# 서울특별시 주거용 아파트 거래 데이터를 대상으로 트리 기반 앙상블 모형을 활용한 자동 평가 모델 개발 및 평가

01

# 01. 데이터 소개

- 2010년 1월부터 2018년 12월까지의 서울시 아파트 실거래가 자료.
- 국토교통부 실거래가 공개시스템에서 얻은 총 710,731개로 구성된 자료.
- 거래가격, 층 , 면적, 단지명, 주소등에 대한정보를 담고 있으며 위치특성을 변수로 넣어주기 위해 카카오 맵 API를 이용해 위도와 경도를 수집한 후 변수로 활용.
- 공동주택 실거래가 가격 지수, 경기종합 선행지수, 생산자 물가 지수, 소비자 물가 지수 4가지 지표를 월별로 수집한 후 변수로 활용. 다만, 예측을 위해서 한 달을 앞당겨서 코딩을 진행.
- PPSM(평방 미터당 거래금액)을 모형의 타겟으로 설정.

# 01

| Variable | Type | Description |
|---|---|---|
| USM | Numeric | 거래된 아파트의 전용면적 |
| Storey | Numeric | 거래된 아파트의 층 |
| Build year | Numeric | 거래된 아파트의 건축연도 |
| Latitude | Numeric | 거래된 아파트의 위도 |
| Longitude | Numeric | 거래된 아파트의 경도 |
| District | Categorical | 거래된 아파트의 구 |
| Sold date | Numeric | 거래된 날짜 |
| Sold Month | Categorical | 거래된 월 |
| Sold price | Numeric | 거래 금액 |
| PPSM | Numeric | 평방 미터당 거래 금액 |
| Apart index | Numeric | 공동 주택 실거래 가격 지수 |
| Leading index | Numeric | 경기종합 선행 지수 |
| Producer index | Numeric | 생산자 물가 지수 |
| Consumer index | Numeric | 소비자 물가 지수 |

Target

01

## 위치특성

| | 강남구 | 강동구 | 강북구 | 강서구 | 관악구 | 광진구 | 구로구 | 금천구 |
|---|---|---|---|---|---|---|---|---|
| 25% | 890.5 | 503.6 | 387.2 | 438.4 | 435.2 | 600.5 | 389.5 | 351.5 |
| 50% | 1140 | 611.3 | 441.6 | 516 | 500.1 | 700.1 | 469.6 | 399 |
| 75% | 1451 | 781.9 | 526.1 | 635.4 | 583.7 | 809.9 | 562.6 | 468.3 |
| | 노원구 | 도봉구 | 동대문구 | 동작구 | 마포구 | 서대문구 | 서초구 | 성동구 |
| 25% | 402.96 | 351.08 | 427.86 | 534.31 | 571.16 | 425.58 | 764.33 | 553.78 |
| 50% | 455.36 | 397.01 | 498.49 | 631.66 | 690.68 | 512.72 | 942.17 | 654.28 |
| 75% | 525.82 | 465.41 | 600.77 | 765.06 | 828.34 | 636.62 | 1196.44 | 823.48 |

- 위치 변수들이 자동 평가 모델의 변수로 사용되는 데 있어서 큰 타당성 부여.

- 위치변수들을 효과적으로 모델링 할 수 있는 모형이 필요함.

02

# 02. 모델 소개

## Random Forest(RF) 모델

- 훈련 데이터로부터 부트스트랩 데이터를 생성하여 부트스트랩 데이터 마다 의사 결정 트리를 생성한 후 그 예측 결과를 앙상블 하는 모형.

- 노드의 분할에서 랜덤으로 선택된 샘플을 이용하여 트리를 생성함. 이는 트리 간의 상관관계를 감소시키는 경향이 있으므로 out-sample 예측을 위한 보다 강력한 모델이 생성됨.

# 02

1) 훈련데이터 D={($x_i$,$y_i$), i=1, ... ,n}을 정의한다. 여기서 $x_i$는 입력변수 벡터이고, $y_i$는 target 변수이다. 입력변수를 p개라 하면 $x_i$=($x_i$1, ... ,$x_i$p)이다.

2) L로부터 B개의 부트스트랩 데이터 $D_1$, ... , $D_B$를 만든다.

b = 1, ..., B에 대하여 단계 3)의 과정을 반복 수행한다.

3) $D_b$를 이용하여 의사 결정 트리 $T_b(x)$를 생산한다. 단, $T_b(x)$를 생성할 때 입력 변수를 랜덤 추출하여 p개가 아닌 R개(p≥R)의 변수로 구성된 입력변수집합을 사용하여 의사 결정 트리를 생성한다.

4)R개의 분류기를 결합시켜 최종예측모형을 만든다. 여기서 x는 예측하고자 하는 관측치의 입력변수 벡터값이다. target 변수가 연속형인 경우 다음과 같이 평균을 구하는 것과 같다.

$$\hat{f}(x)=\sum_{b=1}^{B} T_b(x)/\mathrm{B}$$

02

## Gradient Boosting Machine(GBM) 모델

- Gradient descent를 이용해서 순차적으로 트리를 만들어가며 이전 트리의 오차를 보완하는 방식으로 boosting을 하는 방법.

- Loss function을 최적화하는 알고리즘으로 gradient의 음수 방향을 갖는 함수를 반복적으로 선택함으로써 gradient decent를 수행한다.

- Loss function이 MSE일시 트리는 이전 트리의 잔차를 target값으로 학습하게 된다.

02

1)예측모형 $\hat{f}(x)$=0과 훈련데이터 D={$(x_i,r_i)$, i=1, ... ,n}을 정의한다. 여기서 $r_i$=$y_i$이고, $x_i$는 입력변수 벡터, $y_i$는 target 변수이다

2)전체 트리 개수가 B일때 b=1,2, ..., B에 대하여 단계 3),4),5)의 과정을 반복 수행한다.

3)훈련데이터 D를 이용하여 의사 결정 트리 $\hat{f}_b$를 생산한다.

4)기존의 예측모형, 학습률 $\lambda$, 새로운 의사 결정 트리 $\hat{f}_b$를 이용하여 기존의 예측모형을 업데이트한다.

$$\hat{f}(x) = \hat{f}(x)+\lambda\hat{f}_b(\mathrm{x})$$

5)잔차를 업데이트한다.

$$r_i = r_i\text{-}\lambda\hat{f}_b(x_i)$$

6)최종 예측 모형은 다음과 같다.

$$\hat{f}(x) = \sum_{b=1}^{B}\lambda\hat{f}_b(x)$$

02

## XGBoost 모델

- 기존의 GBM모델 + 병렬화, 가지치기 등을 통한 시스템 최적화
  + 규제 항, Weighted Quantile Algorithm등을 통한 성능향상
  ->기존의 GBM에 비해 속도가 빠르고 일반화된 모델 얻을 수 있음.

- XGBoost는 훈련 과정에서 손실 함수와 규제 항으로 이루어진 목적함수를 최소화하기 위해 훈련됨. 규제 항은 모델의 과적합을 제한하기 위해 추가된 항이다.

최종 예측값

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

K개의 의사결정트리

목적함수

$$Obj = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{K} \Omega(f_k)$$

손실함수

K개의 의사결정트리에 대한 규제항

02

## LightGBM 모델

- XGBoost를 이은 GBM 기반의 모델로서 Leaf-wise 방식을 사용하여 복잡한 모델을 만들어 더욱 높은 정확도를 만들어냄.
- Gradient-based One-Side Sampling(GOSS)와 Exclusive Feature Bundling(EFB)를 통해서 메모리 사용량 감소, 빠른 훈련 속도 등의 장점이 존재.
- 과적합이 쉽게 되는 모델이기에 10,000개 이상의 데이터 세트가 있을 때 적합한 모델.

02

## GOSS

- 작은 Gradient를 가진 데이터는 이미 훈련이 잘 되어있음.
  ->다음 트리를 훈련할 때 큰 Gradinet를 가진 데이터는 유지하고
  작은 Gradient를 가진 데이터만 Sampling을 수행.

- Gradient의 절대값에 따라 데이터를 정렬하고 미리 정한 a, b값에 따라
  상위 a*100% 데이터는 유지, 나머지 데이터는 b*100%를 무작위로 Sampling
  수행.

- 분할점을 찾기 위해 정보획득량 계산시 상수(1-a)/b를 이용하여 작은
  Gradient를 가진 Sample된 데이터를 증폭시켜줌.

02

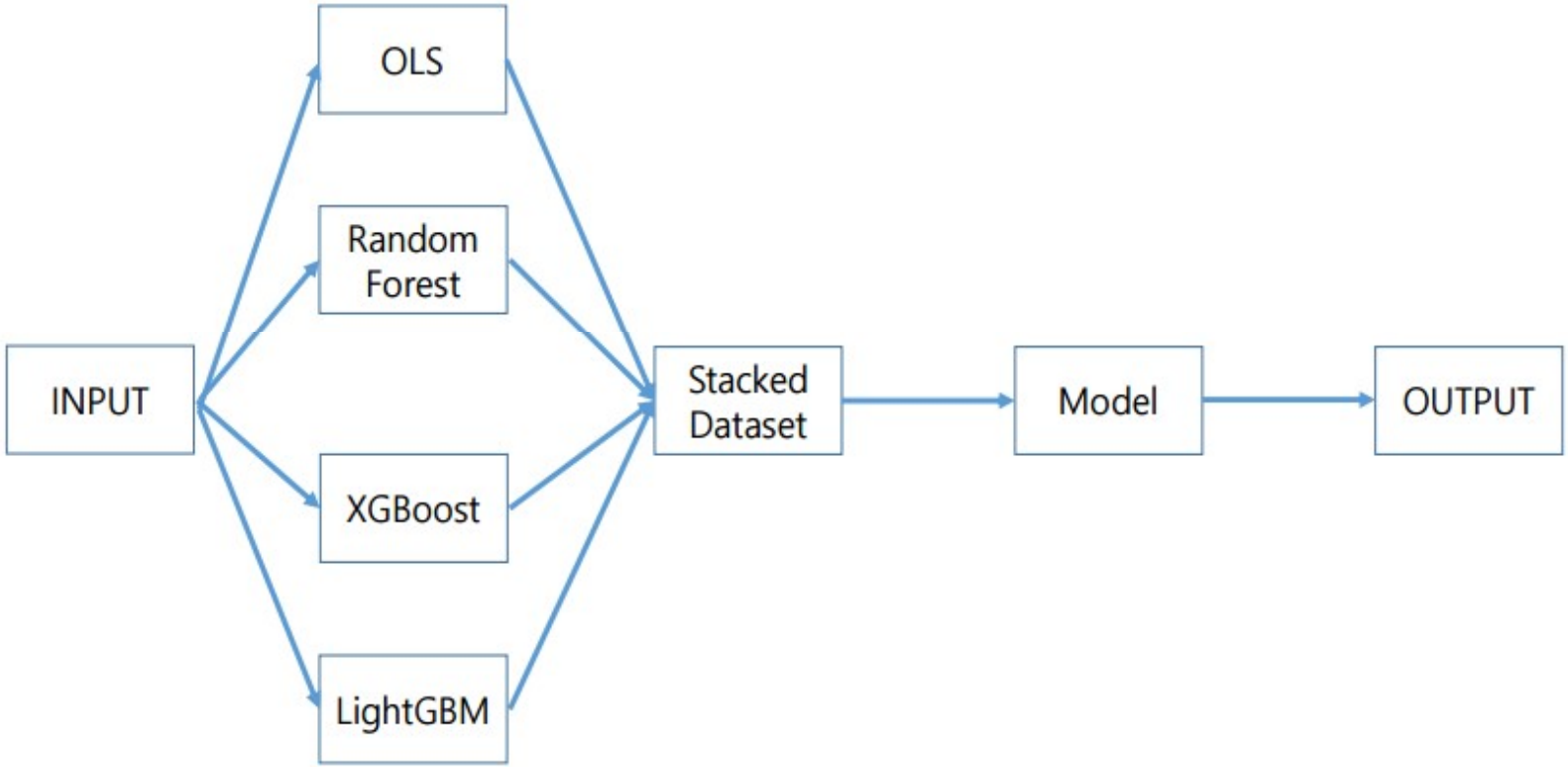
## Stacking 모델

- Submodel들을 사용하여 훈련 데이터의 개별 추정치를 만든 후 Model-Stacker라는 별도의 모델과 추정치들을 이용하여 훈련을 시킨 후 테스트 데이터에 대한 예측값을 생성함.

- 모델들별로 학습하는 방식이 서로 다르므로 서로 결합하였을 때 예측력이 높은 모델을 개발할 가능성이 높아짐.

02

- 자동평가모델에 Stacking 모델을 적용하기 위한 절차



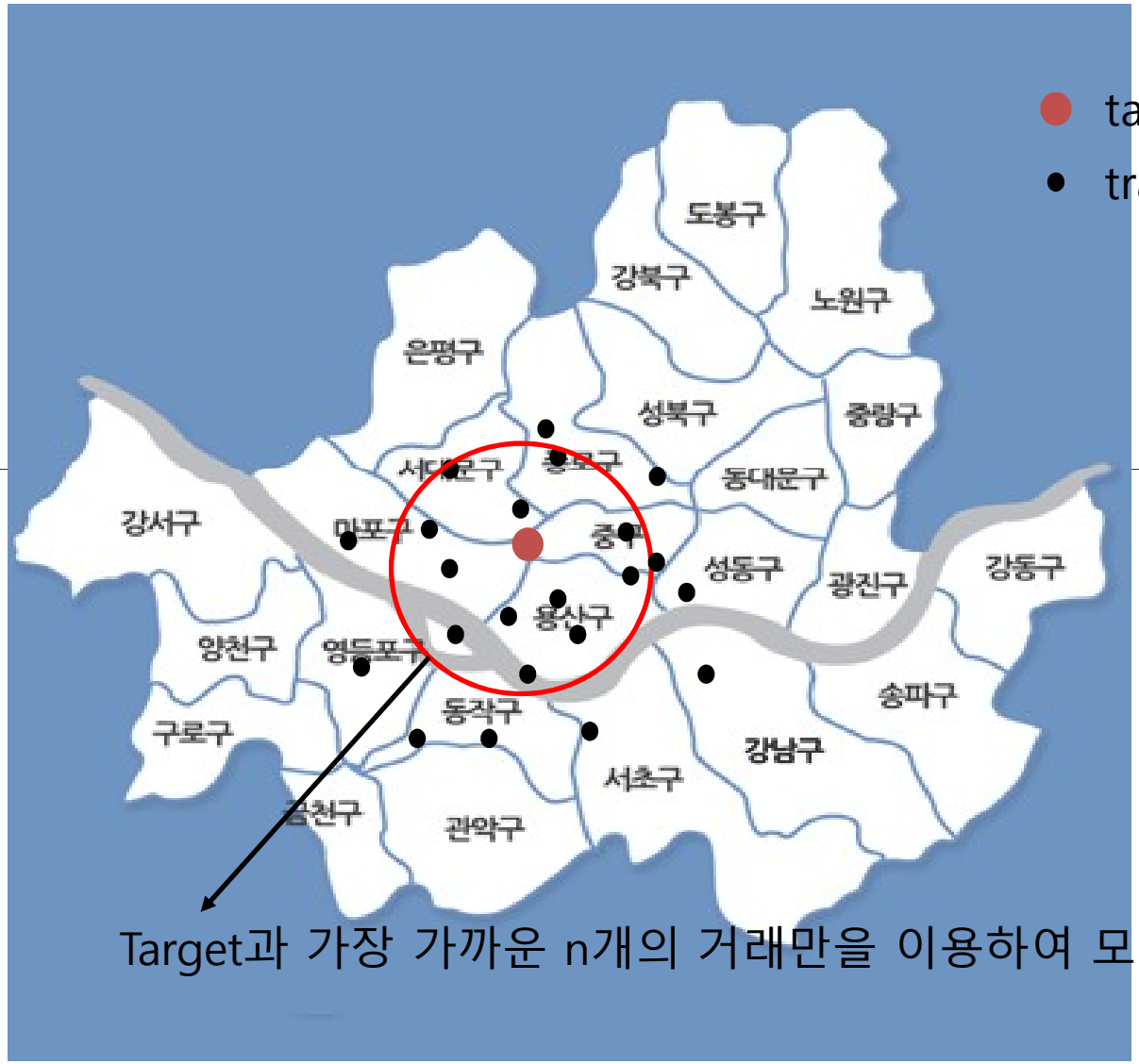
02

## Comparable Market Analysis

- 각각의 타겟을 기준으로 데이터를 맞춤화하고 그 후 하나의 모델을 적합 시키는 것으로 종종 부동산 평가에 적용되는 일반적인 평가 방법이다.

- 자동 평가 모델은 각각의 타겟에 데이터를 맞춤화할 때 지리적으로 근접한 거래를 선택함으로써 이 개념을 적용 시킬 수 있다.

# 02


target
transaction
도봉구
강북구
노원구
은평구
성북구
중랑구
동대문구
강서구
마포구
중구
성동구
광진구
강동구
용산구
양천구
영등포구
송파구
동작구
구로구
강남구
서초구
관악구
Target과 가장 가까운 n개의 거래만을 이용하여 모델링

02

## 예측분석

- 총 710,731개의  데이터를 이상치 제거 및 전처리를 통해서 554,484개의  in-sample과  1,192개의 out-sample로 나눔.
- 이때 in-sample은 2010년 1월부터 2018년 11월까지의 데이터 이며 out-sample은 2018년 12월 데이터이다.
- 모형간의 성능 비교를 위해 Mean Absolute Percentage Error(MAPE), Median Absolute Percentage Error(MdAPE), 결정계수 총 3가지 측도를 사용한다.

02

$$MAPE = \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| * \frac{1}{n} * 100\%.$$

$$p_t = \left| \frac{y_i - \hat{y}_i}{y_i} \right| * 100\%, \qquad MdAPE = median(p1, p2, ..., p_n).$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}.$$

03

# 03. 분석 결과

## 분석결과

- 자동 평가 모델 성능의 비교를 위해 2018년 공동주택공시가격 자료를 이용하여 2018년 12월 서울 특별시 아파트 거래금액을 예측.
- 2010년 1월부터 2018년 11월까지의 데이터를 이용하여 OLS를 기반으로 하는 HPM 모형, Random Forest, XGBoost, LightGBM, Stacking 모형에 적합한 후 2018년 12월 서울 특별시 아파트 거래금액을 예측.
- 4개의 앙상블 모형이 공동주택공시가격 보다 훨씬 뛰어난 예측력과 적합도를 보여주고 있음.

03

| | MAPE | Within 25% | Within 50%(=MdAPE) | Within 75% | 결정계수 |
|---|---|---|---|---|---|
| BaseLine | 41.089 | 36.983 | 42.119 | 46.523 | 0.087 |

| | MAPE | Within 25% | Within 50%(=MdAPE) | Within 75% |
|---|---|---|---|---|
| HPM | 28.202 | 10.254 | 22.053 | 37.237 |
| RandomForest | 9.470 | 2.914 | 6.092 | 10.634 |
| XGBoost | 7.652 | 2.036 | 4.702 | 9.062 |
| LightGBM | 8.039 | 2.481 | 5.141 | 9.722 |
| Stacked Model(XG) | 7.553 | 2.096 | 4.779 | 9.064 |
| Stacked Model(LGBM) | 7.805 | 2.239 | 4.958 | 9.311 |

| | 결정계수 | 훈련시간 |
|---|---|---|
| HPM | 0.427 | 2.58s |
| RandomForest | 0.940 | 23min 6s |
| XGBoost | 0.959 | 1h 39min 30s |
| LightGBM | 0.958 | 7min 10s |
| Stacked Model(XG) | 0.959 | 13min 26s |
| Stacked Model(LGBM) | 0.931 | 1min 25s |

*Note*: Stacked Model의 실행시간은 Model-Stacker의 실행시간 만을 나타낸다. 총 훈련시간은 SubModel의 훈련 시간을 다 더한뒤 Model-Stacker의 훈련시간을 더해야 한다.

03

## 상대적으로 성능이 낮은 HPM모형을 Stacking 모형에 포함하는 것이 성능 향상에 도움이 되는가?

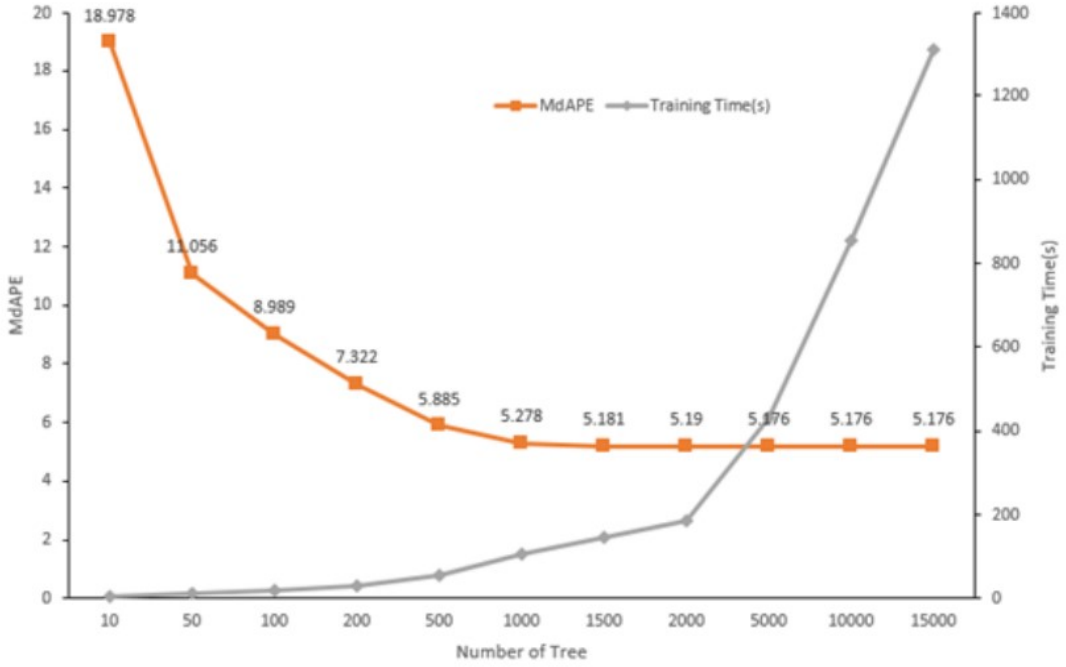
- HPM모형을 제외하고 XGBoost와 LightGBM모형을 Model-Stacker로 하는 Stacking 모형을 적합 시킨 후 예측을 시행함.
- 근소하게나마 성능이 떨어지는 것을 확인. -> 비록 Submodel의 성능이 낮더라도 Stacking 모형은 Submodel들이 다양할 때 효과적임을 확인.

| | MAPE | Within 25% | Within 50%(=MdAPE) | Within 75% |
|---|---|---|---|---|
| Stacked Model(XG) | 7.609 | 2.146 | 4.826 | 9.028 |
| Stacked Model(LGBM) | 7.848 | 2.252 | 4.976 | 9.386 |
| Stacked Model(XG) | 0.958 | 12min 1s | | |
| Stacked Model(LGBM) | 0.929 | 1min 18s | | |

03

## 의사 결정 트리의 개수에 따른 MdAPE와 훈련시간의 변화

- LightGBM 모형을 이용.
- 다소 선형적인 관계를 관찰 할 수 있음. 1500개의 트리만으로 약 5%의 MdAPE를 달성 할 수 있음.



03

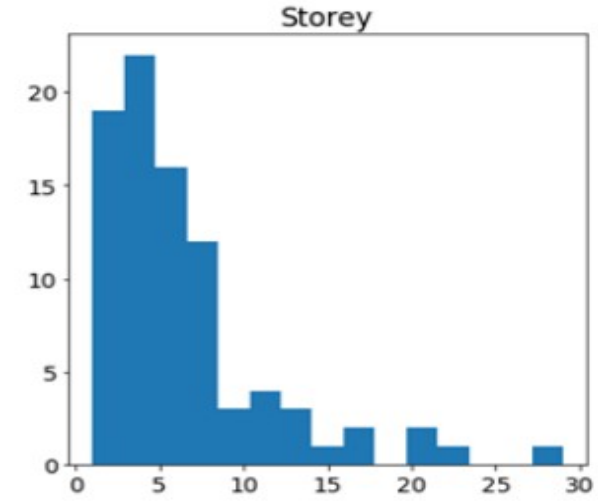
## LightGBM의 예측값과 타겟값의 차이가 크게 나는 데이터들의 특성

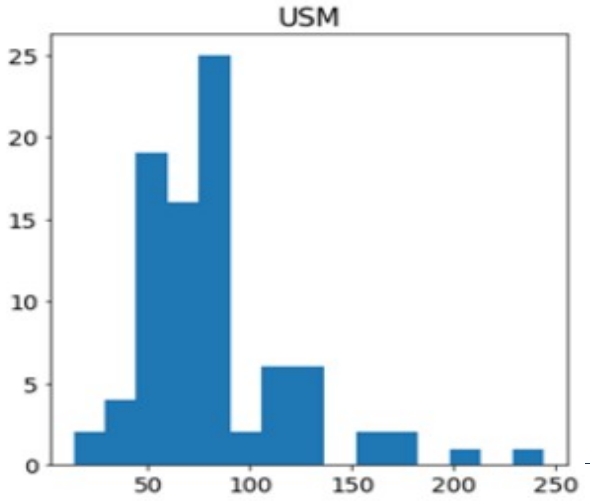
- 타겟값과 20%이상 차이 나는 86개 데이터들의 분포를 변수 별로 살펴봄.
- 1,192개의 테스트데이터에 대응 되는 각 변수의 분포들과 크게 다르지 않음.

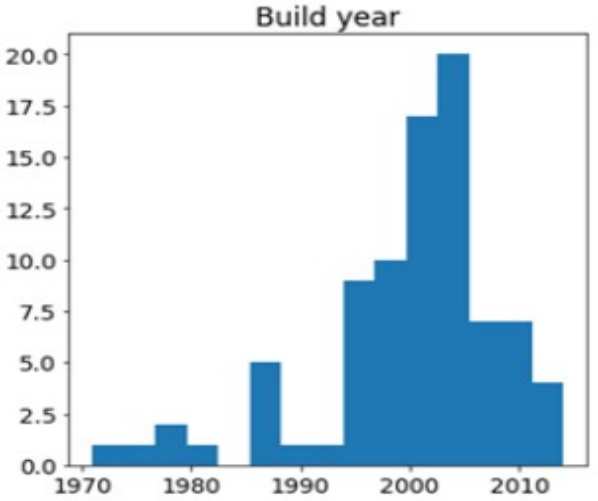
| 종로구 | 강남구 | 동대문구 | 서초구 | 용산구 | 양천구 | 마포구 | 강서구 | 성북구 |
|---|---|---|---|---|---|---|---|---|
| 2 | 7 | 7 | 5 | 3 | 8 | 5 | 7 | 4 |
| 송파구 | 광진구 | 은평구 | 금천구 | 성동구 | 구로구 | 강동구 | 관악구 | 영등포구 |
| 3 | 3 | 5 | 2 | 3 | 4 | 2 | 1 | 3 |

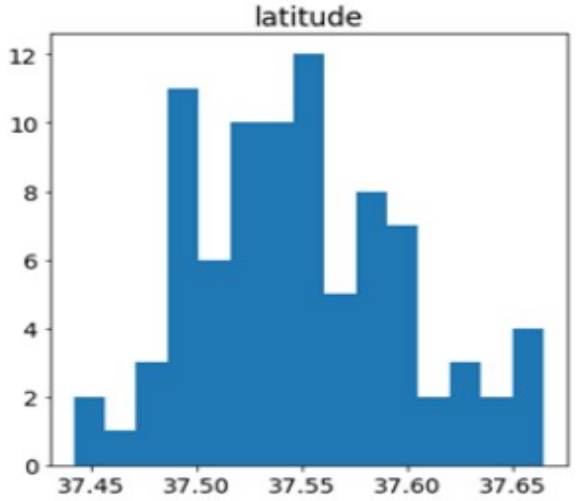
| 노원구 | 강북구 | 도봉구 | 중랑구 | 동작구 |
|---|---|---|---|---|
| 5 | 2 | 1 | 2 | 2 |

# 03


Storey
0
5
10
15
20
0
5
10
15
20
25
30


USM
0
5
10
15
20
25
50
100
150
200
250


Build year
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
1970
1980
1990
2000
2010

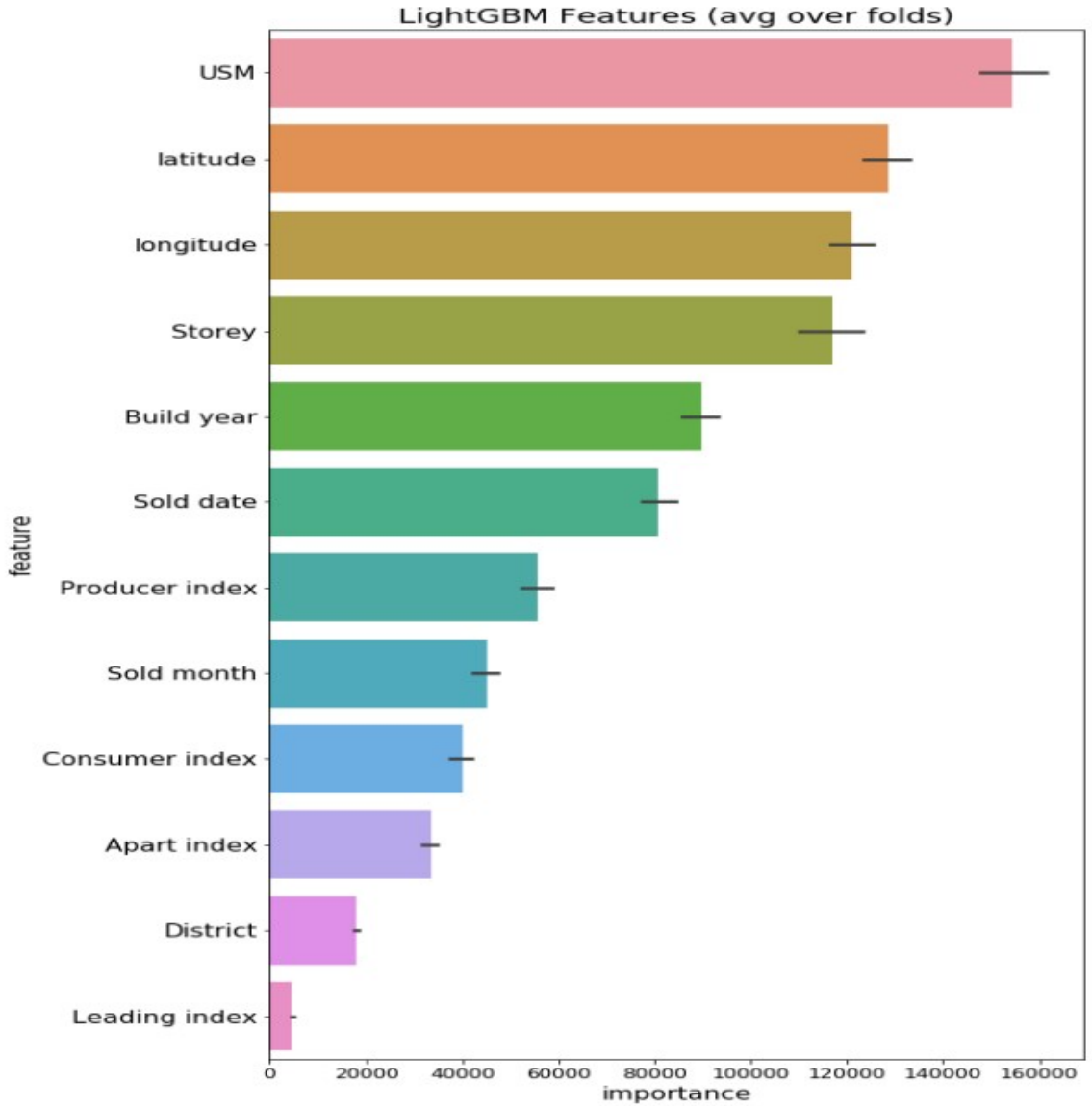

latitude
0
2
4
6
8
10
12
37.45
37.50
37.55
37.60
37.65

03

## 변수 중요도

- LightGBM모형의 경우 나무 모형에 특정 변수를 포함해서 얻을 수 있는 정보이득으로 변수 중요도를 제공함.
- 위도와 경도로 나타내어지는 위치정보와 전용면적, 층, 건축연도같이 건물의 정보가 중요한 변수로 간주됨.

# 03


LightGBM Features (avg over folds)
USM
latitude
longitude
Storey
Build year
Sold date
Producer index
Sold month
Consumer index
Apart index
District
Leading index
feature
0
20000
40000
60000
80000
100000
120000
140000
160000
importance

03

## 모든 변수를 활용한 LightGBM 모델의 적합 및 예측

- Random forest 모델과 XGBoost 모델에서는 범주형 변수를 데이터 세트에 넣을시 One-hot encoding을 필요로함.
- 아파트명, 거리명, 동(행정구역) 3개의 변수는 각각 7106, 3492, 331개의 상당히 많은 개수의 범주형 변수이며 One-hot encoding을 할 시 메모리 자원의 초과 발생.
- 3개의 변수를 데이터세트에 추가하여 LightGBM모델에 적합한 후 예측을 하고 모형을 적합할 때 각변수의 중요도를 살펴봄.

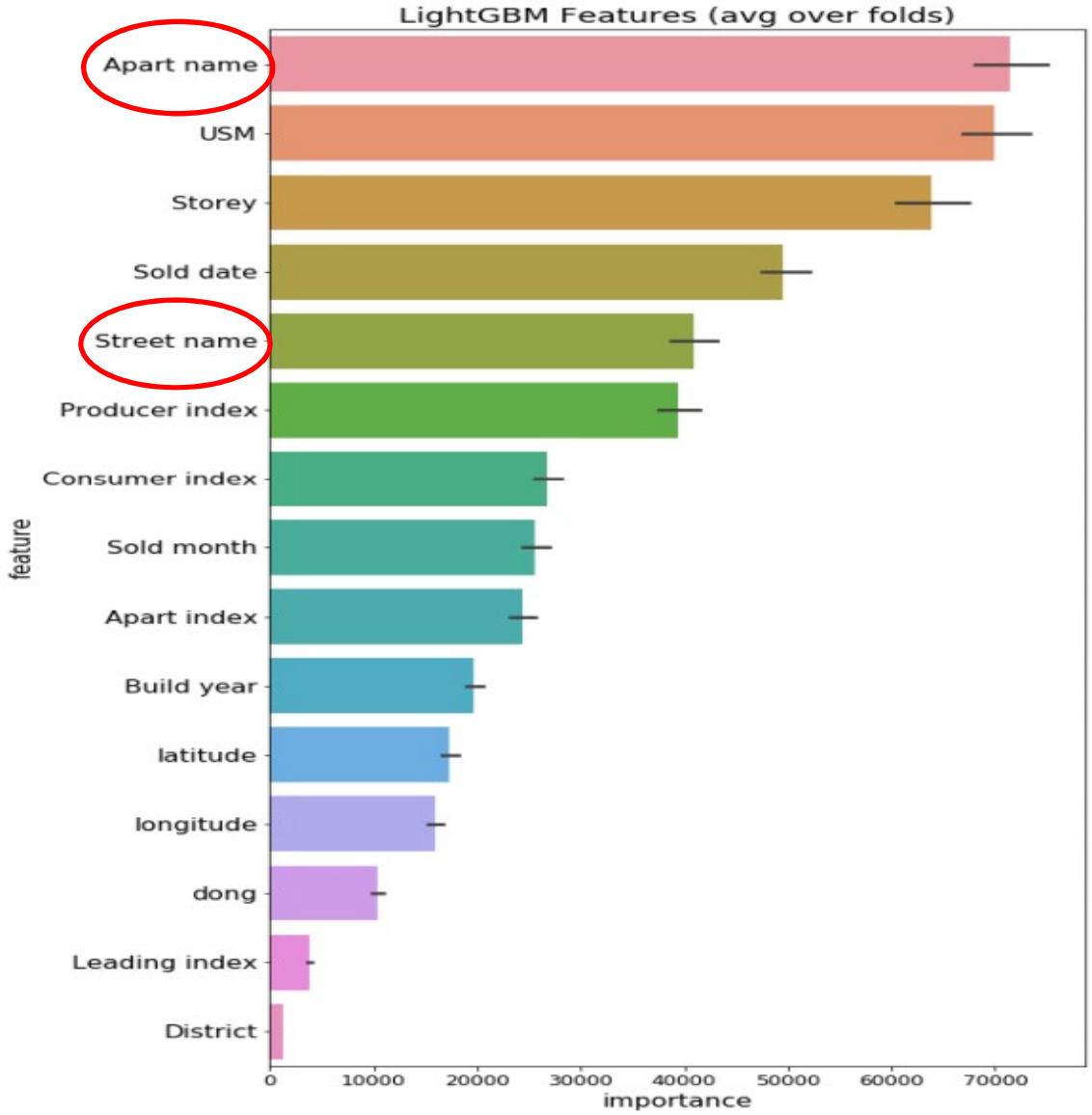
# 03

- MAPE, MdAPE, 결정계수가 각각 6.822, 4.591, 0.967로서 모든 머신러닝 모형들보다도 성능이 뛰어남.
- 변수중요도에서 아파트명, 거리명으로 나타내어지는 위치정보가 중요한 변수로 간주됨. 위도와 경도의 변수 중요도가 상대적으로 떨어진 것이 확인 가능.

| | MAPE | Within 25% | Within 50%(=MdAPE) | Within 75% |
|---|---|---|---|---|
| LightGBM | 6.822 | 2.131 | 4.591 | 8.544 |

| | 결정계수 | 훈련시간 |
|---|---|---|
| LightGBM | 0.967 | 22min 44s |

03


LightGBM Features (avg over folds)
Apart name
USM
Storey
Sold date
Street name
Producer index
Consumer index
Sold month
Apart index
Build year
latitude
longitude
dong
Leading index
District
feature
0
10000
20000
30000
40000
50000
60000
70000
importance

03

## Comparable Market Analysis

- 모든 변수를 활용한 LightGBM 모델과 Comparable Market Analysis 개념을 적용하여 모형을 적합 시킨 후 예측을 시행.

- 하나의 타겟에 대해 지리적으로 근접한 거래를 5,000개, 20,000개로 설정하여 2개의 모형을 적합 시킨 후 예측을 시행.

- 기존의 적합된 LightGBM 모델이 일반화가 잘 되었음을 알 수 있음.

| | MAPE | Within 25% | Within 50%(=MdAPE) | Within 75% |
|---|---|---|---|---|
| LightGBM(base line) | 6.822 | 2.131 | 4.591 | 8.544 |
| LightGBM with 5000 | 7.500 | 2.407 | 4.989 | 9.063 |
| LightGBM with 20000 | 6.958 | 2.142 | 4.754 | 8.697 |

| | 결정계수 | 훈련시간 |
|---|---|---|
| LightGBM(base line) | 0.967 | 22min 44s |
| LightGBM with 5000 | 0.957 | 11h 28min 54s |
| LightGBM with 20000 | 0.964 | 1d 8h 24min 21s |

04

# 04. 결론

- 사분위수와 MAPE를 비교한 결과 트리 기반의 머신러닝 모형들의 우수성 확인.
- 위도와 경도로 나타내어지는 위치정보가 중요 변수로 간주됨.
- 기존의 HPM모형에서 모형화 할 수 없었던 위치와 관련된 변수의 중요성을 확인.
- 이러한 변수들을 효과적으로 모형화하면서 다중공선성 문제를 피해갈 수 있는 트리 기반의 머신러닝 모형들이 상대적으로 자료를 더 잘설명할 수 있음.
- 특히 위치와 관련된 모든 변수들을 활용 시 모델의 성능이 개선되는 것을 확인.