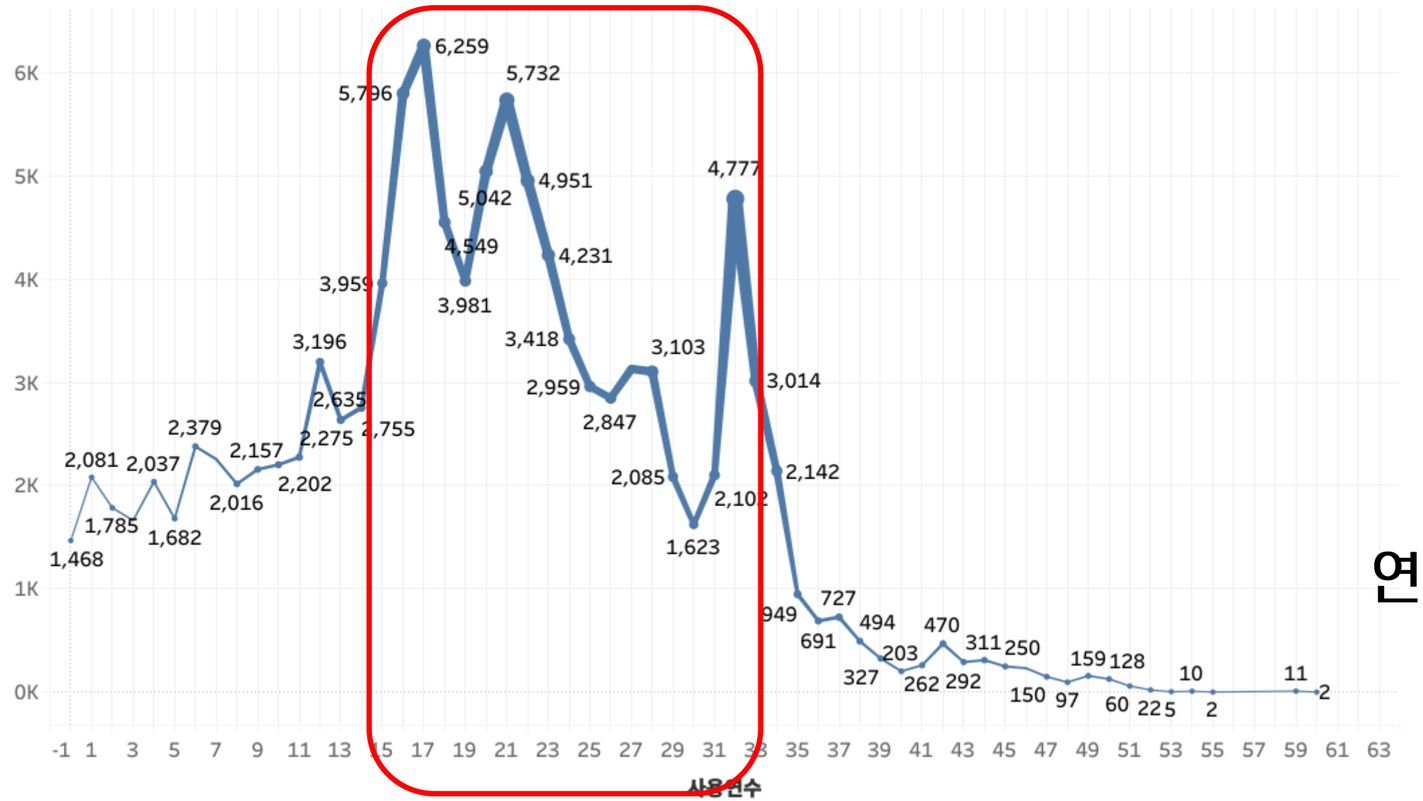
구기준 최다 거래지역 : 노원구

총 거래건수 : 12,066

동기준 최다 거래지역 : 상계동

총 거래건수 : 4899

EDA



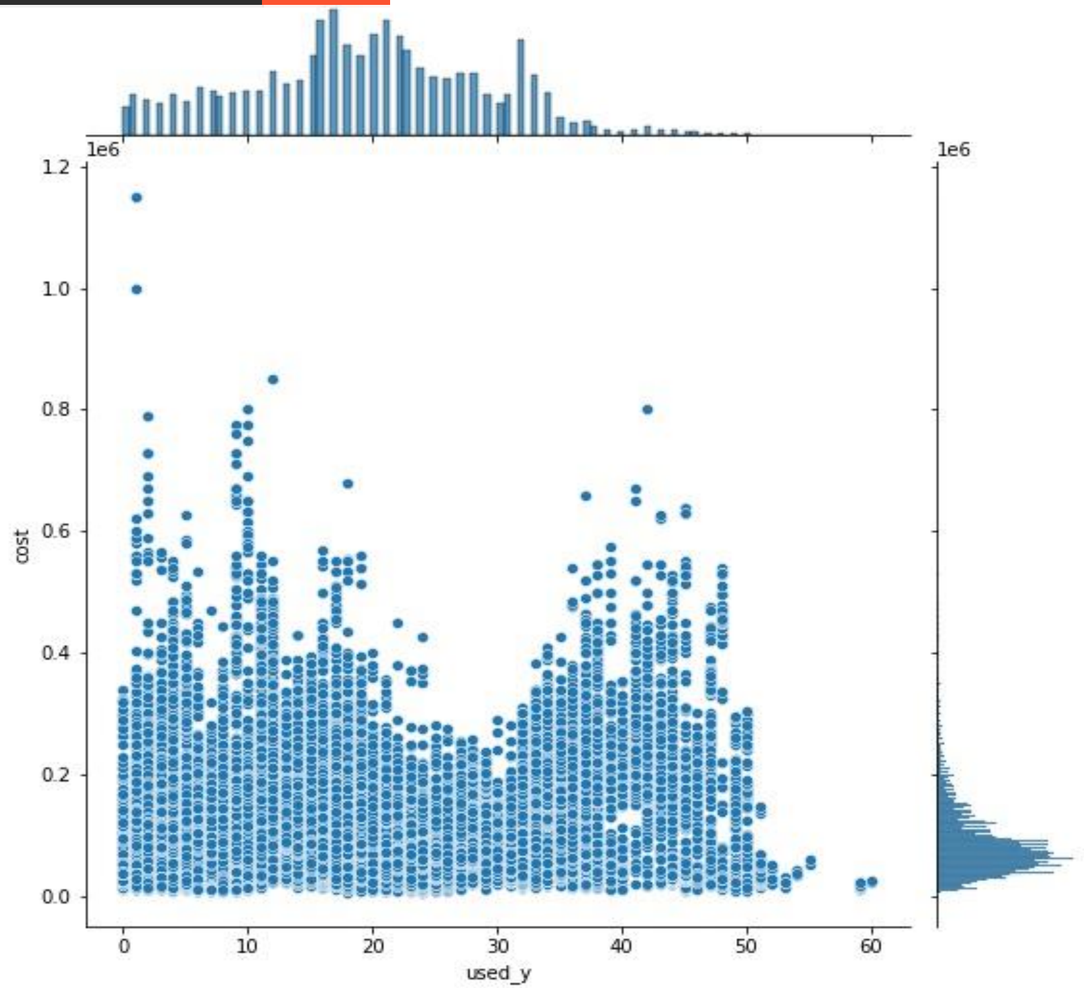
연식에 따라서도

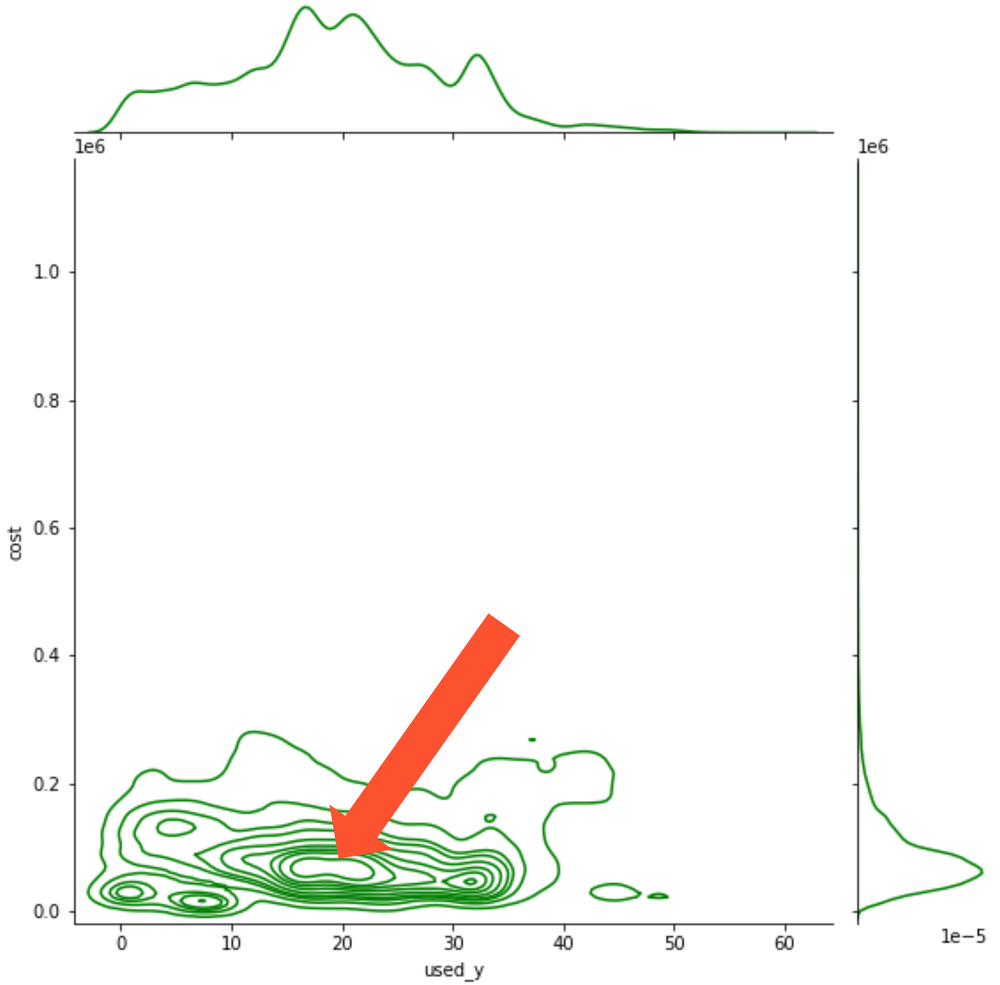
거래량의 차이가 나타남을 확인.

연식기준 최다 거래구간 : 15 ~ 30년 전후

최다 거래건수 : 6,259(17년식)

EDA

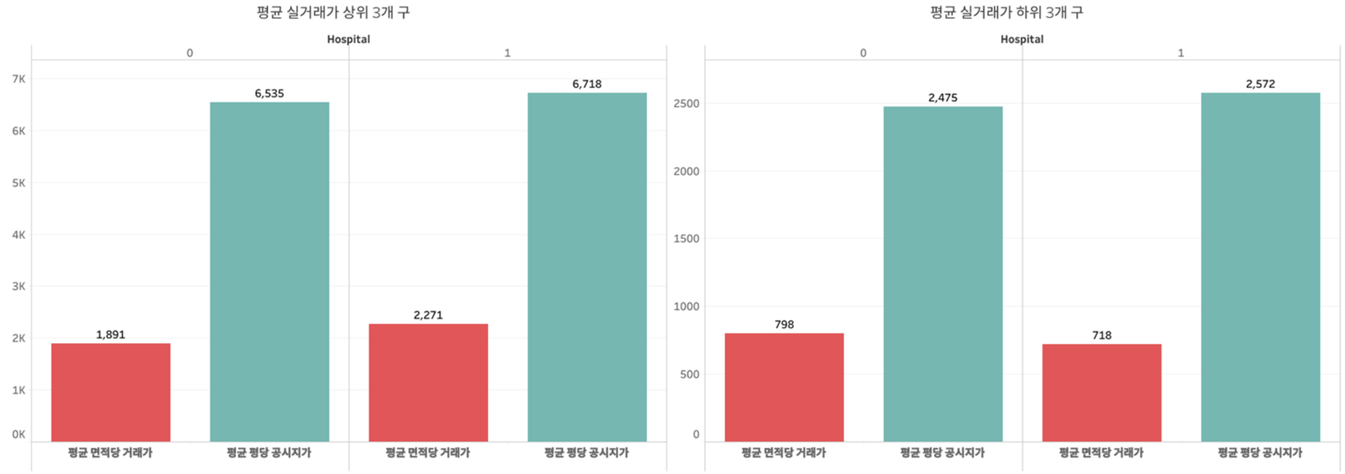




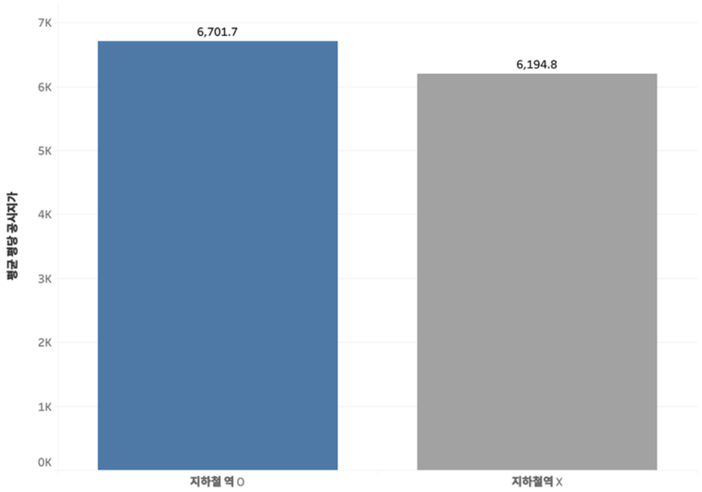
아파트 연식에 따른실제 거래 가격 분포 확인

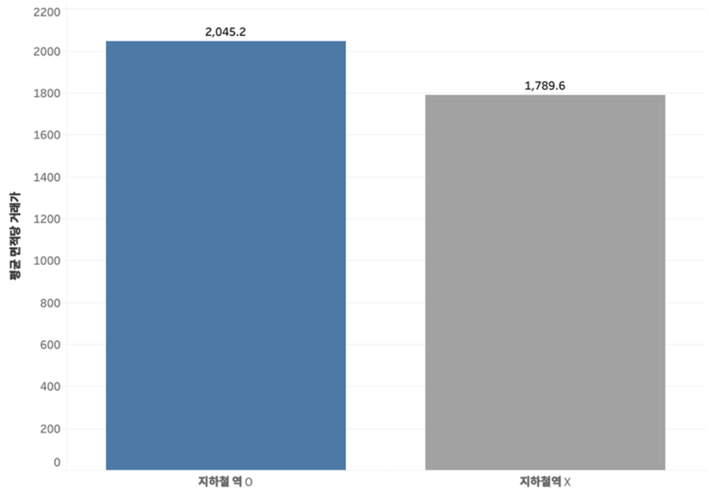
연식 16~23년, Cost 6억~9억 범위에서 거래건수 밀집도가 가장 높음을 확인

EDA



지하철, 병원 유무에 따른
평균 공시지가, 평당 거래가의 차이를 확인





병원이 있는 지역과 지하철이 있는 지역에서
평균 금액이 높게 나타남.

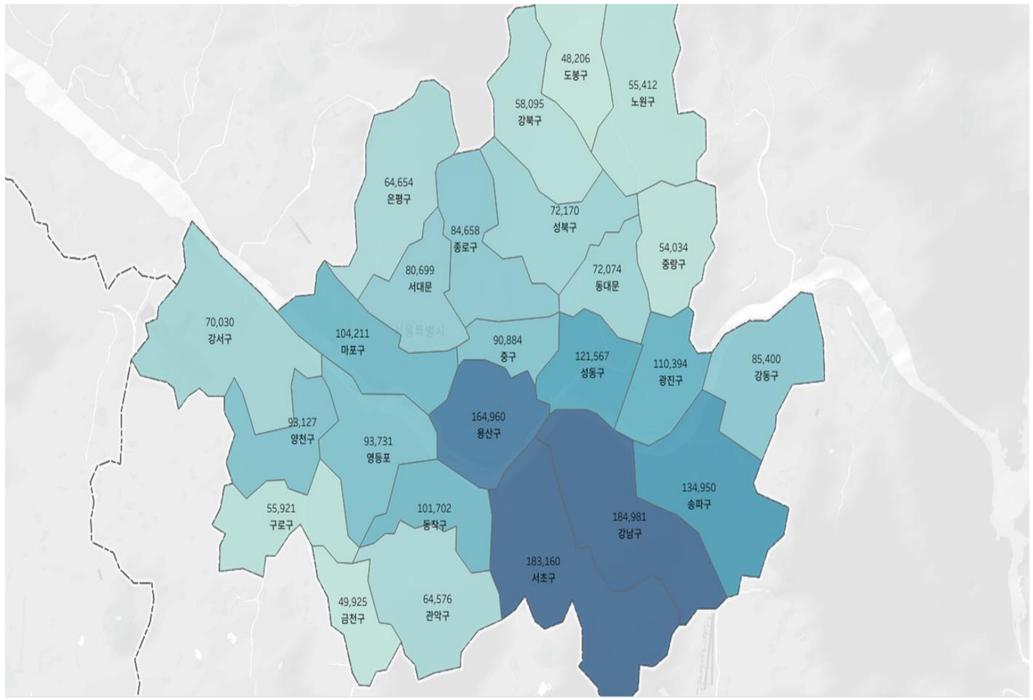
데이터 전처리

서울 아파트 매매 실거래가 가격

2020년 1월 ~ 2021년 7월

주소 분할 ➡ 구, 동

구별 평균 실거래가



건축 년도를 '연식' 으로 추출
지하 층수를 자연수화

- 연식 = -1 (청약인 경우) , floor < 0 인 경우 (지하 처리)
- 연식은 0 으로 초기화
- floor는 지하 층수를 자연수로 표현하기 위해
  최솟값 -3을 확인, +3을 해서 자연수로 변환함.

아파트 명 단어 분리

```
seoul_1['token'].value_counts()[:11]
```

```
          6551
상계주공고층     2392
현대          2315
주공          1618
신           1492
한신          1242
우성          1212
목신시가지       1057
두산           955
벽산           836
삼성래미안        766
Name: token, dtype: int64
```

데이터 전처리

아파트명 단위 상위 10개 추출

상위 10 = 1 나머지 = 0

```
top_list = ['상계주공고층', '현대', '주공', '신', '한신', '우성', '목신시가지', '두산', '벽산', '삼성래미안']
```

```
for i in top_list:
    seoul_1.loc[seoul_1['name'].str.contains(i), 'YN_top10'] = 1
```

지역구와 동 명 라벨링 LabelEncoder ()

문자열의 지역명을 숫자형태로 변환

```
노원구      12066
강서구       7796
구로구       6940
송파구       6059
도봉구       5995
성북구       5966
강동구       5636
강남구       5465
서초구       4859
양천구       4850
영등포구      4673
은평구       4537
동대문구      4416
성동구       4029
중랑구       4009
동작구       3936
마포구       3839
서대문구      3697
관악구       3667
강북구       2848
금천구       2370
용산구       2158
광진구       1783
중구        1468
종로구       1080
Name: gu, dtype: int64
```

```
8       12066
3        7796
6        6940
17       6059
9        5995
16       5966
1        5636
0        5465
14       4859
18       4850
19       4673
21       4537
10       4416
15       4029
24       4009
11       3936
12       3839
13       3697
4        3667
2        2848
7        2370
20       2158
5        1783
23       1468
22       1080
Name: gu_l, dtype: int64
```

데이터 전처리

동 기준 역 개수

해당 동 내
역개수를 원자값으로 하여
Station 컬럼 생성

	st	dong	st_r
0	역 명	행정동(법정동)명	
1	서울(1)역	회현동(봉래동2가)	서울
2	시청(1)역	명동(태평로1가)	시청
3	종각(1)역	종로1.2.3.4가동(종로1가)	종각
4	종로3가(1)역	종로1.2.3.4가동(종로3가)	종로3가

동 기준 병원 근접도 확인

해당 동 내
상급병원, 종합병원 유무를
0,1값으로 변환하여
hospital 컬럼 생성

	Gu	Dong	count
0	강남구	도곡동	1
1	강남구	역삼동	1
2	강남구	일원동	1
3	강동구	길동	1
4	강동구	둔촌동	1
5	강동구	상일동	1
6	강북구	수유동	1

데이터 전처리

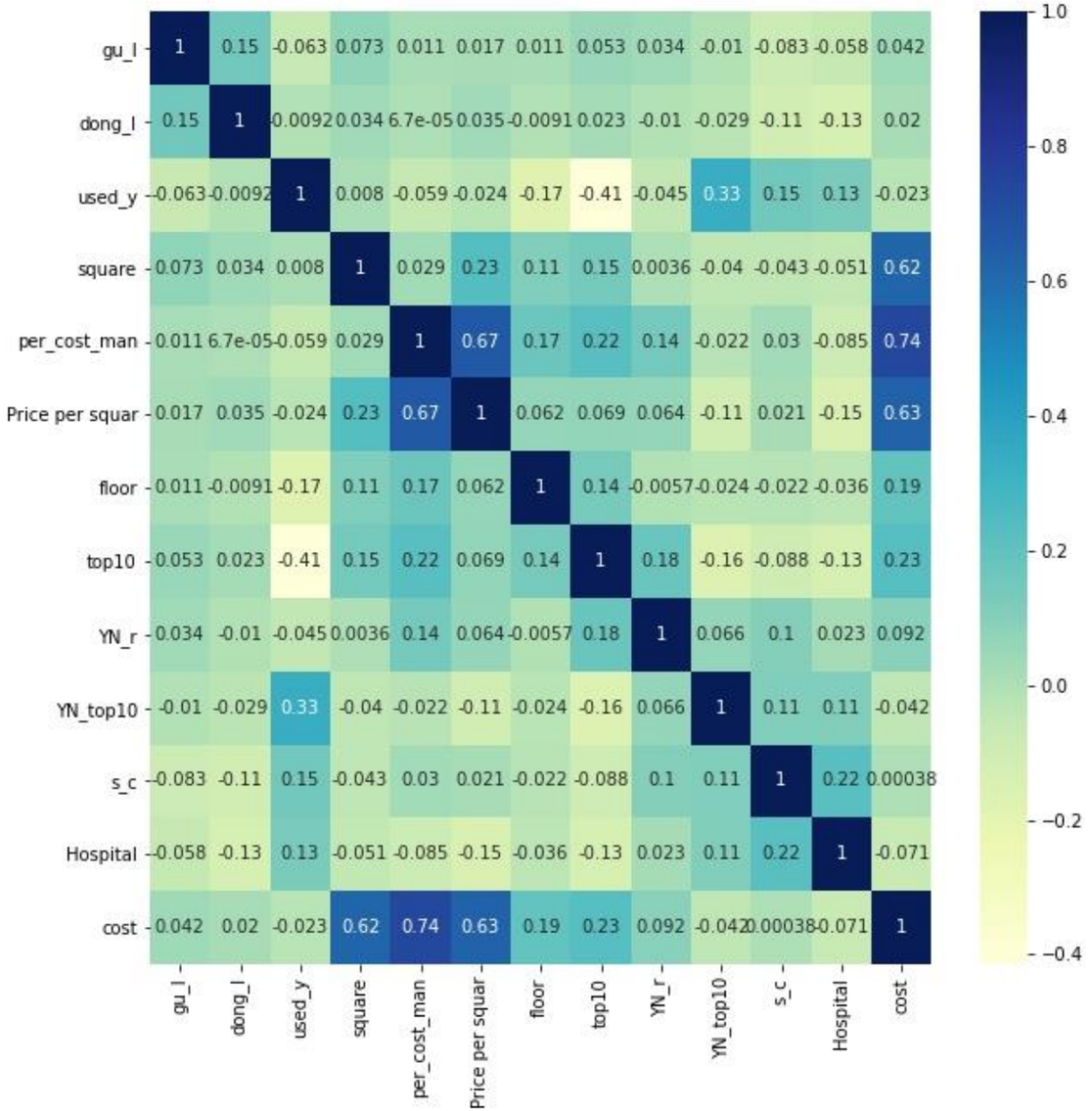
최종 데이터 셋 구성

Price per square : 평당 공시지가
per_cost : 해당 아파트 면적당 가격
top10 : 시공사 기준 상위 10개 유무
name_top10 : 서울 아파트 이름 빈도 수 상위 10개 유무

YN_r : 아파트 이름에 동 이름이 들어 있는지
Hospital : 해당 동에 병원이 있는 지
Station : 해당 동에 있는 지하철 역의 수
Transaction_real_price : 아파트 가격 - 타켓

Gu_Label	Dong_Label	Year	Exclusive_area	Floor	Price per squar	per_cost	top10	name_top10	YN_r	Hospital	Station	Transaction_real_pri
0	8	32	77.75	7	7003.020394	1922.829582	0	1	1	0	1	149500
0	8	33	54.98	7	7003.020394	2619.134231	0	1	1	0	1	144000
0	8	33	79.97	7	7003.020394	2035.138177	0	1	1	0	1	162750
0	8	33	79.97	8	7003.020394	2000.750281	0	1	1	0	1	160000
0	8	33	79.97	5	7003.020394	1875.703389	0	1	1	0	1	150000
0	8	33	67.28	5	7003.020394	2452.437574	0	1	1	0	1	165000
0	8	33	67.28	7	7003.020394	1783.590963	0	1	1	0	1	120000
0	8	33	67.28	5	7003.020394	2452.437574	0	1	1	0	1	165000

상관관계



피어슨 상관계수를 통한 상관관계 시각화

Price per square(구별 평당 공시지가)

Per_cost(해당 아파트 평당 가격)

Square(전용 면적)

Top10(아파트 브랜드)

Floor(층수)

5개 변수와 cost의 양의 상관관계를 확인

모델 비교&선정

적용 모델 목록

KNNregression

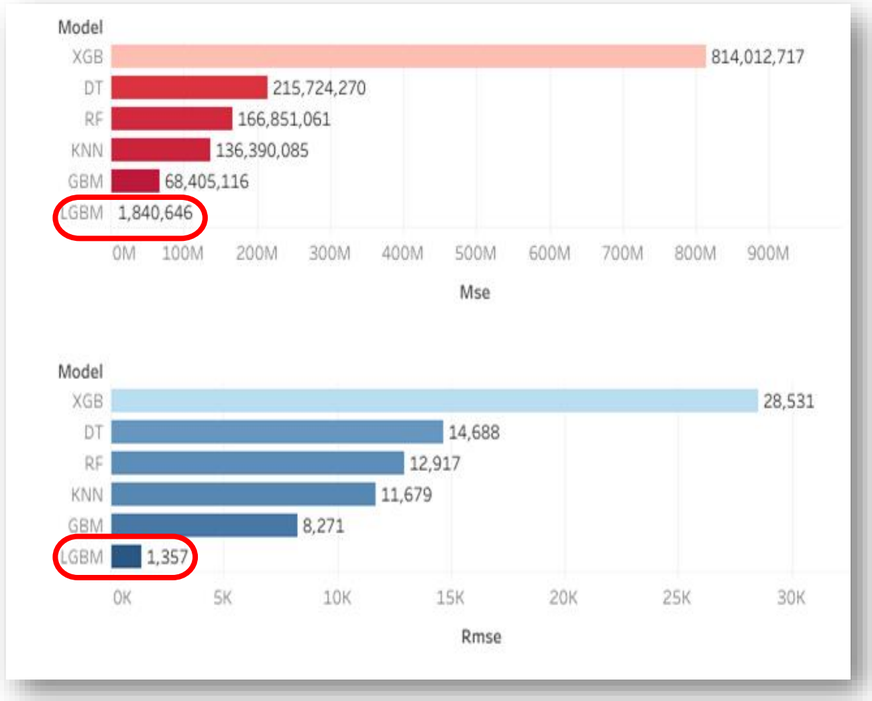
Decision Tree

Random Forest

GBM

LGBM

XGboost



각 모델 별 MSE & RMSE 시각화

XGB > DT > RF > KNN > GBM > LGBM

순으로 모델 성능 확인

평균 오차(MSE)가

가장 낮게 나오는

LGBM 모델 선정

모델 설명

Gradient Boosting 프레임워크로 Tree 기반 알고리즘
Boosting기법을 바탕으로하여 정답지와 오답지간의 차이를 반복적으로 학습하여 모델을 개선한다.

장점	단점
- 빠른 학습 속도 - 상대적으로 적은 메모리 사용량 - categorical feature들의 자동 변환과 최적 분할 - GPU 학습 지원	- 작은 Data set의 경우, 과적합 가능성이 있음

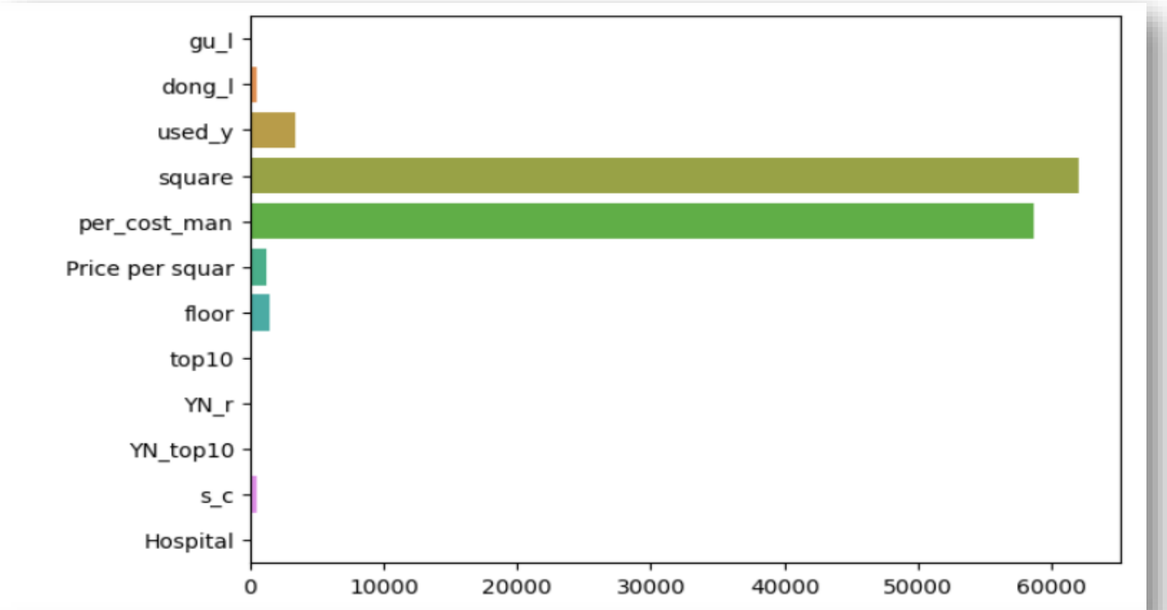
모델 설명

전체 데이터로 LGBM을 실행

```
%time
lgb = LGBMRegressor(linear_tree = True,
                    boosting_type = 'gbdt',
                    objective = 'regression',
                    n_estimators = 9000,
                    learning_rate = 0.001,
                    max_depth = 4,
                    n_jobs = -1)

lgb.fit(X_train, y_train)
```

```
MSE  :  1829758.059959414
RMSE  :  1352.685499278902 6
```




```
lgb.feature_importances_
```

```
array([   88,   566,  3361, 62039, 58621,  1239,  1521,    51,   168,
          78,   483,     0], dtype=int32)
```

RMSE가 1352(만원) 정도로 측정된다.
다른 모델보다는 오차가 낮게 측정되지만 더 낮게 할 순 없을까?

어떻게 하면 오차를 더 낮게 만들 수 있을까?

1. 변수 선택이 잘못 되었을 가능성 확인
2. 하이퍼 파라미터 조정

위의 변수중요도를 참고하여, 변수 선택 실시해보자

모델 설명

변수 중요도 하위 변수들을 제거하면서 다시 모델을 진행

시공사, 병원 수 변수 삭제

```
lgb = LGBMRegressor(linear_tree = True,
                    boosting_type = 'gbdt',
                    objective = 'regression',
                    n_estimators = 9000,
                    learning_rate = 0.001,
                    max_depth = 4,
                    n_jobs = -1)

lgb.fit(X_train, y_train)

MSE = mean_squared_error(y_test, lgb.predict(X_test))
print(f'MSE : {MSE}')
print(f'RMSE : {sqrt(MSE)}')
```

```
CPU times: user 3 µs, sys: 0 ns, total: 3 µs
Wall time: 5.96 µs
MSE : 1830351.8187582497
RMSE : 1352.9049555524032
```

시공사, 병원 수, 구, 지역명, 상위빈도 이름 변수 삭제

```
lgb = LGBMRegressor(linear_tree = True,
                    boosting_type = 'gbdt',
                    objective = 'regression',
                    n_estimators = 9000,
                    learning_rate = 0.001,
                    max_depth = 4,
                    n_jobs = -1)

lgb.fit(X_train, y_train)

MSE = mean_squared_error(y_test, lgb.predict(X_test))
print(f'MSE : {MSE}')
print(f'RMSE : {sqrt(MSE)}')
```

```
CPU times: user 2 µs, sys: 0 ns, total: 2 µs
Wall time: 5.48 µs
MSE : 1837465.7173544266
RMSE : 1355.5315257692928
```

RMSE가 1350(만원) 정도로 오차 변화가 거의 없다.
그렇다면 하이퍼 파라미터 튜닝으로 오차를 줄여보자

하이퍼 파라미터

하이퍼 파라미터 튜닝 과정

```
# 하이퍼 파라미터 튜닝
%time
from sklearn.model_selection import GridSearchCV

params = {
    'learning_rate' : [0.1, 0.01, 0.001, 0.0001],
    'max_depth' : [1,2,3,4]
}

grid_cv = GridSearchCV(lgb, param_grid = params, cv=4, scoring='neg_mean_squared_error', verbose=1)
grid_cv.fit(X_train, y_train)
```

```
CPU times: user 3 µs, sys: 0 ns, total: 3 µs
Wall time: 5.72 µs
Fitting 4 folds for each of 16 candidates, totalling 64 fits
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done  64 out of  64 | elapsed: 25.8min finished
```

```
GridSearchCV(cv=4,
             estimator=LGBMRegressor(learning_rate=0.001, linear_tree=True,
                                     max_depth=4, n_estimators=9000,
                                     objective='regression'),
             param_grid={'learning_rate': [0.1, 0.01, 0.001, 0.0001],
                         'max_depth': [1, 2, 3, 4]},
             scoring='neg_mean_squared_error', verbose=1)
```

```
최적의 하이퍼 파라미터 : {'learning_rate': 0.1, 'max_depth': 4}
예측 오차 : -140939.36042289424
```

1. GridSearchCV 활용
2. learning_rate, max_depth 전환 대입
3. 교차검증 진행
4. 최적의 하이퍼 파라미터를 도출

실제 적용

최적의 하이퍼 파라미터를 적용하여 모델링 후 실제값과 예측값 비교

```
lgb_r = LGBMRegressor(linear_tree = True,
                      boosting_type = 'gbdt',
                      objective = 'regression',
                      n_estimators = 9000,
                      learning_rate = 0.1,
                      max_depth = 4,
                      n_jobs = -1)

lgb_r.fit(X_train, y_train)
```

```
CPU times: user 2 µs, sys: 0 ns, total: 2 µs
Wall time: 5.25 µs
```

```
MSE : 537552.7584483256
RMSE : 733.1798950109895
```

RMSE가 733(만원) 정도로 오차가 파라미터 튜닝 전보다 절반정도로 줄었다.

RMSE 결과값 변화 추이(단위 : 1만원)

초기 Dataset	1차 하위변수 제거	2차 하위변수 제거	파라미터 튜닝 후
1352	1352	1355	733

튜닝한 하이퍼 파라미터를 통해서 test 데이터 확인

```
pred[0]
```

```
62496.679686278745
```

```
y_test.iloc[0]
```

```
62500
```

예측값은 62496.68정도이고,
실제값이 62500 정도로
유의미한 예측을 하는 것으로
판단할 수 있다.

선행연구 비교

선행연구	내용	한계	비교우위
모델의 불확실성을 반영한 아파트가격지수 예측 모형 연구	베이지안 방법론과 시계열 예측을 활용하여 아파트 가격지수를 예측하였다.	헤도닉적 요인이 아닌, 거시경제학적 요인을 위주를 분석을 진행.	1. 헤도닉 주택가격 모형을 기반으로한 입지적, 물리적 특성 반영 1. 주택 가격의 경우 시계열 분석이 주를 이루고 있으나, 머신러닝을 활용한 예측모델의 가능성을 시사함.
아파트 가격에 영향을 미치는 요인의 시공간적 영향력 변화 연구 서울시 25개 구를 대상으로	시계열 분석과 헤도닉 요인들을 동시에 분석하여 서울시 25개 구에 대한 아파트 가격 형성 요인 분석을 실행하였다.	신축되는 아파트의 가격 결정요인을 구체적으로 다루지 못함	
빅데이터를 활용한 글로벌 부동산 가격 분석	의사결정트리를 이용하여 나라별 부동산 가격 추세를 분석하였다.	거시적인 관점에서 부동산 시장에 접근함.	
단독주택가격 추정을 위한 기계학습 모형의 응용	비모수 모형을 활용하여 기존 설명 중심의 모형에서 예측 중심의 모형을 만들고자 하였다.	모수모형과 비모수모형의 비교는 이루어지지 않음.	
머신러닝을 이용한 서울특별시 부동산 지수 예측 모델 비교	시계열 분석과 머신러닝 기법을 모두 활용하여 부동산 지수를 예측, 그 성능을 비교분석하였다.	거시적인 관점과 변수들을 활용하여 부동산 지수를 예측하였음.	
머신러닝 기법을 통한 대한민국 부동산 가격 변동 예측	선형회귀모델을 사용하여 지역별 부동산 가격 변동을 예측하였다.	선형회귀모델의 독립변수로 연도를 설정하여, 구체적인 변수의 상관관계와 영향력을 밝히지 못함.	
기계학습을 활용한 주택매도 결정요인 분석 및 예측모델 구축	다양한 머신러닝 모형을 사용하여 주택배도 결정요인과 예측모델을 구현하였다.	주택매매가격에 대한 부분은 분석하지 않았음.	

활용/발전방향

정부수준

아파트 거래 밀집 지역 및
예상 실거래가 파악

투기성 부동산 거래에 대한
규제 및 대책 실행

민간수준

아파트 주변 입지 영향과
실거래가 정보를 반영하여
합리적인 부동산 거래

발전방향

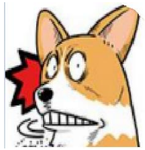
1. 학교 수를 변수로 활용하려 했지만 프로젝트 기간과 데이터문제로 반영하지 못했으나
   이와 같은 지역, 물리적 특성을 추가적으로 반영한다면 모델의 성능 ↑

2.아파트 가격의 시계열적인 특성을 반영할 수 있는 요인을 추가한다면 완성도 ↑

개발후기/소감

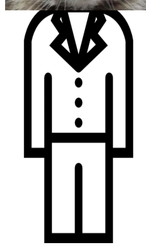
신다롬

데이터 수집, 전처리부터 모델링을 선정하기까지 배운걸 실전에 적용하는게 어려웠지만 그만큼 기억에 많이 남았던 경험이었다



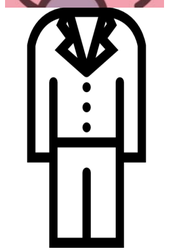
채길호

버전관리의 중요성을 깨닫게 되었다.



이민성

기술적으로 크게 성장할 수 있었던 시간이었다. 특히나 EDA나 전처리 과정에서 다른 사람들의 문제해결 방식을 보며 다양한 관점과 기술을 접할 수 있었다.



이우재

선행연구를 탐색하면서 접근법을 생각해야 할 필요성을 느꼈다.

김보아

요즘 이슈가 되는 아파트 거래에 대하여 영향 변수들을 파악하여 예측해보았다는 것이 흥미로웠다. 팀원분들 모두 감사합니다 !


THANK YOU!