

# ESC Final project

**아파트 실거래가 구하기**  
**With Bayesian Regression**

박재현, 임세희, 김송희, 전상후, 박태주, 정연섭

2022. 11. 24

# 목차

a table of contents

---

- 1 실거래가 & 기타 용어정리
- 2 베이지안 회귀분석이란?
- 3 데이터 소개
- 4 분석 결과
- 5 기타 방법론 소개

---

Part 1

# 실거래가 & 기타 용어정리

# 1. 실거래가 & 기타 용어정리

## 실거래가란?

부동산이 **실제로 시장에서** 거래된 가격

-> “절대 바꿀 수 없는 가격 정보“ (by 부동산실거래가 신고의무제도)

-> 부동산 거래 시 취득세나 양도세의 기준이 되는 가격

-> **계약이 이루어져야** 나타나는 정보

단점: 거래가 자주 일어나지 않는 지역이 아니라면,  
부동산 조사나 가격 동향 조사할 때 실거래가를 사용하기는 어렵다.

-> 현재의 가격을 유추하기 위해 **다른 가격 용어**들을 사용하게 된다.

### 아파트 실거래가지수 변동률

단위: %    ■ 8월(전월 대비)    ■ 1~8월 누적



김민지 기자 20221018

# 1. 실거래가 & 기타 용어정리

## 호가란?

집 주인이 집을 팔고자 부르는 가격

-> 집 주인이 기대하는 기대가격

-> if (부동산 과열): 호가는 계속 올라가고 실거래가는 밑에서 빠르게 증가

장점: 구매자 눈 앞에 직접 제시된 가격으로, 바로 살 수 있다는 의미.  
즉, 주택을 거래하기 위해 필요한 최고액

단점:

- 1) 호가는 판매자의 가격이기 때문에 실제 시세 범위에 비해 호가는 높게 형성되었다.
- 2) 판매자들의 담합에 의해 호가 범위를 높게 붙잡는 경우도 있다.

<포항 주요아파트 매물 및 실거래 현황(1월 현재)>

아파트명	실매물 정보	전용면적(m <sup>2</sup> )	12월 시세	12월 실거래	1월 실거래	매물 호가
포항자이	317건 (아파트 전체)	84.95m <sup>2</sup> (26평)	59,500	11건	1건	65,000
효자웰빙타운SK뷰(1차)	209건	84.94m <sup>2</sup> (26평)	39,000	13건	1건	50,000
효자그린(2차)	233건	84.97m <sup>2</sup> (26평)	36,000	4건	0건	45,000
장성푸르지오	59건	84.98m <sup>2</sup> (26평)	48,000	1건	0건	60,000
두호SK뷰푸르지오(1단지)	140건	84.99m <sup>2</sup> (26평)	45,000	6건	1건	60,000
창포메트로시티(1단지)	90건	84.69m <sup>2</sup> (26평)	41,000	8건	0건	47,000
창포주공(1차)	95건	49.77m <sup>2</sup> (15평)	6,750	15건	0건	12,000
두호주공아파트(2차)	50건	46.68m <sup>2</sup> (14평)	15,250	31건	1건	17,500

자료:KB부동산/국토교통부 실거래가 공개시스템/디디하우스

# 1. 실거래가 & 기타 용어정리

## 시세란?

현재 상황을 고려했을 때 **거래 될 가격**

만약, 실거래 내역 자체가 오래되어 현재 가격을 제대로 반영하지 못한다면??

→ 부동산 업계에서 해당 아파트 단지의 **예상 가격을 선정**

→ 즉, 부동산마다 기준이 다르기 때문에 아파트의 “하한가”와 “상한가”가 설정될 것이다.

\* KB 부동산 시세가 가장 공신력 있는 것으로 여겨진다.



■ 시세 ■ 실거래가 ■ 현재 매물가격 2021.11.15. 한국부동산원 기준 / 2021.11. 국토교통부 기준



**22억 7,000**  
하한가

**24억**  
상한가

**37~38%**  
매매가 대비 전세가

# 1. 실거래가 & 기타 용어정리

## 공시지가란?

세금 납부에 대한 기준을 매기기 위해, 정부와 감정평가사가 함께 매년 매기는 부동산 가치의 가격 (= 개별 공시지가)

-> 통상적으로 주택 실거래가의 50% ~ 70% 를 반영

\* 2021년에는 전국 평균 19% 상승, 세종시의 경우 약 70% 상승

The screenshot shows the KREIS website interface. At the top, there are logos for the Ministry of Land, Urban and Planning (국토교통부) and the Korea Real Estate Council (KREIS). The main navigation bar includes icons for public housing, standard housing, individual housing, standard land, and individual land. The 'Individual Land' (개별공시지가) icon is highlighted with a red box. Below the navigation bar, there is a large green banner titled '개별토지에 대한 “개별공시지가”' (Individual Public Market Value for Individual Land). The banner text explains that the market value is determined by the Ministry of Land, Urban and Planning based on the standard public market value, and is used for land-related taxes and other purposes. It also mentions that the value is updated annually on January 1st and July 1st. A magnifying glass icon is shown over a map of Korea, and a button labeled '자세히보기' (View Details) is at the bottom.

# 1. 실거래가 & 기타 용어정리

## 요약

일반적으로, 부동산 호가  $\geq$  실거래가, 시세가  $\geq$  공시지가

가장 중요한 정보: **최근의 실거래가** (= 실제로 계약이 되었던 가격) 으로, 실제로 이 가격에서 큰 변동이 없는 편이다.

용어	의미	주의점
분양가	건설사가 정한 첫 주택 가격	만드는 건설사에 따라 가격이 다르게 정해짐
실거래가	실제 계약이 이루어진 가격	오래되거나 없으면 현재의 가치를 알 수 없음
부동산 호가	판매자가 원하는 가격(매물가)	실제 실거래가보다 가격이 높게 형성
시세가	업계에서 예측하는 실제 판매가격	업체마다 다른 기준, 매물수에 따라 시세가 정확하지 않을 수 있음
공시지가	부동산 세금 기준에 쓰이는 부동산자산가격	1년에 한번으로 시세의 급격한 변화를 반영하지 못함
감정평가액	공시지가에 '그밖의 요인'을 포함한 부동산 가치 평가금액	평가 방법에 따른 금액이 달라짐



---

Part 2

# 베이지안 회귀분석이란?

# Bayesian Model

We have observed data  $(X_i, Y_i)$  for  $i = 1, \dots, n$  and we will regress  $Y_i$  onto  $X_i$ .

$$Y_i = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5 + \beta_6 X_6 + \beta_7 X_7 + \beta_8 X_8 + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2)$$

$X_1$ : city (1 if Seoul, 0 if Busan)

$X_2$ : dong

$X_3$ : apt

$X_4$ : year of completion

$X_5$ : transaction year month

$X_6$ : floor

$X_7$ : top10

$X_8$ : log\_area

log\_price

Binary 범주형 변수 (City, top10)

**Beta distribution**

연속형 변수 (Log \_area, dong, apt, floor 등등)

**Normal distribution**

## 범주형 변수 initial value 설정

train seoul(1) : 790438  
train busan(0) : 426115

test seoul(1) : 4016  
test busan(0) : 1447

train top10(1) : 150150  
train not top10(0) : 1066403

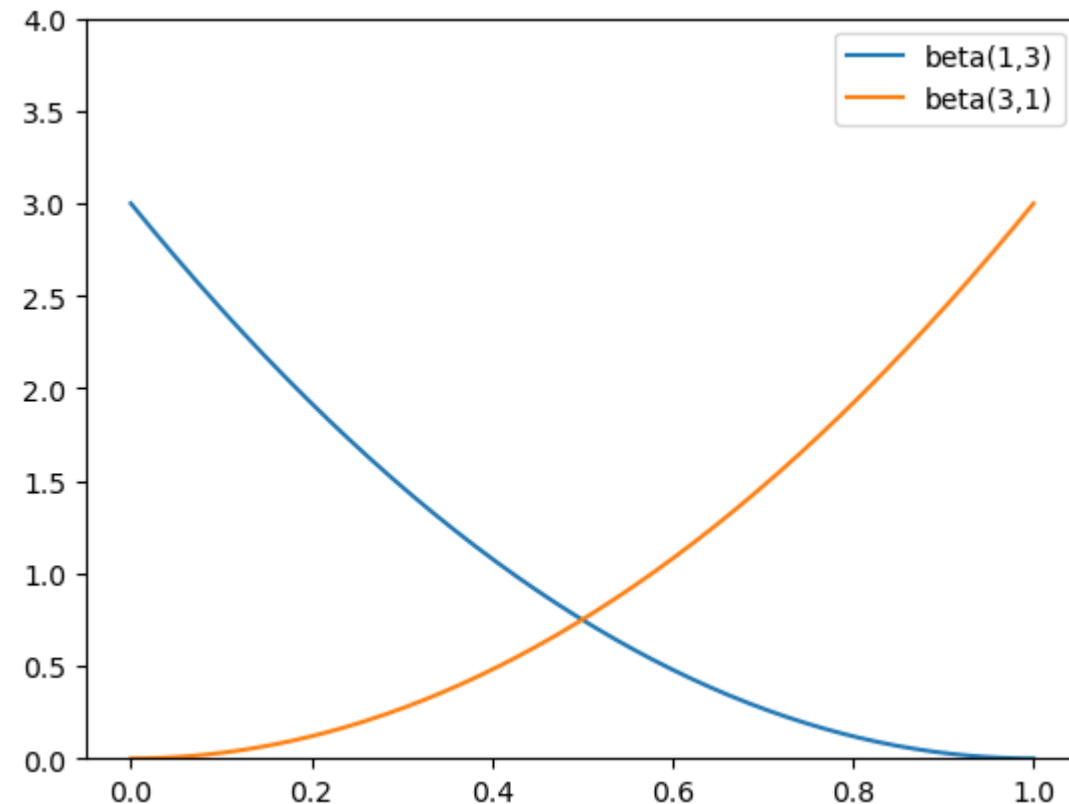
test top10(1) : 654  
test not top10(0) : 4809

### City :

1인 data 수 > 0인 data 수  
→  $\alpha=3, \beta=1$  사용

### Top 10 :

1인 data 수 < 0인 data 수  
→  $\alpha=1, \beta=3$  사용



# Posterior Distribution

The posterior can be written up to some constant.

$$\textit{posterior} \propto \textit{likelihood} \times \textit{prior}$$

$$p(\beta_0, \beta_1, \beta_2, \dots, \beta_8, \sigma^2 | \mathbf{X}, \mathbf{Y})$$

$$\propto \prod_{i=1}^n p(\mathbf{X}_i, \mathbf{Y}_i | \beta_0, \beta_1, \beta_2, \dots, \beta_8, \sigma^2) \times p(\beta_0)p(\beta_1) \cdots p(\beta_8)p(\sigma^2)$$

*likelihood*

# Bayesian Regression 원리

Posterior distribution으로부터 각 parameter마다 Sampling을 진행한다.

방법) Gibbs Sampler, Metropolis-Hastings Algorithm



각 parameter마다 만들어진 sample에서 **posterior mean**을 구한다.

**Estimator**

---

# Part 3

## 데이터 소개





# Steps

---

1. Build a formula relating the features to the target and decide on a prior distribution for the data likelihood
2. Sample from the parameter posterior distribution using MCMC

# pm.Model()

```
In [30]: 1 with pm.Model() as independent_regression_model_full:
2         # data
3         x1 = pm.Data('x1',train_['city'].values)
4         x2 = pm.Data('x2',train_['dong'].values)
5         x3 = pm.Data('x3',train_['apt'].values)
6         x4 = pm.Data('x4',train_['year_of_completion'].values)
7         x5 = pm.Data('x5',train_['transaction_year_month'].values)
8         x6 = pm.Data('x6',train_['floor'].values)
9         x7 = pm.Data('x7',train_['top10'].values)
10        x8 = pm.Data('x8',train_['log_area'].values)
11        y1 = pm.Data('y1',train_['log_price'].values)
12
13        # prior
14        alpha = pm.Normal('intercept_1',0,10)
15        # beta_1 = pm.Normal('coeff_1',0,10)
16        beta_1 = pm.Beta('coeff_1',3,1) #binary
17        beta_2 = pm.Normal('coeff_2',0,10)
18        beta_3 = pm.Normal('coeff_3',0,10)
19        beta_4 = pm.Normal('coeff_4',0,10)
20        beta_5 = pm.Normal('coeff_5',0,10)
21        beta_6 = pm.Normal('coeff_6',0,10)
22        # beta_7 = pm.Normal('coeff_7',0,10)
23        beta_7 = pm.Beta('coeff_7',1,3) #binary
24        beta_8 = pm.Normal('coeff_8',0,10)
25        sigma = pm.Exponential('error',lam=1)
26
27        mu = alpha + beta_1*x1 + beta_2*x2 + beta_3*x3 + beta_4*x4 + beta_5*x5 + beta_6*x6 + beta_7*x7 + beta_8*x8
28        y_hat = pm.Normal('y_hat',mu,sigma,observed=y1)
29
30        trace_independent_regression_full = pm.sample(draws=2000,tune=1000,init="adapt_diag")
```

data

prior

formula

sampling

# Regression Model

## ♥ Data

pm.Data()로 모델에 불러와서 각각 x1~x8로 변수 지정

## ♥ Prior

regression coefficients: **alpha(intercept)**, **beta\_1~beta\_8**, **sigma**에 대한 prior를 각각 지정

- numerical type은 일괄  $\text{normal}(0, 10)$
- binary type인
  - ‘city’에 해당하는  $\text{beta}_1 \sim \text{beta}(3, 1)$ ,
  - ‘top10’에 해당하는  $\text{beta}_7 \sim \text{beta}(1, 3)$
- $\text{sigma} \sim \text{Exponential}$

# Regression Model

## ♥ Formula

```
27 mu = alpha + beta_1*x1 + beta_2*x2 + beta_3*x3 + beta_4*x4 + beta_5*x5 + beta_6*x6 + beta_7*x7 + beta_8*x8
28 y_hat = pm.Normal('y_hat',mu,sigma,observed=y1)
```

## ♥ MCMC Sampling

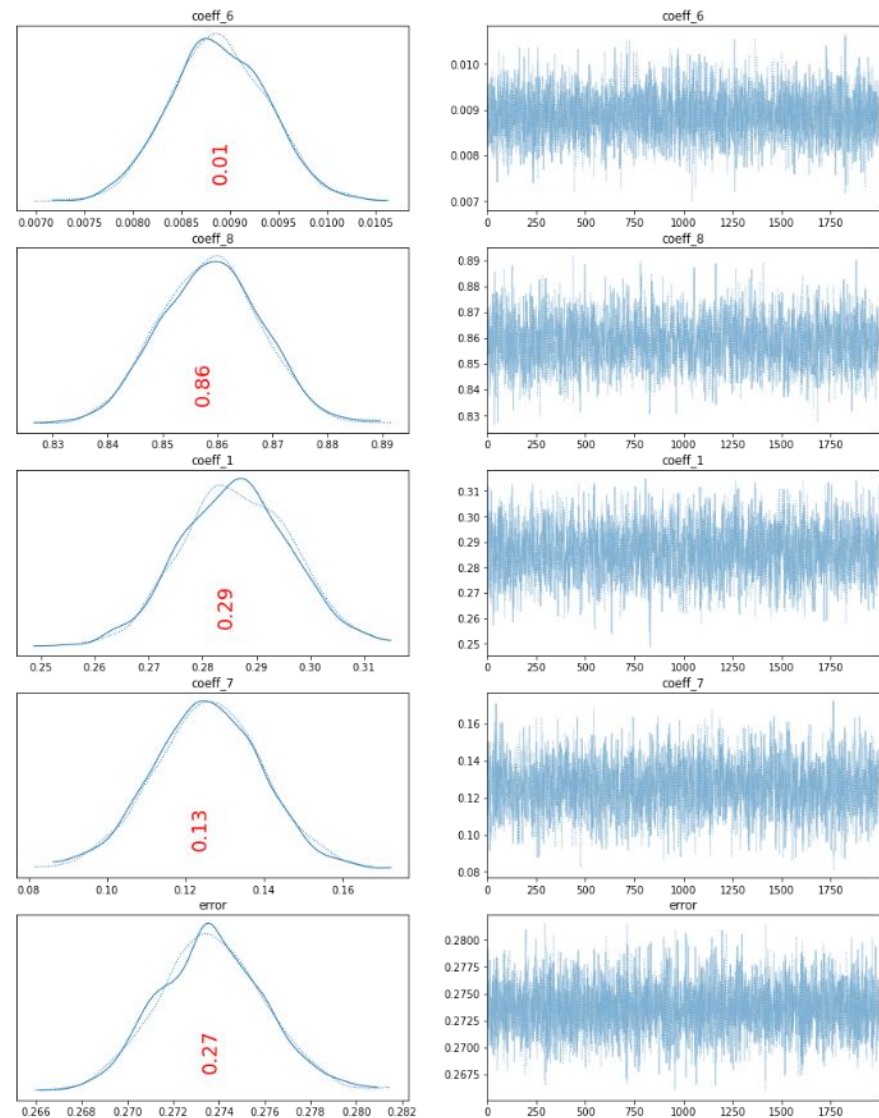
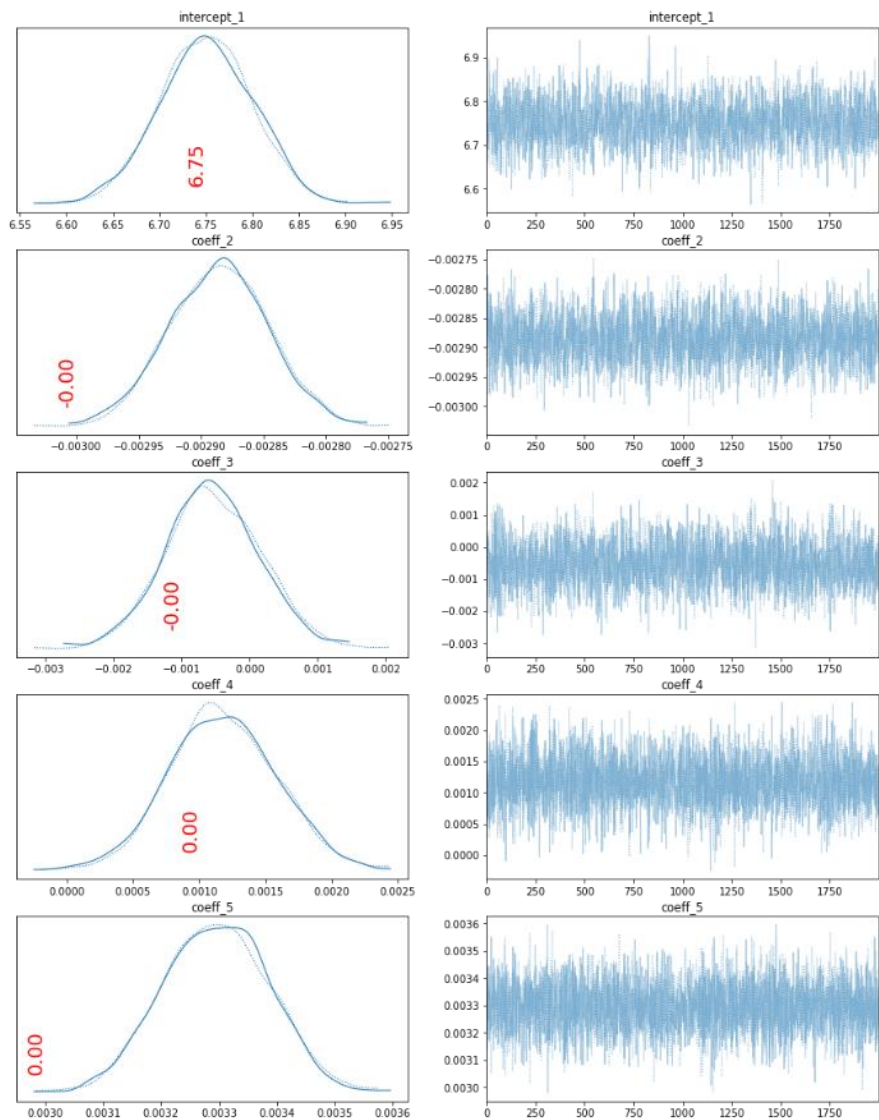
총 3000개를 sampling해서 1000개를 burn-in으로 취급, 최종 2000개 samples.

---

# Part 4

## 분석 결과

# MCMC 결과: plot



# MCMC 결과: summary

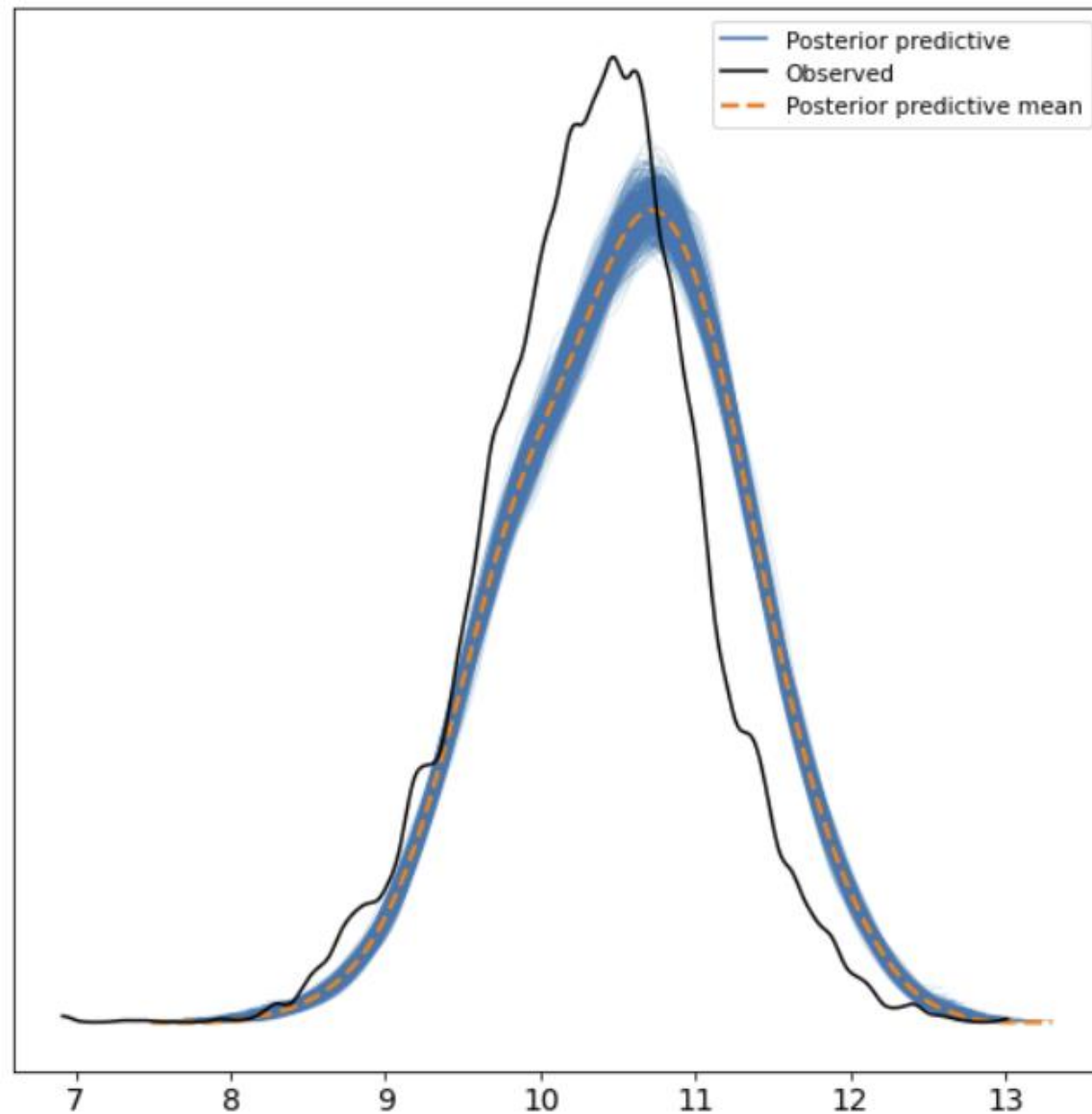
In [33]: 1 az.summary(trace\_independent\_regression\_full)

Got error No model on context stack. trying to find log\_likelihood in translation.  
 /Users/happyducky/opt/anaconda3/lib/python3.8/site-packages/arviz/data/io\_pymc3\_3x.py:98: FutureWarning: Using `from\_pymc3` without the model will be deprecated in a future release. Not using the model will return less accurate and less useful results. Make sure you use the model argument or call from\_pymc3 within a model context.  
 warnings.warn(

Out [33]:

	mean	sd	hdi_3%	hdi_97%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
<b>intercept_1</b>	6.750	0.050	6.657	6.843	0.001	0.001	1864.0	2406.0	1.0
<b>coeff_2</b>	-0.003	0.000	-0.003	-0.003	0.000	0.000	2271.0	2347.0	1.0
<b>coeff_3</b>	-0.001	0.001	-0.002	0.001	0.000	0.000	2462.0	2293.0	1.0
<b>coeff_4</b>	0.001	0.000	0.000	0.002	0.000	0.000	4111.0	2285.0	1.0
<b>coeff_5</b>	0.003	0.000	0.003	0.003	0.000	0.000	4272.0	2955.0	1.0
<b>coeff_6</b>	0.009	0.001	0.008	0.010	0.000	0.000	3961.0	2913.0	1.0
<b>coeff_8</b>	0.859	0.009	0.842	0.876	0.000	0.000	2239.0	2642.0	1.0
<b>coeff_1</b>	0.286	0.010	0.268	0.306	0.000	0.000	2449.0	2323.0	1.0
<b>coeff_7</b>	0.126	0.014	0.098	0.150	0.000	0.000	2609.0	2532.0	1.0
<b>error</b>	0.274	0.002	0.269	0.278	0.000	0.000	4517.0	2379.0	1.0

# “Observed Y” vs “Posterior predictive Y”





# Validation

♡ Validation set에 위의 independent\_regression\_full을 fitting시킴.

♡ Bayesian vs Frequentist

Frequentist: point estimate만 반환

Bayesian: 분포로서 반환

=> prediction 위해서는 Bayesian 방식을 써서 posterior 분포를 구했음지라도 하나의 추정치가 필요

=> posterior mean을 사용

♡ MAE, RMSE

```
mae = 0.20821793668513697  
rmse = 0.2714614655785255
```

# Submission

42등 나이스~

DACON

커뮤니티

대회

교육

랭킹

더보기



태주

대회안내









데이터

코드 공유

토크

리더보드

제출

38	rollcake		19,373.18717	3	2년 전
39	ES		20,134.06269	4	3년 전
40	ahj		20,907.17812	2	일 년 전
41	선호		21,018.25906	3	일 년 전
42	태주		23,697.9639	1	2분 전
43	sally		25,415.04747	1	2년 전
44	bossal00		25,893.89176	2	6달 전
45	bacon		28,627.24332	2	2년 전

---

Part 5

# 기타 방법론 소개

# 모델링 기타 방법론 소개

Scikit-learn  
LinearRegression



Linear Regression

Python lightgbm  
+  
Optuna



LightGBM

# Linear Regression - Coding

기타 방법론 소개

```
[32] from sklearn.linear_model import LinearRegression
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```



LinearRegression 패키지

```
cut = int(len(train_df)*0.8)
h_train = train_df[:cut]
h_valid = train_df[cut:]

h_train_X = h_train.drop('log_price', axis=1)
h_train_y = h_train['log_price']
h_valid_X = h_valid.drop('log_price', axis=1)
h_valid_y = h_valid['log_price']
```



데이터 분할

```
[36] dat = LinearRegression()
dat.fit(h_train_X, h_train_y)

LinearRegression()

[38] y_pred = dat.predict(h_valid_X)
```



Fitting 및 prediction

```
[19] def RMSE(y, y_pred):
    rmse = mean_squared_error(y, y_pred) ** 0.5
    return rmse

RMSE(h_valid_y, y_pred)

0.27355908467866197
```



RMSE: 0.2735

# Light GBM + Optuna 기타 방법론 소개

	Model	Score
0	LinearRegression	0.288778
1	Ridge	0.288777
2	Lasso	0.299014
3	ElasticNet	0.297440
4	DecisionTreeRegressor	0.309207
5	RandomForestRegressor	0.276021
6	XGBRegressor	0.247439
7	LGBMRegressor	0.239455

```

from optuna.samplers import TPESampler

sampler = TPESampler(seed=10)

def objective(trial):
    dtrain = lgb.Dataset(h_train_X, label=h_train_y)
    dtest = lgb.Dataset(h_valid_X, label=h_valid_y)

    param = {
        'objective': 'regression', # 회귀
        'verbose': -1,
        'metric': 'rmse',
        'max_depth': trial.suggest_int('max_depth', 3, 15),
        'learning_rate': trial.suggest_loguniform('learning_rate', 1e-8, 1e-2),
        'n_estimators': trial.suggest_int('n_estimators', 100, 3000),
        'min_child_samples': trial.suggest_int('min_child_samples', 5, 100),
        'subsample': trial.suggest_loguniform('subsample', 0.4, 1),
    }

    model = lgb.LGBMRegressor(**param)
    lgb_model = model.fit(h_train_X, h_train_y, eval_set=[(h_valid_X, h_valid_y)], verbose=0, early_stopping_rounds=25)
    rmse = RMSE(h_valid_y, lgb_model.predict(h_valid_X))
    return rmse

study_lgb = optuna.create_study(direction='minimize', sampler=sampler)
study_lgb.optimize(objective, n_trials=100)

```

```

[I 2021-10-03 21:46:14,337] A new study created in memory with name: no-name-efe5cd63-f0e7-4b3e-8af4-1f6830447f94
[I 2021-10-03 21:46:29,273] Trial 0 finished with value: 0.7371358593423419 and parameters: {'max_depth': 13, 'learning_rate': 1.332022915
[I 2021-10-03 21:46:44,414] Trial 1 finished with value: 0.7369466723386557 and parameters: {'max_depth': 5, 'learning_rate': 1.5430400149
[I 2021-10-03 21:46:45,360] Trial 2 finished with value: 0.4897832583187339 and parameters: {'max_depth': 11, 'learning_rate': 0.005252427

```

# Light GBM + Optuna

기타 방법론 소개

```
trial = study_lgb.best_trial
trial_params = trial.params
print('Best Trial: score {},#nparams {}'.format(trial.value, trial_params))
```

```
Best Trial: score 0.2073642671090797,
params {'max_depth': 13, 'learning_rate': 0.009480267802321527, 'n_estimators': 2566, 'min_child_samples': 84, 'subsample': 0.5742064444292969}
```

Best Score: 0.207

```
] final_lgb_model = lgb.LGBMRegressor(**trial_params)
final_lgb_model.fit(train_X, train_y)
final_lgb_pred = final_lgb_model.predict(test_df)
```

감사합니다