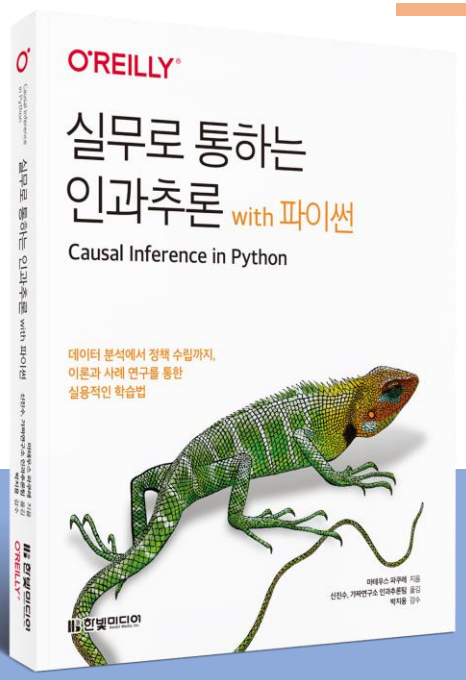
가짜연구소

Causal-Lab

『실무로 통하는 인과추론 with 파이썬』 특강



게임 업데이트 효과에 따른 이질적 처치효과 추정

박이삭

발표자 소개

박이삭 Linkedin

게임 회사 데이터 분석 (2018~)

- 유저 LTV 추정
- LTV 극대화

통계학 전공

아프지 않고 건강하게 오래오래 사는게 목표

취미: 우쿨렐레

※ 모든 그림은 교재 Github 자료와 직접 코드를 작성하여 만들었습니다. (+ChatGPT)

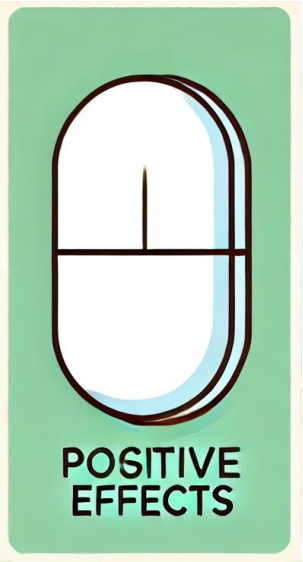
Contents

Causal-Lab

이질적 처치효과 복습

Causal-Lab

이상적인 상황과 달리 현실 세계에서는 처치를 가하면 다양한 효과가 나타납니다.



이질적 처치효과 복습

Causal-Lab

이상적인 상황과 달리 현실 세계에서는 처치를 가하면 다양한 효과가 나타납니다.

$$ATE = E[Y_1 - Y_0]$$

$$CATE = E[Y_1 - Y_0|X]$$

※ Conditional ATE

$E[Y_1 - Y_0|남성]$ $E[Y_1 - Y_0|나이]$

$E[Y_1 - Y_0|여성]$ $E[Y_1 - Y_0|신장]$

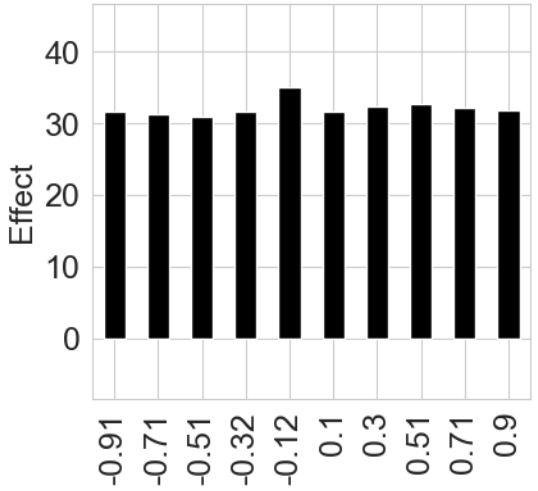
이질적 처치효과 복습

Causal-Lab

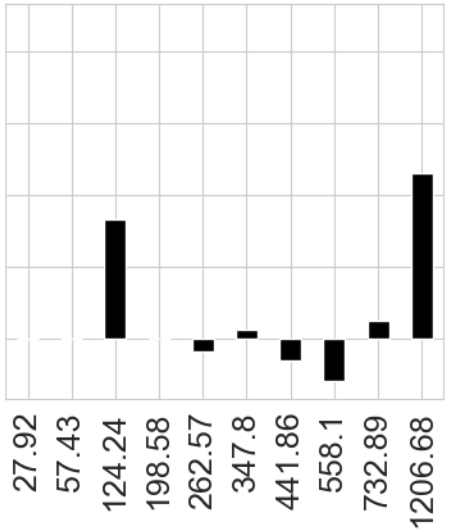
회귀분석으로 처치 효과 구하기

ATE	CATE
$y = \beta_0 + \beta_1 T + \beta_2 X + e$	$y = \beta_0 + \beta_1 T + \beta_2 X + \beta_3 TX + e$
$\frac{\delta y}{\delta T} = \beta_1$	$\frac{\delta y}{\delta T} = \beta_1 + \beta_3 X$

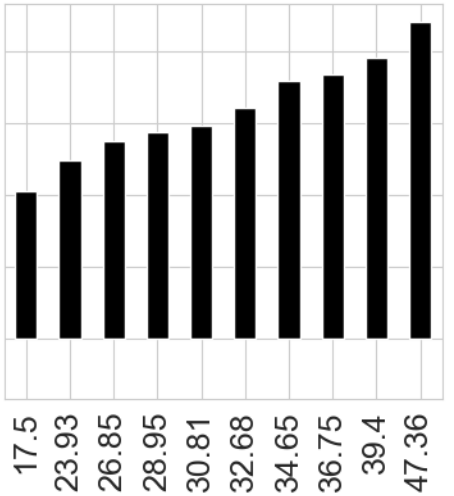
이질적 처치효과 복습



난수모델(rand_m_pred)이
예측한 분위수



머신러닝 모델(ml_pred)이
예측한 분위수



CATE 모델(cate_pred)이
예측한 분위수

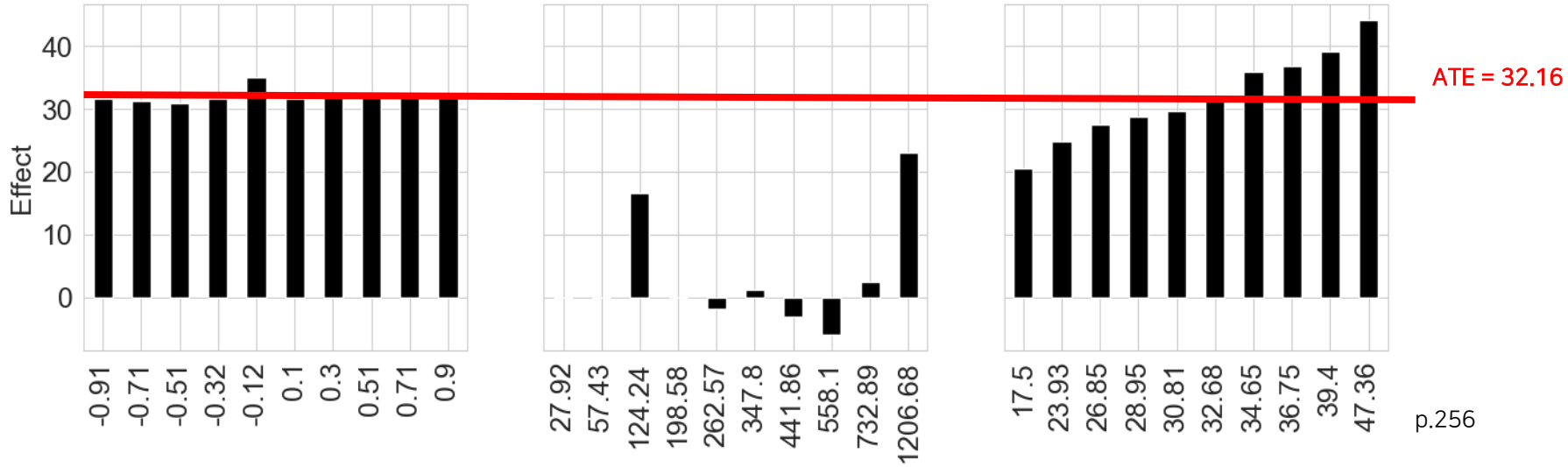
p.256

목적: X(공변량)에 대한 T(처치)의 매출 민감도를 측정하길 원함 p.246

```
1  import statsmodels.formula.api as smf
2
3  X = ["C(month)", "C(weekday)", "is_holiday", "competitors_price"]
4  regr_cate = smf.ols(f"sales ~ discounts*({'+'.join(X)})",
5                      data=data).fit()
```

Executed at 2024.07.14 11:10:29 in 422ms

이질적 처치효과 복습



p.256

난수모델(rand_m_pred)이
예측한 분위수

머신러닝 모델(ml_pred)이
예측한 분위수

CATE 모델(cate_pred)이
예측한 분위수

```
# 간단하게!
X = ["C(month)"]
regr_model = smf.ols(f"sales ~ discounts*({'+'.join(X)})",data=train).fit()
```

이질적 처치효과 복습

Causal-Lab

```python
# 간단하게!
X = ["C(month)"]
regr_model = smf.ols(f"sales ~ discounts*({'+'.join(X)})",data=train).fit()
```

$$y = \beta_0 + \beta_1 T + \beta_2 X + \beta_3 TX + e$$

$$\frac{\delta y}{\delta T} = \beta_1 + \beta_3 X$$

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	49.9739	7.728	6.467	0.000	34.824	65.124
C(month)[T.2]	-9.1498	11.160	-0.820	0.412	-31.028	12.728
			...			
discounts	38.3301	0.555	69.025	0.000	37.241	39.419
discounts:C(month)[T.2]	-6.1997	0.778	-7.965	0.000	-7.726	-4.674
discounts:C(month)[T.3]	-7.0223	0.786	-8.931	0.000	-8.564	-5.481
discounts:C(month)[T.4]	-8.7973	0.780	-11.277	0.000	-10.327	-7.268
discounts:C(month)[T.5]	-11.3028	0.768	-14.726	0.000	-12.808	-9.798
discounts:C(month)[T.6]	-11.7491	0.798	-14.728	0.000	-13.313	-10.185
discounts:C(month)[T.7]	-6.5614	0.759	-8.647	0.000	-8.049	-5.074
discounts:C(month)[T.8]	-11.4423	0.754	-15.184	0.000	-12.920	-9.965
discounts:C(month)[T.9]	-7.3423	0.776	-9.458	0.000	-8.864	-5.820
discounts:C(month)[T.10]	-3.6777	0.759	-4.843	0.000	-5.166	-2.189
discounts:C(month)[T.11]	-2.6320	0.801	-3.285	0.001	-4.203	-1.061
discounts:C(month)[T.12]	-2.0715	0.776	-2.668	0.008	-3.594	-0.549

이질적 처치효과 복습

Causal-Lab

```python
# 간단하게!
X = ["C(month)"]
regr_model = smf.ols(f"sales ~ discounts*({'+'.join(X)})",data=train).fit()
```

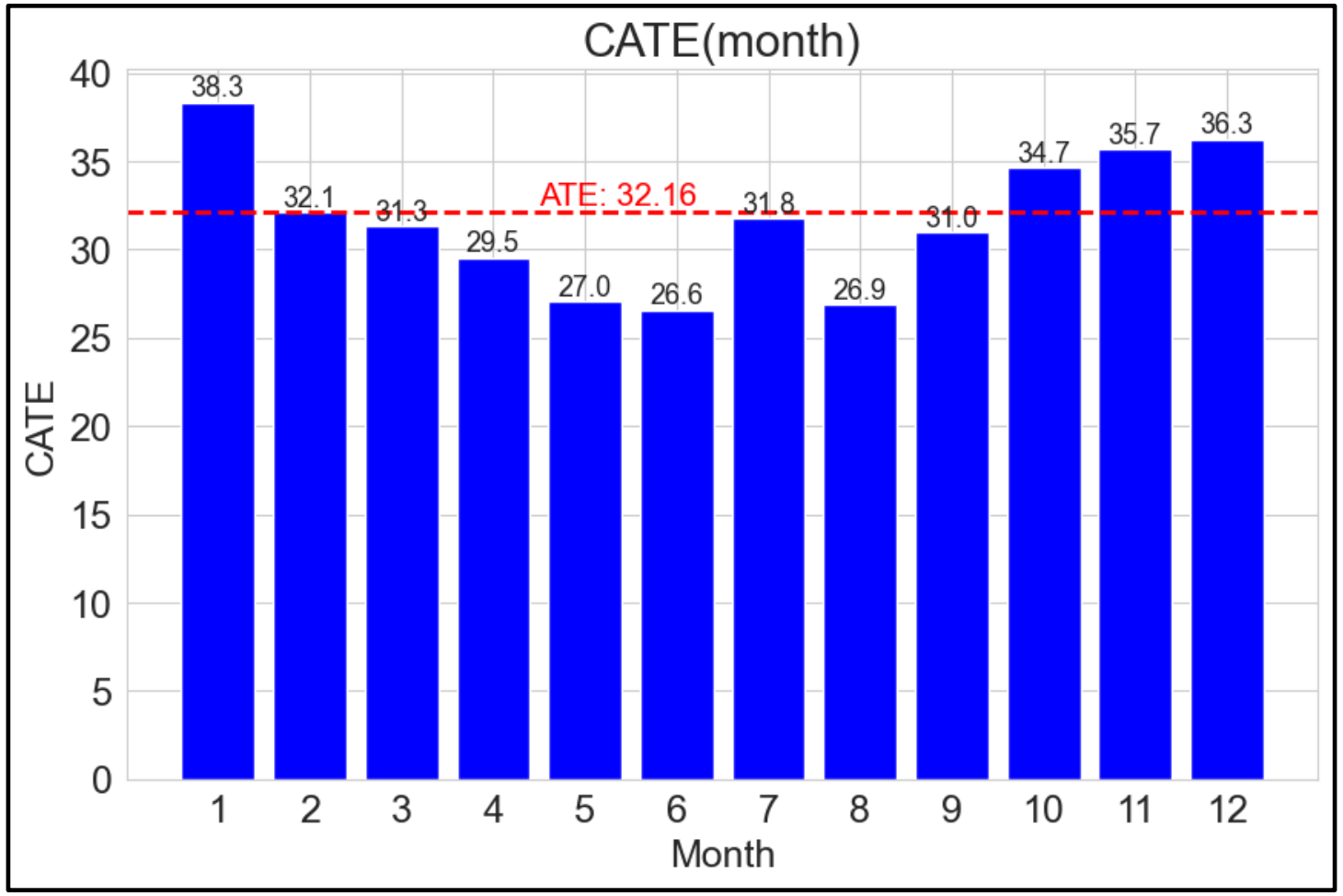
$$y = \beta_0 + \beta_1 T + \beta_2 X + \beta_3 TX + e$$

$$\frac{\delta y}{\delta T} = \beta_1 + \beta_3 X$$

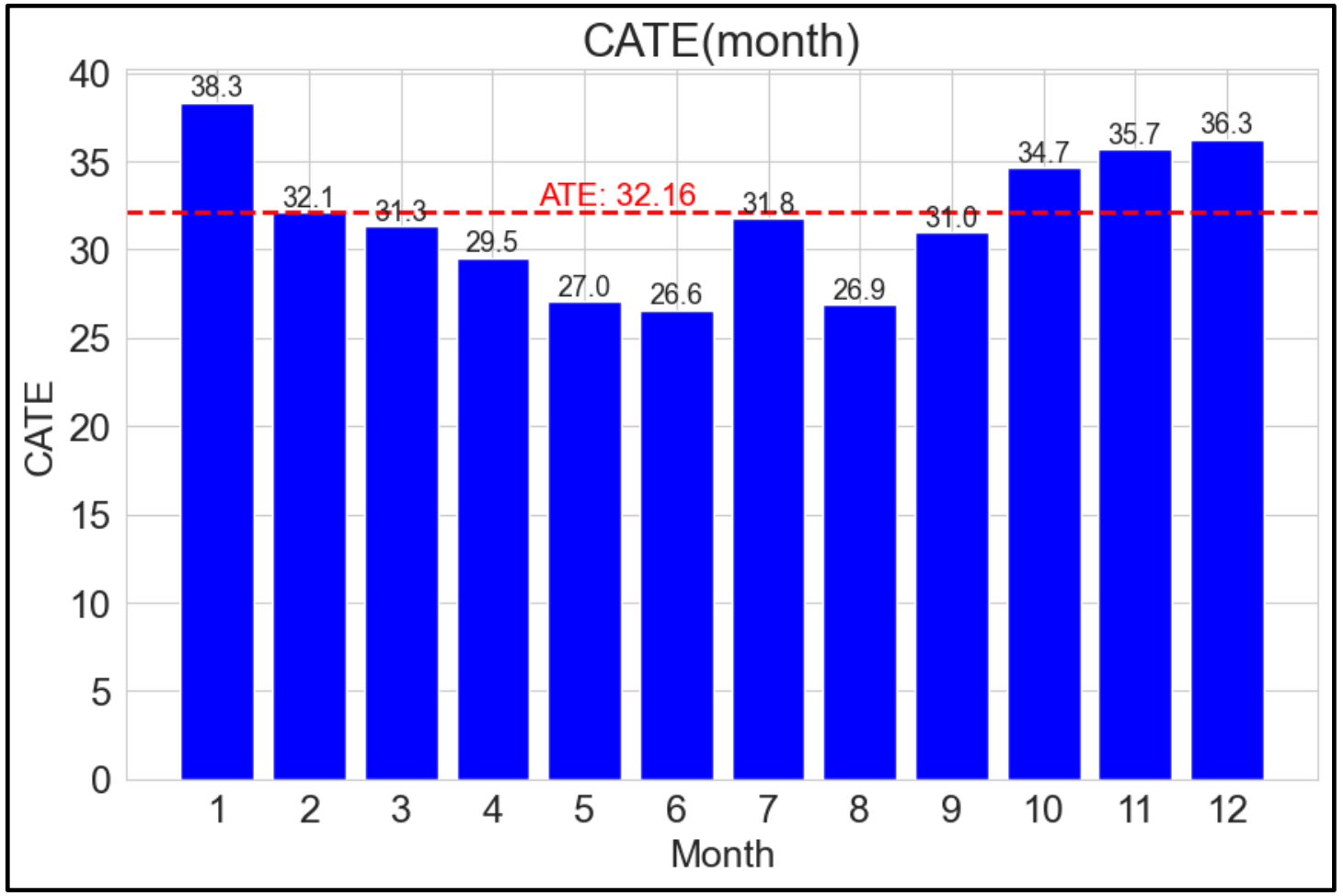
discounts	38.3301
discounts:C(month)[T.2]	-6.1997
discounts:C(month)[T.3]	-7.0223
discounts:C(month)[T.4]	-8.7973
discounts:C(month)[T.5]	-11.3028
discounts:C(month)[T.6]	-11.7491
discounts:C(month)[T.7]	-6.5614
discounts:C(month)[T.8]	-11.4423
discounts:C(month)[T.9]	-7.3423
discounts:C(month)[T.10]	-3.6777
discounts:C(month)[T.11]	-2.6320
discounts:C(month)[T.12]	-2.0715

β_1	β_3	$X(= Month)$	$\frac{\delta y}{\delta T}$
38.3301	0	1월	38.3301
38.3301	-6.1997	2월	32.1304
38.3301	-7.0223	3월	31.3078
…			
38.3301	-2.0705	12월	35.2596

이질적 처치효과 복습


CATE(month)
CATE
Month
ATE: 32.16
38.3
32.1
31.3
29.5
27.0
26.6
31.8
26.9
31.0
34.7
35.7
36.3
0
5
10
15
20
25
30
35
40
1
2
3
4
5
6
7
8
9
10
11
12

이질적 처치효과 복습



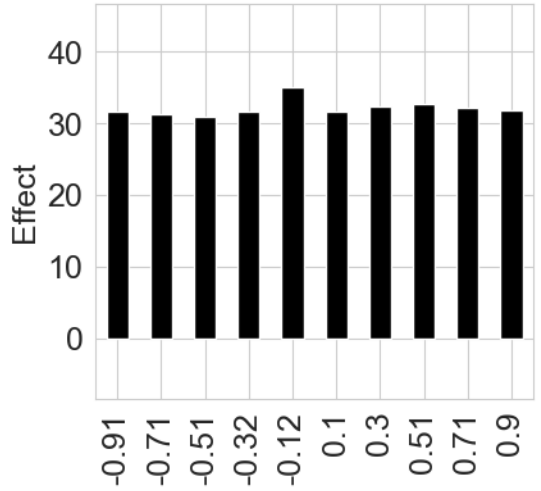
각 월별 ATE값 구해본다면...

※ CATE 값은 참 값과 다를 수 있으니,
값 자체보다는 순서를 이용해야 합니다.

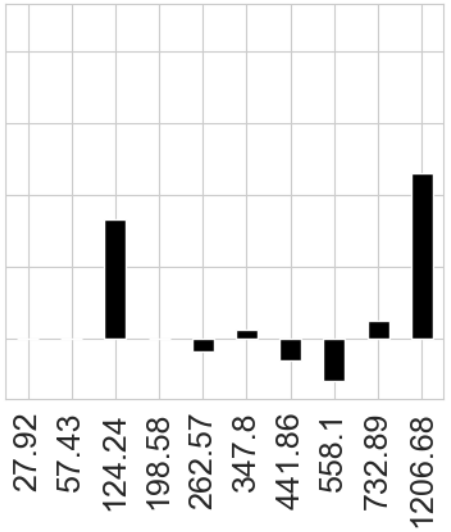
- 1월 ATE: 34.275
- 2월 ATE: 31.372
- ...

데이터를 이용해 구했더니 CATE값과는 다른 값이 나왔음

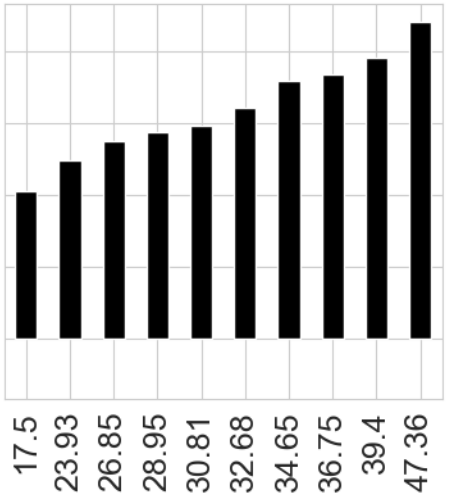
CATE 모델 평가 – 누적 효과 곡선



난수모델(rand_m_pred)이
예측한 분위수



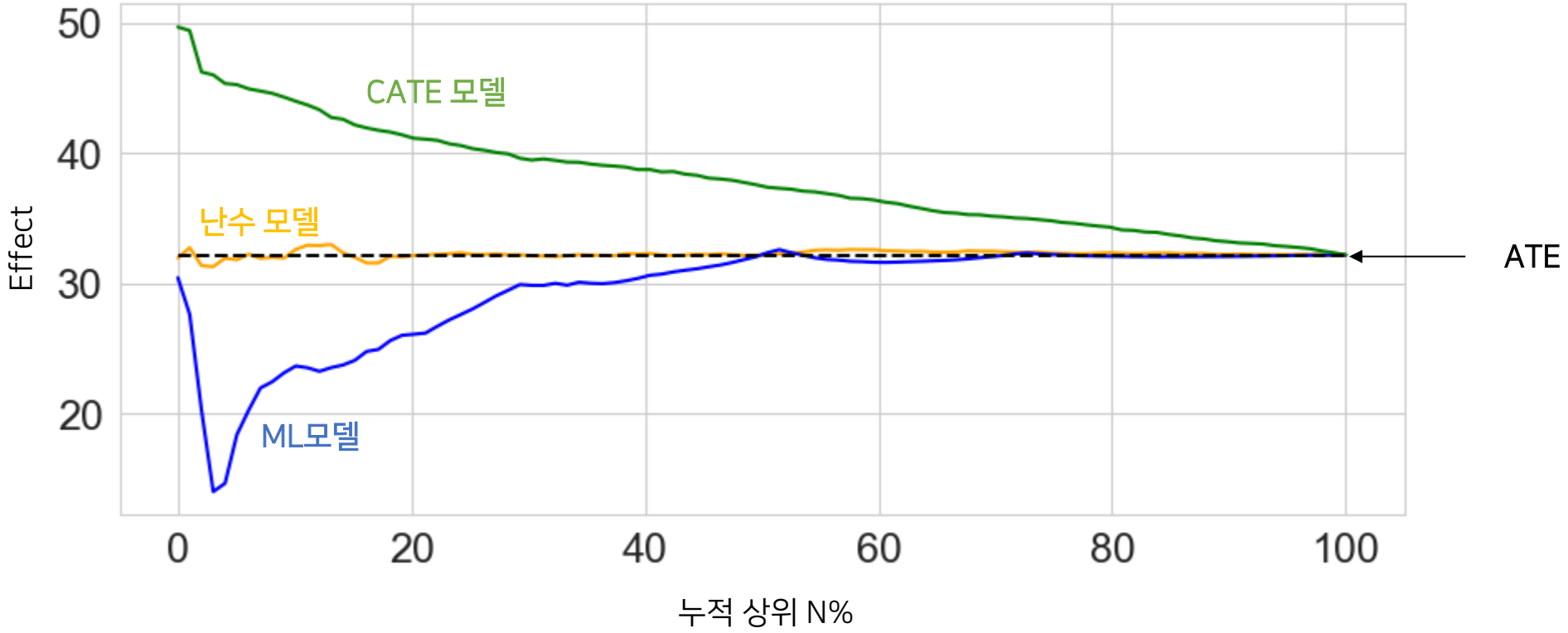
머신러닝 모델(ml_pred)이
예측한 분위수



CATE 모델(cate_pred)이
예측한 분위수

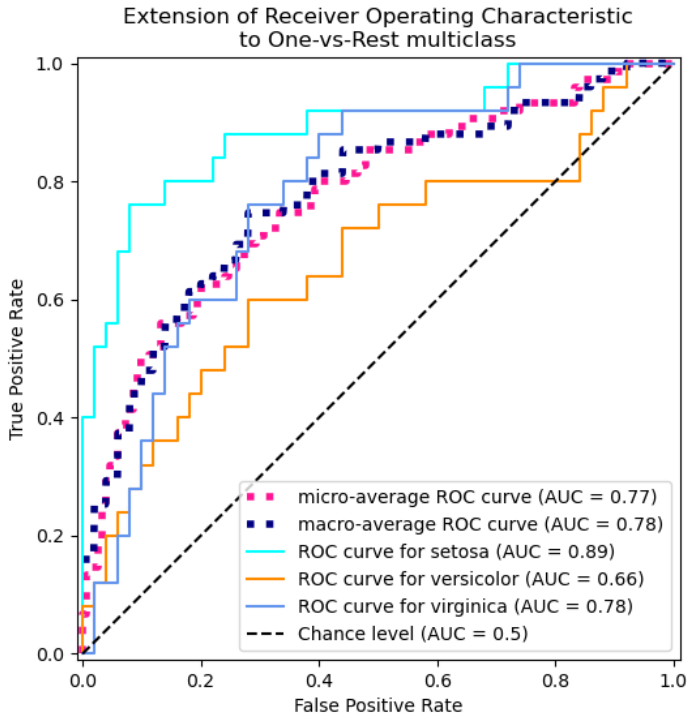
- CATE를 구한다는 건 X조건에 따른 순서를 구하는 것
- 모델이 순서를 잘 반영하는지 확인하는 방법
  분위수별 CATE 예측값이 점점 커지거나 작아지면 잘 예측한 것
- 1% 단위로 CATE를 구해보고 그 형태를 확인

CATE 모델 평가 – 누적 효과 곡선



- X축이 1일 때는 누적 상위 1%, X축이 2일 때는 누적 상위 2%의 효과를 확인하는 그림입니다.
- CATE 모델은 매우 높게 시작하여 점차 ATE로 수렴합니다.
- 난수 모델과 ML 모델은 ATE 주변에서 진동하고 빠르게 수렴합니다.

CATE 모델 평가 – 누적 효과 곡선



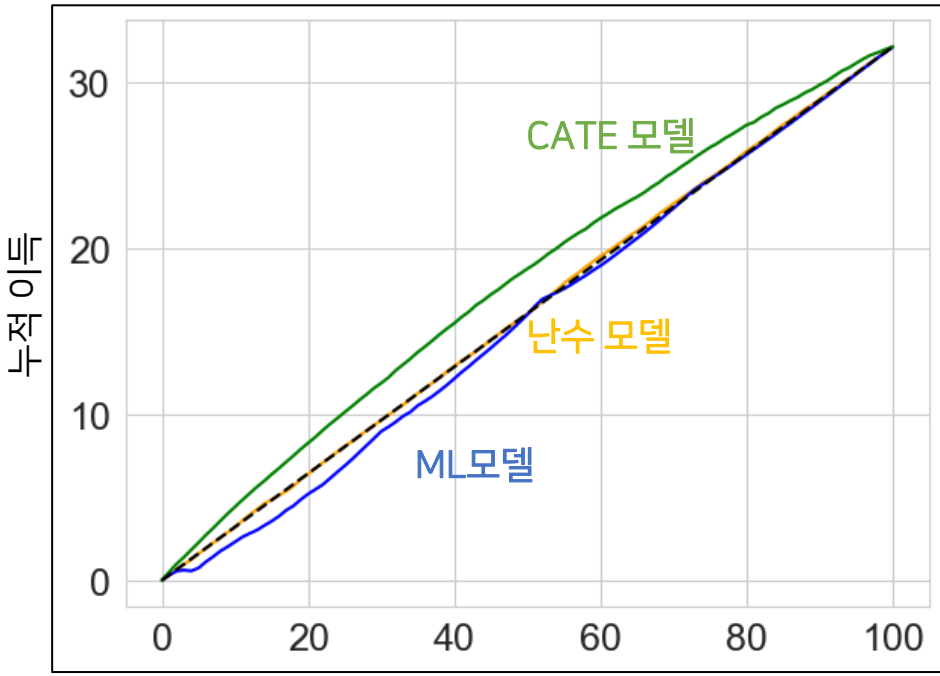
Sklearn ROC-AUC 예시 Link

- 누적 효과곡선은 CATE 예측 성능을 확인하는 효과적인 방법
- 매번 그림을 그려 확인하는 대신, 곡선과 ATE 사이의 면적을 계산해 단일 숫자로 표현할 수 있습니다. (ROC 커브의 AUC 값처럼)
- 단점: X가 1일 때 가장 큰 값을 차지하지만, 누적 상위 1% 표본 크기가 작습니다.
  이를 보완하기 위해 **누적 이득 곡선**을 사용합니다.

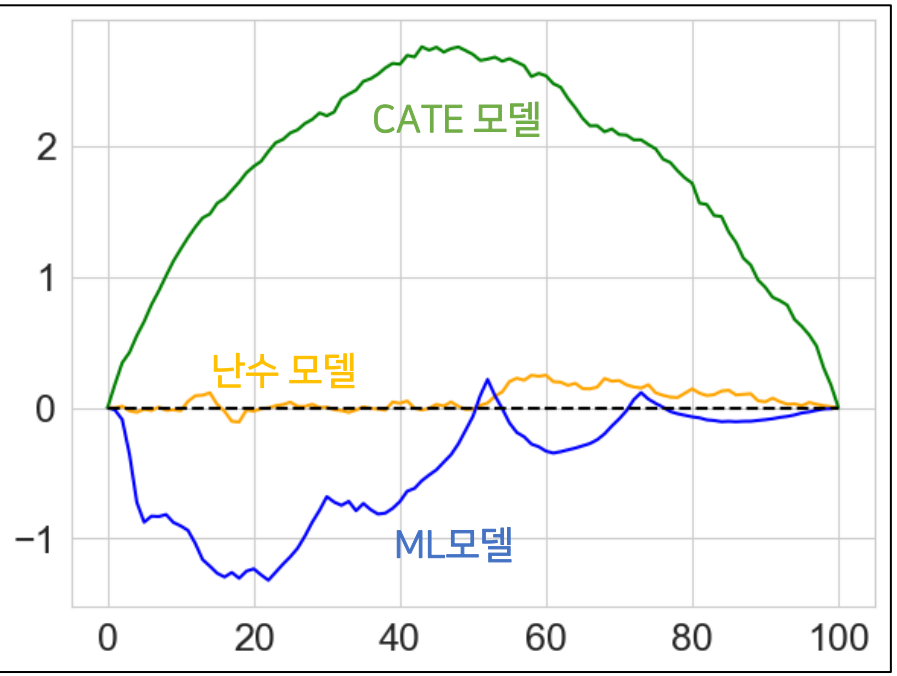
CATE 모델 평가 – 누적 이득 곡선

Causal-Lab

누적 이득 곡선



정규화된 누적 이득 곡선



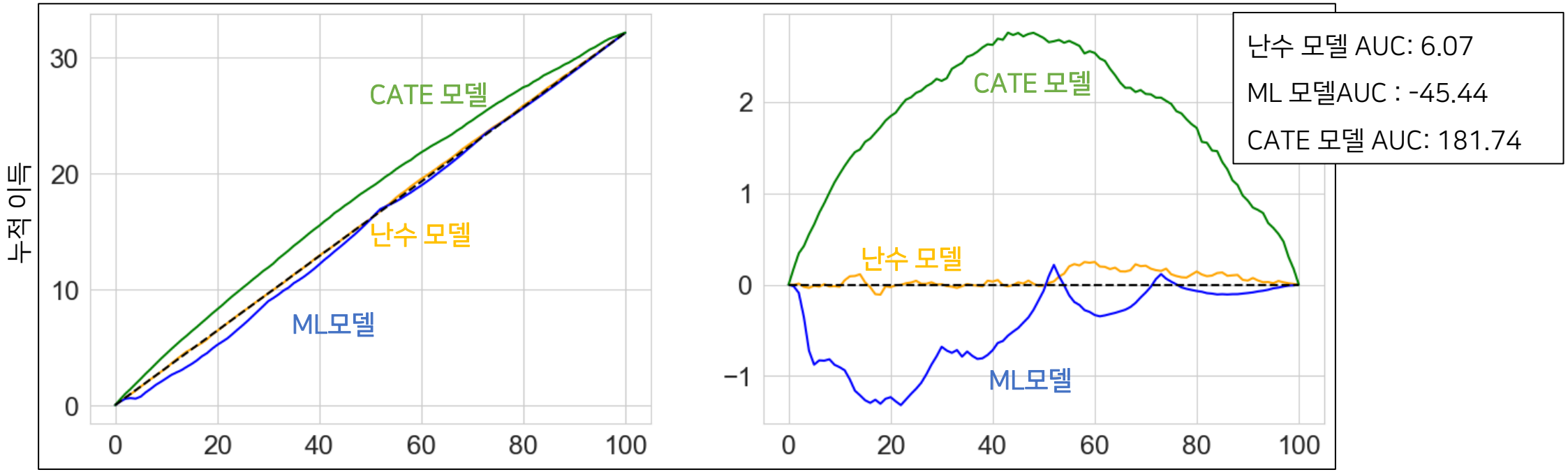
- 누적 효과 곡선의 문제점: 상위 % 표본이 부족해 불확실성이 크다는 단점 존재
- 해결책: 효과를 구할 때마다 누적 표본($\frac{N_{cum}}{N}$)을 곱해 정규화 (정규화로 인해 상위 % 효과가 축소됨)
- 이를 "**누적 이득 곡선**"이라 부르며, 왼쪽 그림으로 표현.
- 오른쪽 그림은 ATE로 정규화한 그림.
- AUC 값으로 모델의 CATE 순서 예측 성능 확인 가능.

CATE 모델 평가 – 누적 이득 곡선

Causal-Lab

누적 이득 곡선

정규화된 누적 이득 곡선



- 누적 효과 곡선의 문제점: 상위 % 표본이 부족해 불확실성이 크다는 단점 존재
- 해결책: 효과를 구할 때마다 누적 표본($\frac{N_{cum}}{N}$)을 곱해 정규화 (정규화로 인해 상위 % 효과가 축소됨)
- 이를 **"누적 이득 곡선"**이라 부르며, 왼쪽 그림으로 표현.
- 오른쪽 그림은 ATE로 정규화한 그림.
- AUC 값으로 모델의 CATE 순서 예측 성능 확인 가능.

CATE 모델 평가 – 정리

Causal-Lab

- CATE? X를 조건으로 하는 조건부 평균 처치 효과이며, Y를 T로 미분해서 얻음

$$y = \beta_0 + \beta_1 T + \beta_2 X + \beta_3 TX + e$$

$$\frac{\delta y}{\delta T} = \beta_1 + \beta_3 X$$

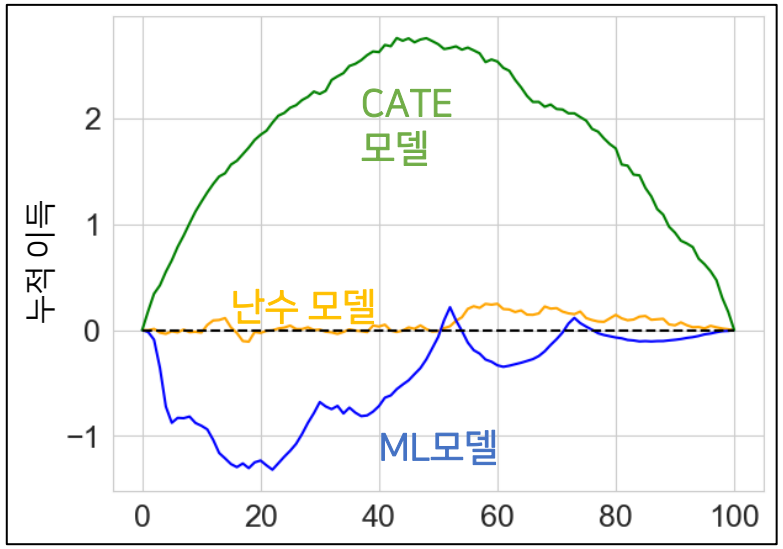
- 단, CATE는 값 자체가 아닌 순서를 사용해야 함 (회귀 계수 추정값이기 때문)

- 모델의 순서가 잘 정렬되었는지 확인하는 방법: "**누적 효과 곡선**", "**누적 이득 곡선**"
- 이를 이용해 AUC(Area Under The Curve)를 구하면 모델을 숫자 하나로 비교 가능

※ 정확한 CATE 값을 예측하는 것이 중요하다면 "누적 이득 곡선"만 이용하는 것은 지양 (하지만 아쉽게도 대안은 없습니다)

※ 모든 인과추론 과정이 그렇지만 교란이 없는 데이터를 이용하는 것이 핵심

정규화된 누적 이득 곡선



CATE 모델 평가 – 목표변환

$$E[Y_i^*] = \tau_i$$

$$Y_i^* = \frac{(Y_i - \hat{\mu}_y(X_i))(T_i - \hat{\mu}_t(X_i))}{(T_i - \hat{\mu}_t(X_i))^2} = \frac{Y_i - \hat{\mu}_y(X_i)}{T_i - \hat{\mu}_t(X_i)}$$

※ $\hat{\mu}_y$: Y를 예측한 모델의 예측값

※ $\hat{\mu}_t$: T를 예측한 모델의 예측값

즉, τ_i를 예측하는데 효과적이라면 다음과 같은 MSE 공식도 이용할 수 있습니다.

$$MSE(\,\mathrm{CATE}(X_i)\,, Y_i^*)$$

만약, 앞서 구한 3가지 모델의 CATE 예측값이 τ_i를 예측하는데 효과적이라면 MSE는 0에 가까운 값을 가질 것 입니다.

Y_i^*는 정확하게 구할수록 분모가 0에 가까워지기 때문에 Weighted MSE 값을 이용해 구합니다.

$$= \frac{\sum_{i=1}^{n} w_i(CATE(X_i) - Y_i^*)}{\sum_{i=i}^{n} w_i}$$

단, $w_i = (T_i - \hat{\mu}_t(X_i)^2)$

CATE 모델 평가 – 목표변환

Causal-Lab

```
X = ["C(month)", "C(weekday)", "is_holiday", "competitors_price"]

y_res = smf.ols(f"sales ~ {'+'.join(X)}", data=test).fit().resid # Y를 예측하는 모델의 잔차
t_res = smf.ols(f"discounts ~ {'+'.join(X)}", data=test).fit().resid # T를 예측하는 모델의 잔차

tau_hat = y_res/t_res
```

Executed at 2024.07.20 13:25:19 in 275ms

```
from sklearn.metrics import mean_squared_error

for m in ["rand_m_pred", "ml_pred", "cate_pred"]:
    wmse = mean_squared_error(tau_hat, test_pred[m],
                              sample_weight=t_res**2) # weighted MSE
    print(f"MSE for {m}:", wmse)
```

- 난수 모델의 MSE : 1115.80
- ML 모델의 MSE : 576256.74
- CATE 모델의 MSE : 42.90

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

대부분 게임사는 복귀유저 캠페인을 진행 중

- 신규 유저대비 획득 비용이 낮은 장점 (=낮은 CAC)
- 신유 유저 대비 로그인 유지 비율이 높음 (=높은 Retention)

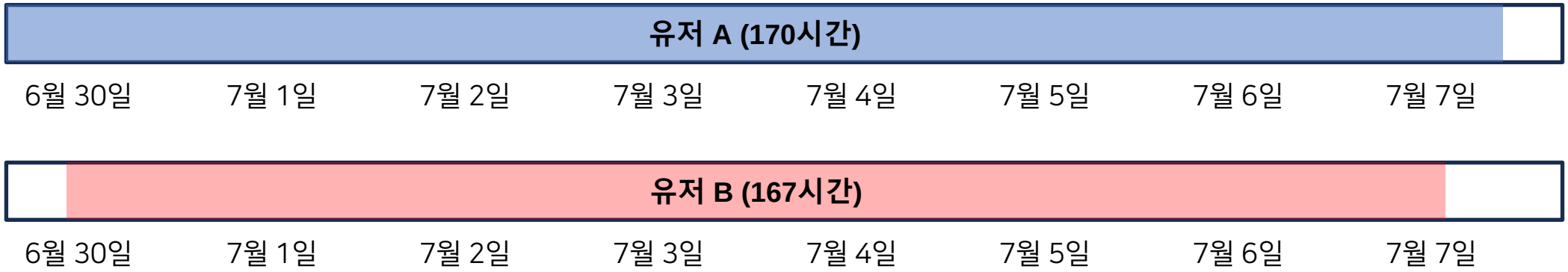
사례

이번 업데이트로, 7일만에 복귀 시, 아이템을 주는 업데이트를 진행

즉, 마지막 접속일로부터 7일 이상 지나면 아이템 지급 대상

아이템 지급 효과는 있었지만... **문제가 발생!**

업데이트 효과에 따른 이질적 처치 효과 추정



- 미 접속 기간이 7일 이상인 유저는 복귀 보상을 획득
- A와 B유저는 6월 30일 마지막 접속 후, 7월 7일 복귀
- 하지만 A만 복귀 보상을 받았는데, 이유는 마지막 접속으로 부터 7일(=168시간)이 지났기 때문

이거... 실험을 할 수 있는 조건인데??!

T: 복귀 보상 지급

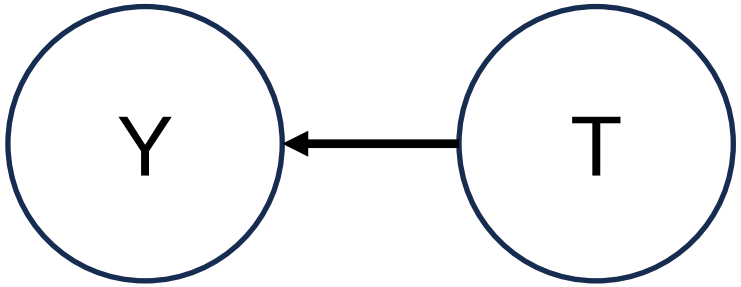
Y: Retention

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

diff_hour	Reward(T)	7일 뒤, Retention (Y)
158	0	0
176	1	1
163	0	0
164	0	0
168	1	1
172	1	0
168	1	0
167	0	0
167	0	0
179	1	0

<데이터 셋트 예시>



- 복귀 7일차 유저 중, 168시간 이후 복귀 유저인 경우 Reward(보상) 획득
- 168시간 미만의 경우 Reward를 획득하지 못함
- 다시 7일 뒤, 해당 유저들이 접속여부를 측정
- 로지스틱 회귀분석 이용

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

회귀분석으로 처치 효과 구하기

ATE

$$y = \frac{1}{1 + e^{-(\beta_0 + \beta_1 T)}}$$

$$ATE = \frac{1}{N}\sum_{i=1}^{N} P(Y|T=1) - P(Y|T=0)$$

CATE

$$y = \frac{1}{1 + e^{-(\beta_0 + \beta_1 T + \beta_2 X + \beta_3 TX)}}$$

$$CATE = \frac{1}{N}\sum_{i=1}^{N} P(Y|T=1, X) - P(Y|T=0, X)$$

업데이트 효과에 따른 이질적 처치 효과 추정

diff_hour	Reward(T)	7일 뒤, Retention (Y)
158	0	0
176	1	1
163	0	0
164	0	0
168	1	1
172	1	0
168	1	0
167	0	0
167	0	0
179	1	0

<데이터 셋트 예시>

- 복귀 7일차 유저 중, 168시간 이후 복귀 유저인 경우 Reward(보상) 획득
- 168시간 미만의 경우 Reward를 획득하지 못함
- 다시 7일 뒤, 해당 유저들이 접속여부를 측정

<집계 결과>

	복귀 보상	
7일뒤 접속	미획득	획득
X	10,193	12,777
O	625	1,006

오즈	6.13%	7.87%	=1006/12777
오즈비		1.2841	=7.87/6.13

```
T = df['reward']
Y = df['retention']
# 로지스틱 회귀모델 생성 및 학습
data = pd.DataFrame( data: {'Y': Y, 'T': T}, index = X.index)
# 로지스틱 회귀 모델 피팅
model = smf.logit( formula: 'Y ~ T', data).fit()

# 결과 요약 출력
print(model.summary())
```

```
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -2.7917      0.041    -67.747      0.000      -2.872      -2.711
T              0.2500      0.053      4.750      0.000       0.147       0.353
==============================================================================
```

```
np.exp(0.25): 1.2840254166877414
```

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

diff_hour	Reward(T)	7일 뒤, Retention (Y)
158	0	0
176	1	1
163	0	0
164	0	0
168	1	1
172	1	0
168	1	0
167	0	0
167	0	0
179	1	0

<데이터 셋트 예시>

```
# ATE값 확인
ATE = (
    df.query("reward == 1")["retention"].mean()
    - df.query("reward == 0")["retention"].mean()
)
print(f"ATE(%): {ATE*100:.2f}%")
```

ATE(%): 1.52%

$$ATE = \frac{1}{N}\sum_{i=1}^{N} P(Y|T=1) - P(Y|T=0)$$

- 복귀 7일차 유저 중, 168시간 이후 복귀 유저인 경우 Reward(보상) 획득
- 168시간 미만의 경우 Reward를 획득하지 못함
- 다시 7일 뒤, 해당 유저들이 접속여부를 측정

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

diff_hour	Reward(T)	7일 뒤, Retention (Y)	누적 결제금액(X)
158	0	0	17,000
176	1	1	26,000
163	0	0	8,000
164	0	0	0
168	1	1	0
172	1	0	0
168	1	0	0
167	0	0	0
167	0	0	180,000
179	1	0	86,000

<데이터 셋트 예시>

```python
X = df['cum_purchase_amt']
T = df['reward']
Y = df['retention']
# 로지스틱 회귀모델 생성 및 학습
data = pd.DataFrame( data: {'Y': Y, 'T': T, 'X': X}, index = X.index)
# 로지스틱 회귀 모델 피팅
model = smf.logit( formula: 'Y ~ T + X + T:X', data).fit()

# 결과 요약 출력
print(model.summary())
```

```
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -2.7960      0.041    -67.501      0.000      -2.877      -2.715
T              0.2451      0.053      4.625      0.000       0.141       0.349
X           5.831e-08   4.84e-08      1.204      0.228   -3.66e-08    1.53e-07
T:X         5.375e-08   6.67e-08      0.806      0.420    -7.7e-08    1.84e-07
==============================================================================
```

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

회귀분석으로 처치 효과 구하기

ATE

$$y = \frac{1}{1 + e^{-(\beta_0 + \beta_1 T)}}$$

$$ATE = \frac{1}{N}\sum_{i=1}^{N} P(Y|T=1) - P(Y|T=0)$$

CATE

$$y = \frac{1}{1 + e^{-(\beta_0 + \beta_1 T + \beta_2 X + \beta_3 TX)}}$$

$$CATE = \frac{1}{N}\sum_{i=1}^{N} P(Y|T=1, X) - P(Y|T=0, X)$$

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

```python
# 특정 값 x에 대해 CATE 계산
1 usage
def calculate_cate(x):
    coef = model.params
    p1 = 1 / (1 + np.exp(-(coef['Intercept'] + coef['T'] + coef['X']*x + coef['T:X']*x)))
    p0 = 1 / (1 + np.exp(-(coef['Intercept'] + coef['X']*x)))
    return p1 - p0

# 예시: X = 1에 대한 CATE
result = pd.DataFrame()
for x_value in range(0, 1000000+100000, 100000):
    cate_value = calculate_cate(x_value)
    print(f"CATE at X={x_value}: {cate_value}")
    df =pd.DataFrame( data: {'sales':x_value, 'cate':cate_value}, index =[0])
    result = pd.concat([result, df])
```

$$CATE = \frac{1}{N}\sum_{i=1}^{N} P(Y|T=1,X) - P(Y|T=0,X)$$

X	CATE
0	1.483%
100,000	1.527%
200,000	1.571%
300,000	1.616%
400,000	1.662%
500,000	1.708%
600,000	1.755%
700,000	1.802%
800,000	1.850%
900,000	1.898%
1,000,000	1.948%

업데이트 효과에 따른 이질적 처치 효과 추정

Causal-Lab

구분	Effect
ATE	1.52%
CATE(x = 누적 결제 금액)	X가 커질수록 증가

- ATE, CATE 값만 구하는 것으로 끝이 아님
- 보상 제공이 Retention을 1.52% 상승시키고, 결제금액이 많을수록 효과가 큼
- 분석 이유: 보상을 못 받은 유저와 받은 유저 간 차이 확인
- 168시간 대신 7일(day)로 구분하자고 유관 부서를 설득하는 자료로 사용 가능

Causal-Lab

감사합니다

CATE 모델 평가 – 누적 효과 곡선

Causal-Lab

```python
def cumulative_effect_curve(dataset: pd.DataFrame, prediction: str, y: str, t: str,
                            ascending: bool = False, steps: int = 100) -> np.ndarray:
    size = len(dataset) # 데이터 셋 크기 설정
    ordered_df = (dataset # prediction 값을 기준으로 순서를 재정렬
                  .sort_values(prediction, ascending=ascending)
                  .reset_index(drop=True))

    steps = np.linspace(size/steps, size, steps).round(0) # 균등하게 분포된 steps 개수만큼의 숫자를 생성. round는 정수형으로 바꿔주기 위해 추가

    return np.array([effect(ordered_df.query(f"index<={row}"), t=t, y=y) # effect 함수(회귀 계수 추정)를 이용해 상위 row%씩 효과 계산
                     for row in steps])

cumulative_effect_curve(test_pred, "cate_pred", "sales", "discounts")
```

Executed at 2024.07.20 09:28:00 in 219ms

1-10 100 rows × 1 columns

	0
0	49.651163
1	49.377125
2	46.203603
3	45.977068
4	45.317118