

ABC BANK

고객이탈예측

BY ML 분류모델

AIB_15_박준영

CONTENT

상황 설정

문제 정의

DATA SET

DATA EDA

MODELING

해석 및 결론

상황설정

- ABC Bank 데이터 직군
- 고객 이탈에 대한 분석 설명
- 비데이터 직군에게 설명



문제정의

고객이탈문제

- 은행 기본 수익 구조 :
예대마진 / 비이자수익
- 고객이 예치한 돈이 수익의 기반
- 이탈한 고객은 경쟁 회사로 유입

과정 및 목표

- **분류 알고리즘**을 활용한 고객 이탈 모델
- 고객이탈에 주요 영향을 미치는 특성 분석
- 어떤 고객을 대상으로 무슨 상품을 만들까?
- 어떤 고객에게 상품 관련 알리를 보낼까?

DATA SET

10,000

고객 자료 수

12

제공 특성 수

CHURN

target

0

결측, 중복



DATA EDA

특성

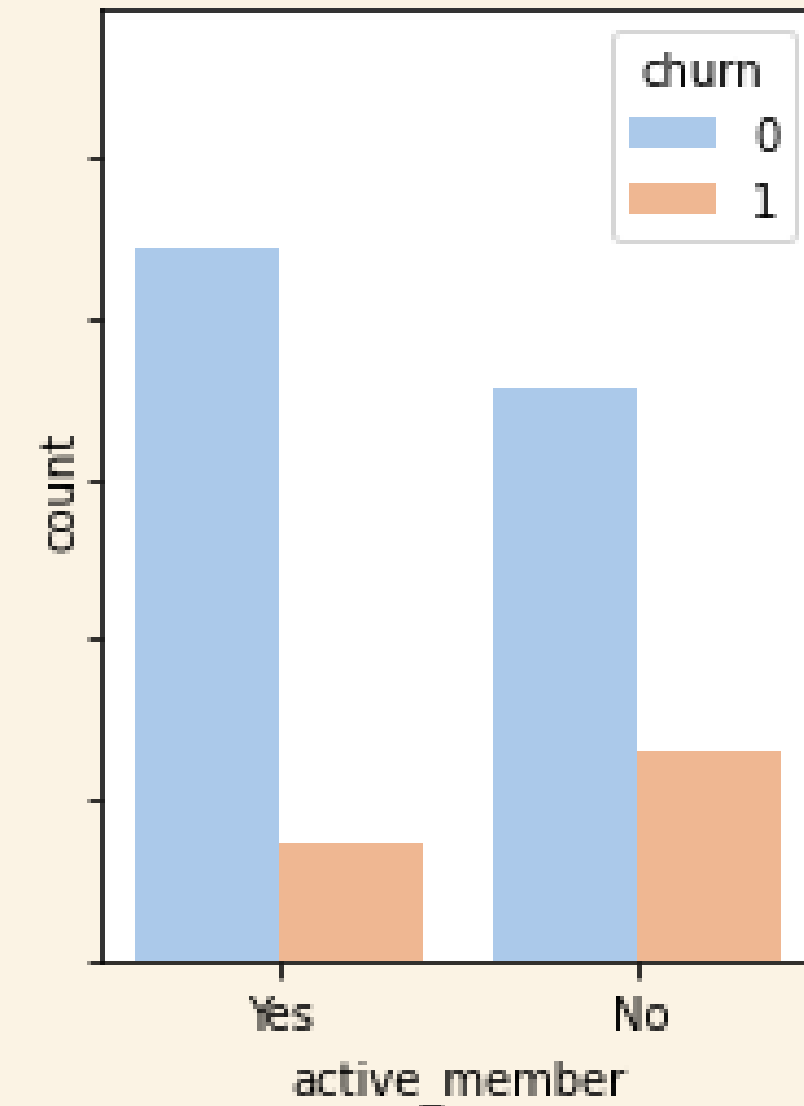
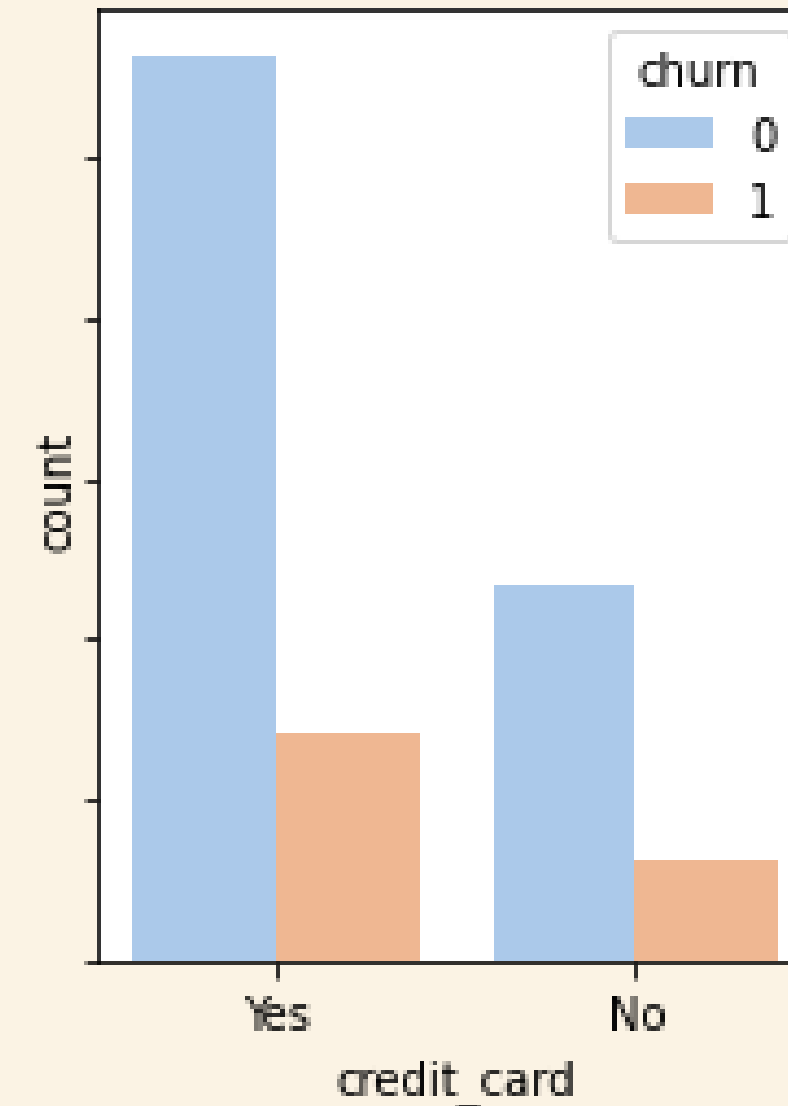
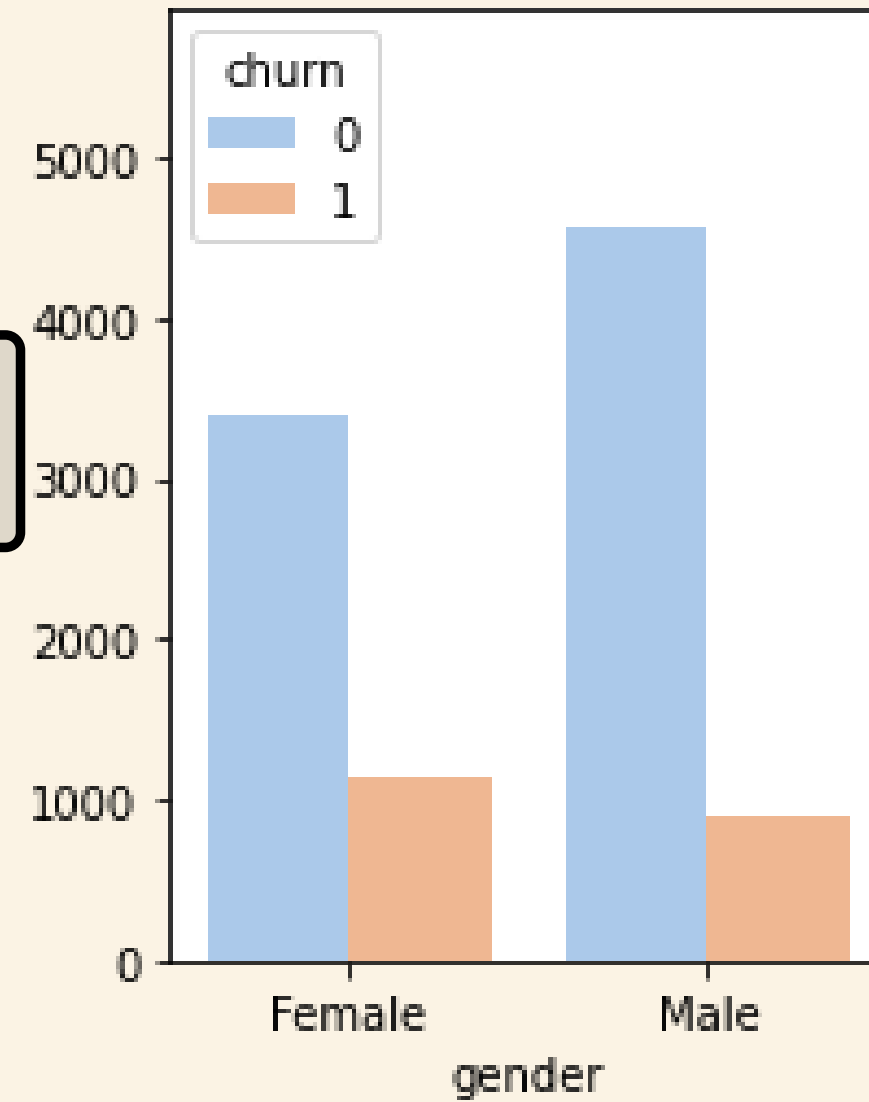
1. customer_id : 고객 고유 번호
2. credit_score : 고객 신용 점수
3. country : 국적
4. gender : 성별
5. age : 나이
6. tenure : 은행 계좌 보유 년수
7. balance : 계좌 잔액
8. products_number : 보유 상품
9. credit_card : 신용카드 보유 여부 (보유 : 1, 미보유 : 0)
10. active_member : 은행 정회원 여부 (회원 : 1, 회원아님 : 0)
11. estimated_salary : 추정 수입
12. churn : 이탈 여부 (이탈 : 1, 유지 : 0) \Rightarrow target

특성 편집

1. customer_id : 고객 고유 번호 \rightarrow 삭제
2. age_to_join : 상품 가입 나이 \rightarrow age - tenure
3. credit_rating : 신용 등급 \rightarrow 미국의 FICO 신용점수 기준에 따라 5등급으로 분류

DATA EDA

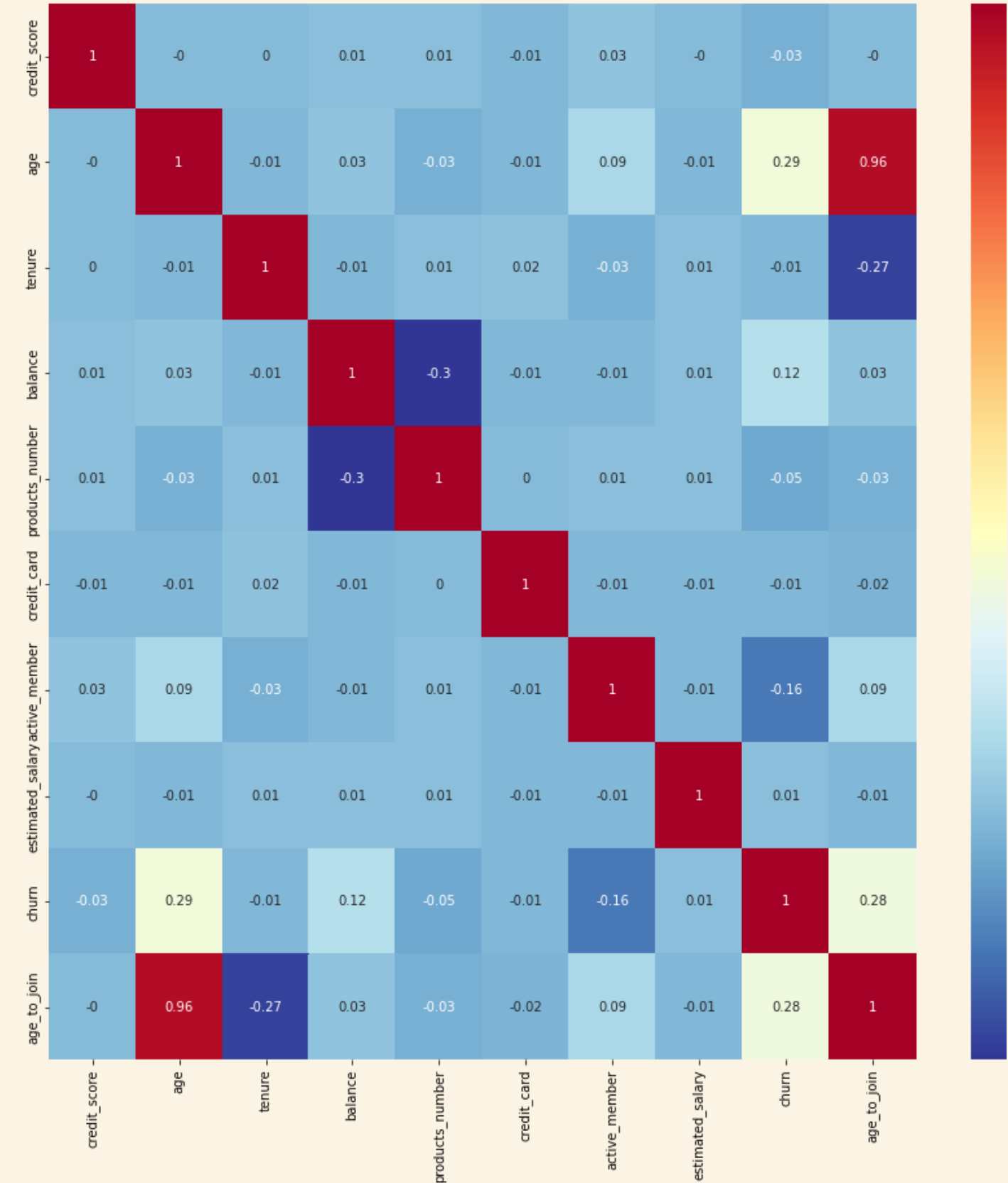
이진 분류 특성 별 고객 이탈



DATA EDA

특성 간 상관관계

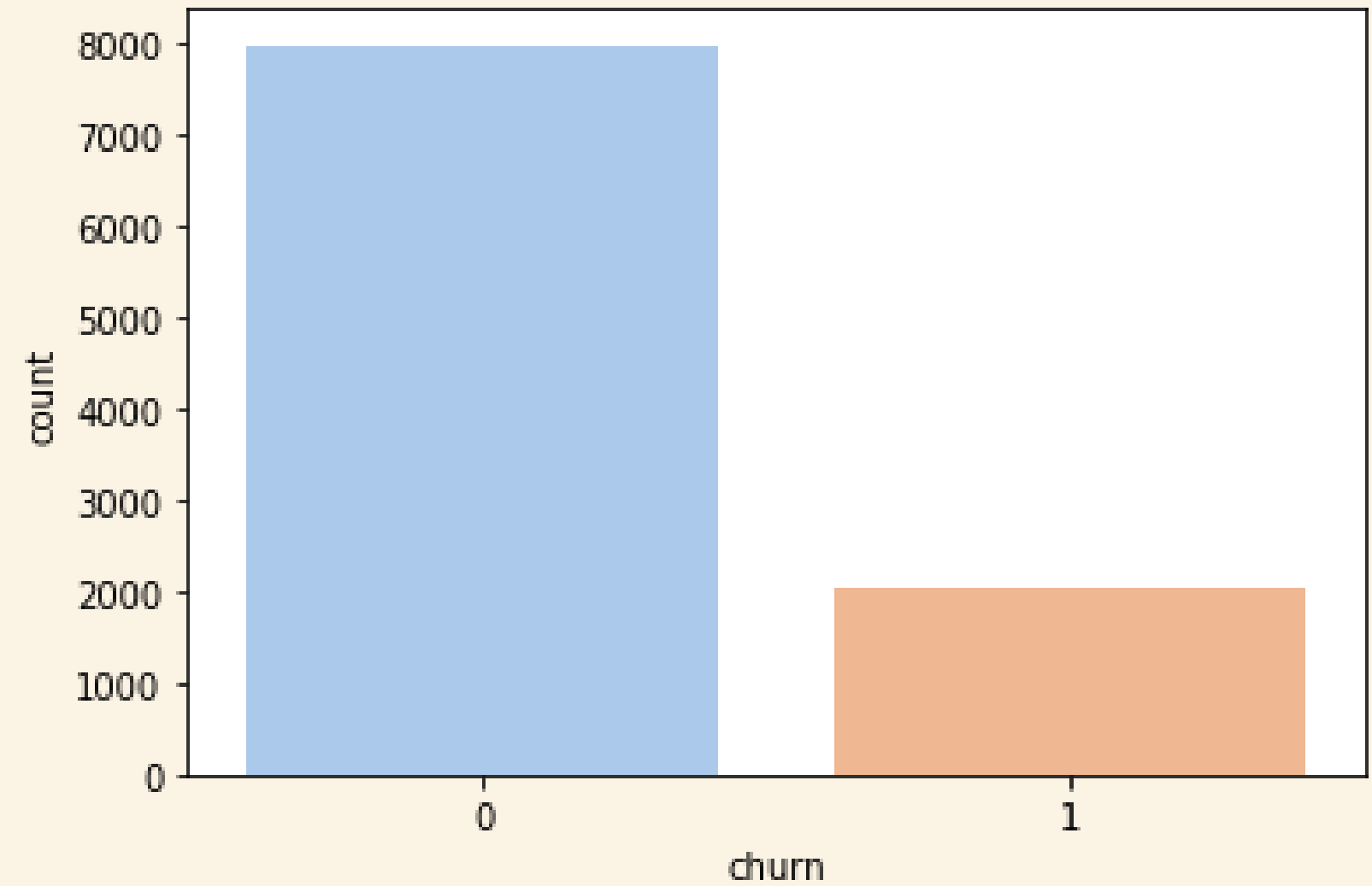
- 특성 별로 영향을 주는 관계인지 확인
- 모델 분석에 전제조건
- 특성 간 특별한 상관관계 x



DATA EDA

타겟 데이터 비율

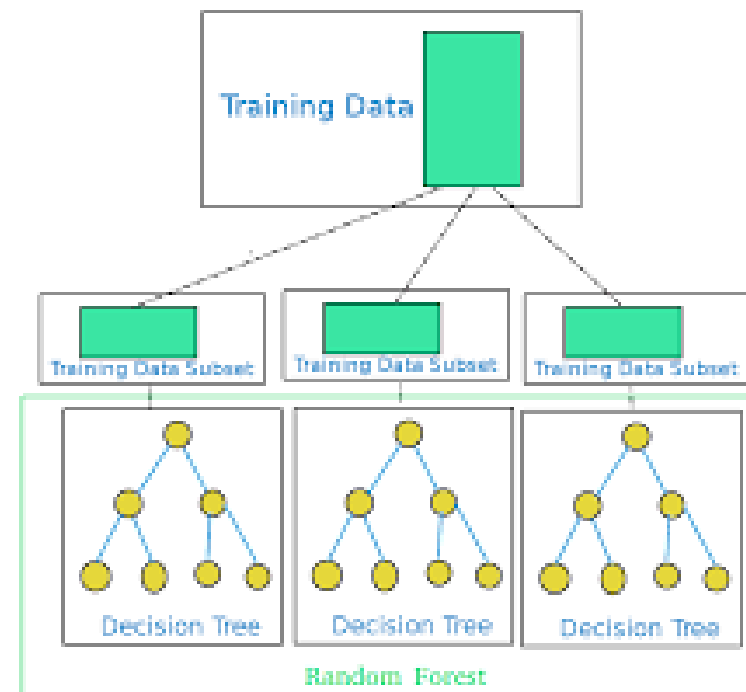
- 8 : 2
- 데이터 불균형 처리 필요



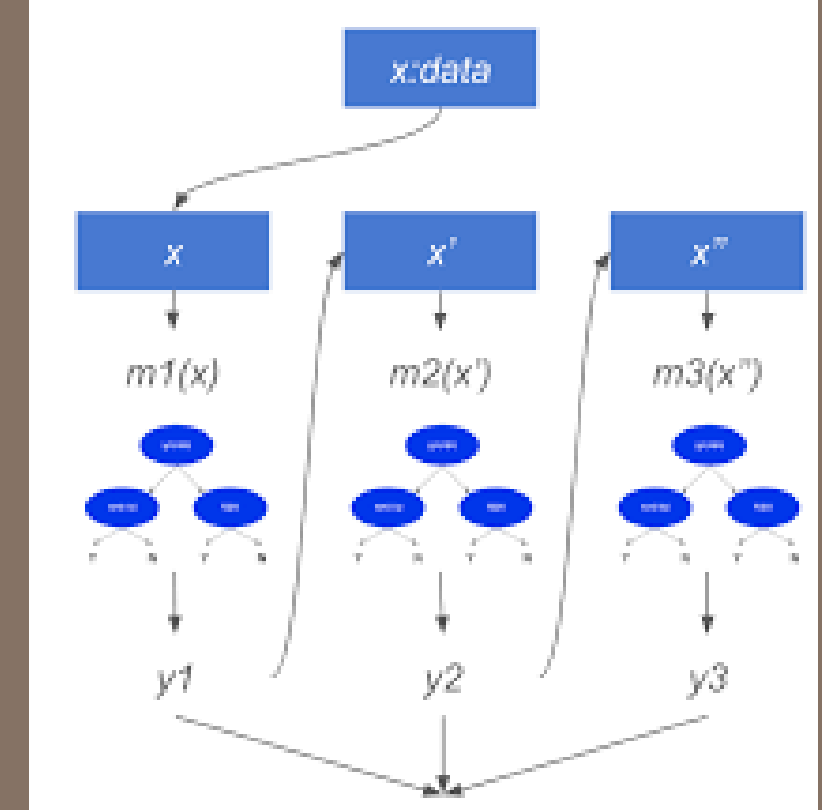
MODELING : 분류문제 모델



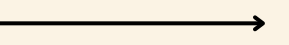
LOSISTIC



RANDOMFOREST

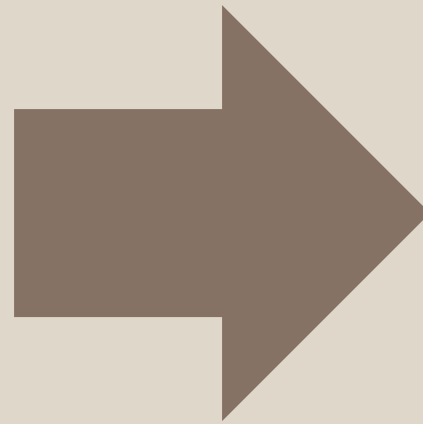
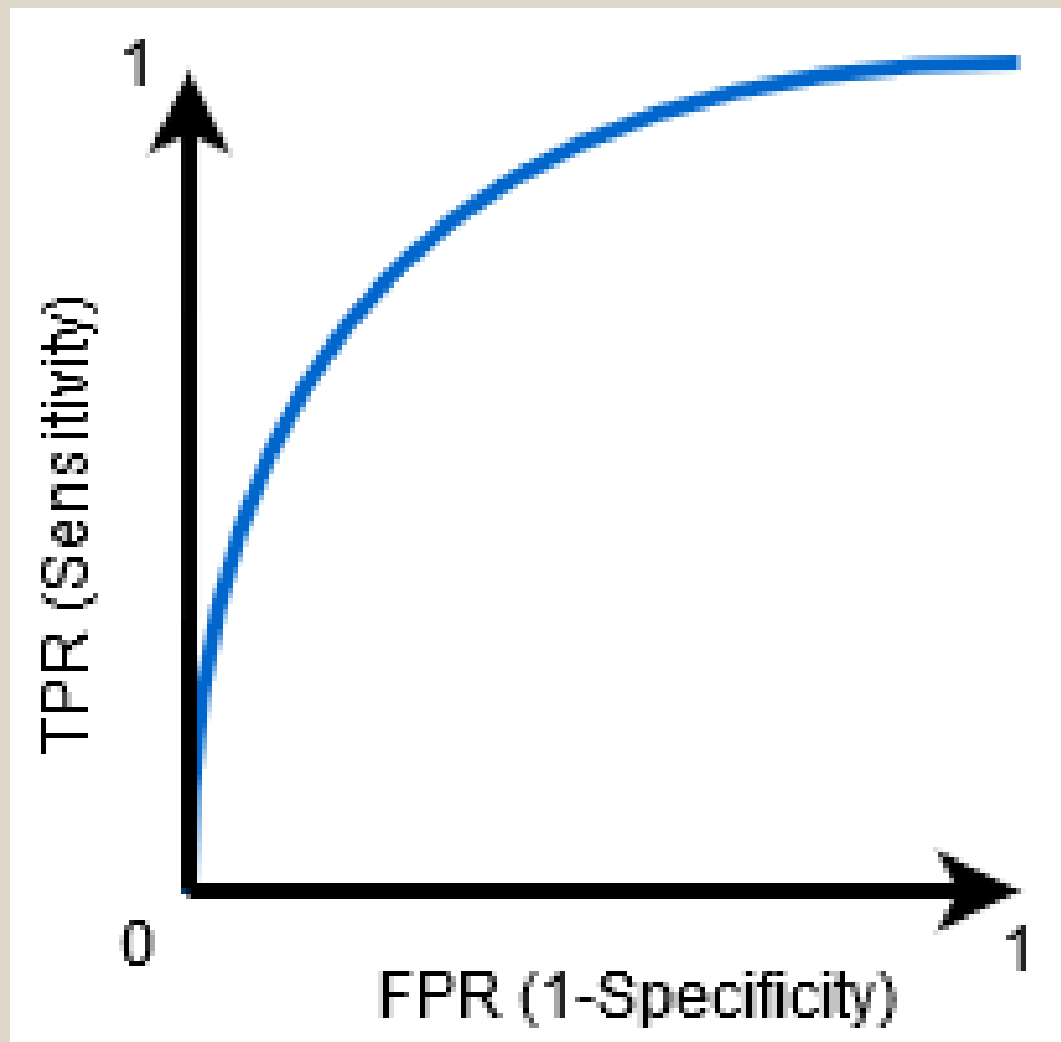


XGBOOST



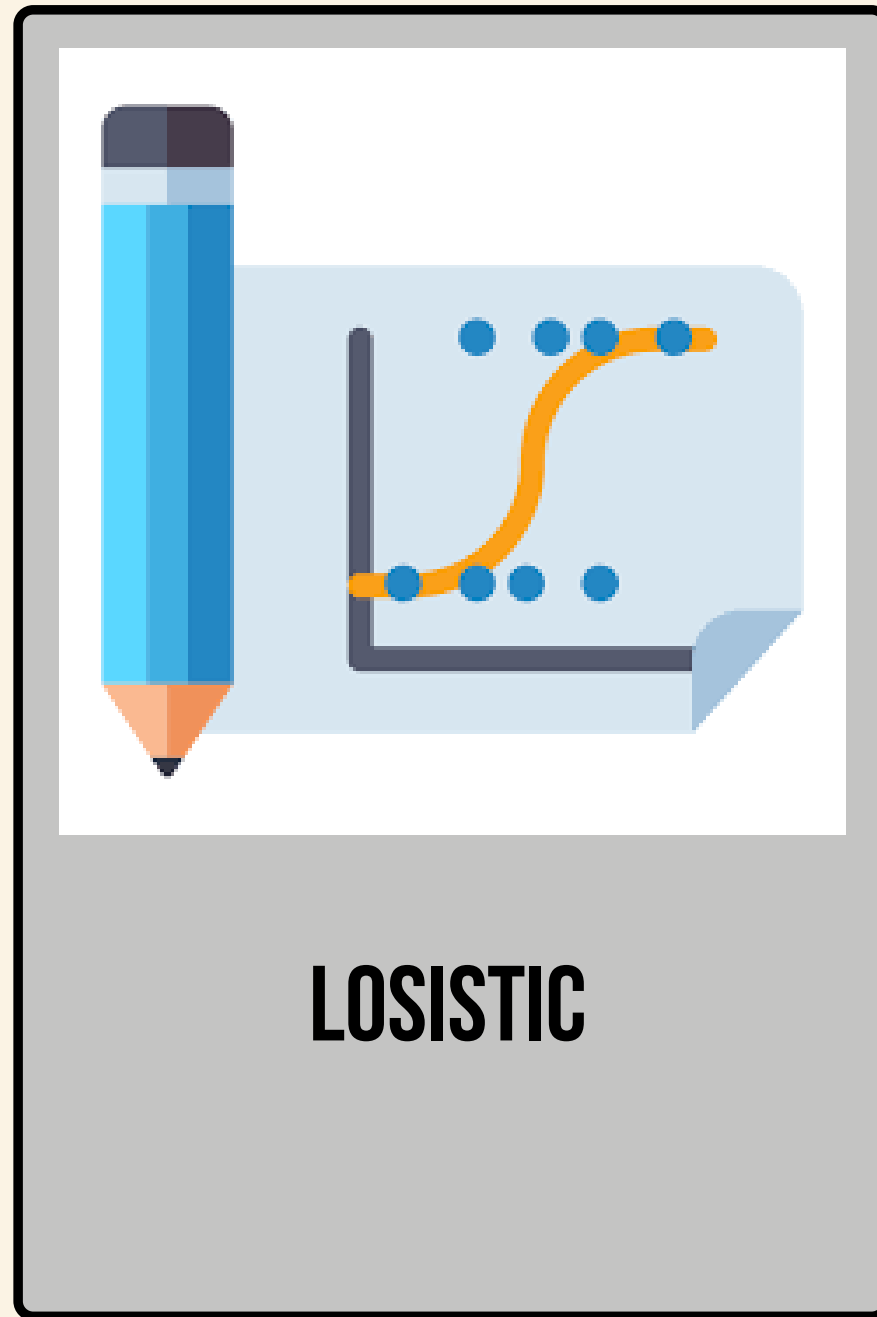
MODELING : 평가지표

AUC SCORE

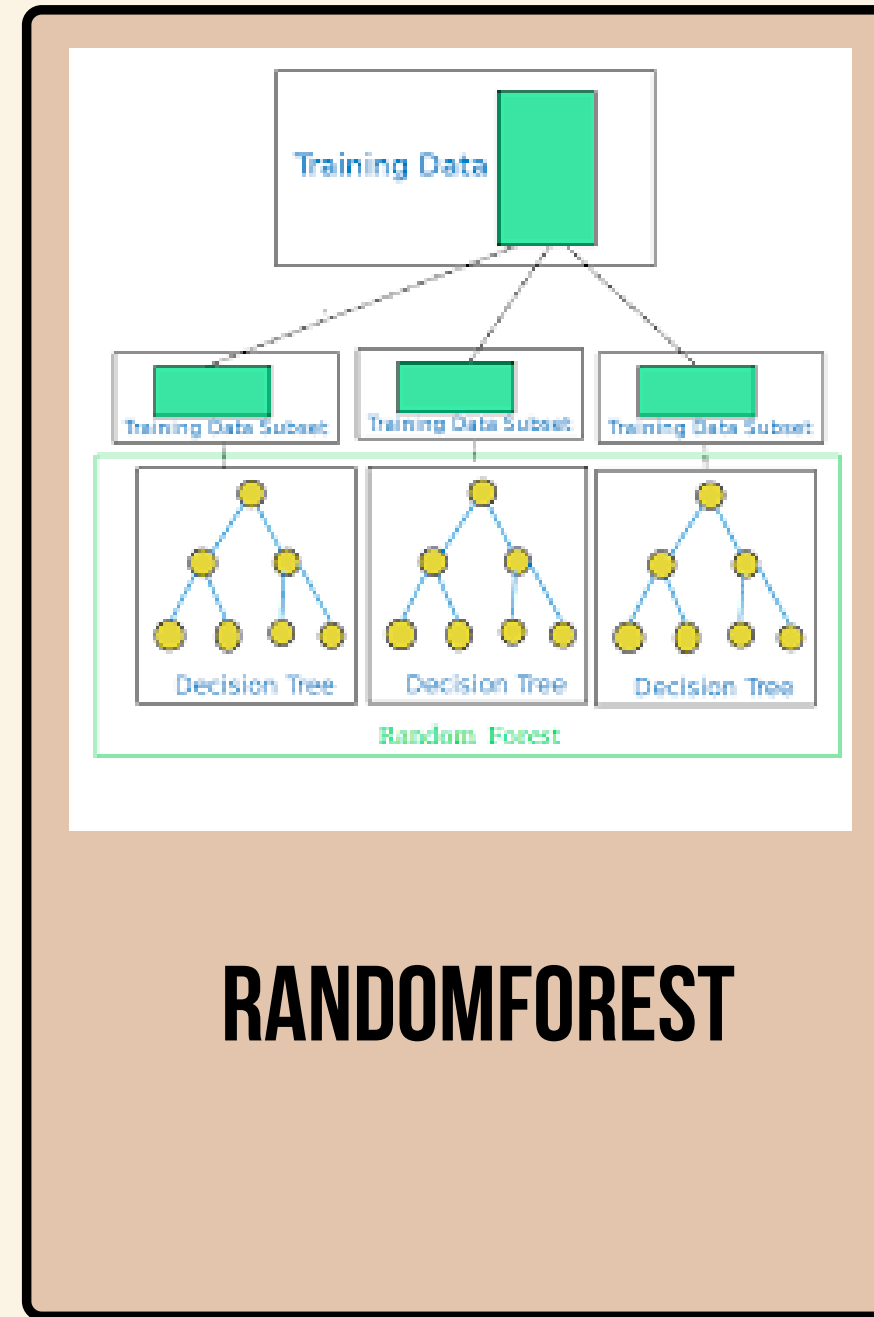


- TPR : 이탈고객을 이탈로 예측
- FPR : 유지고객을 유지로 예측
- TPR \uparrow , FPR \downarrow : 좋은 모델
- AUC : ROC곡선 아래 면적
- 0.8 이상이면 좋은 모델로 예측

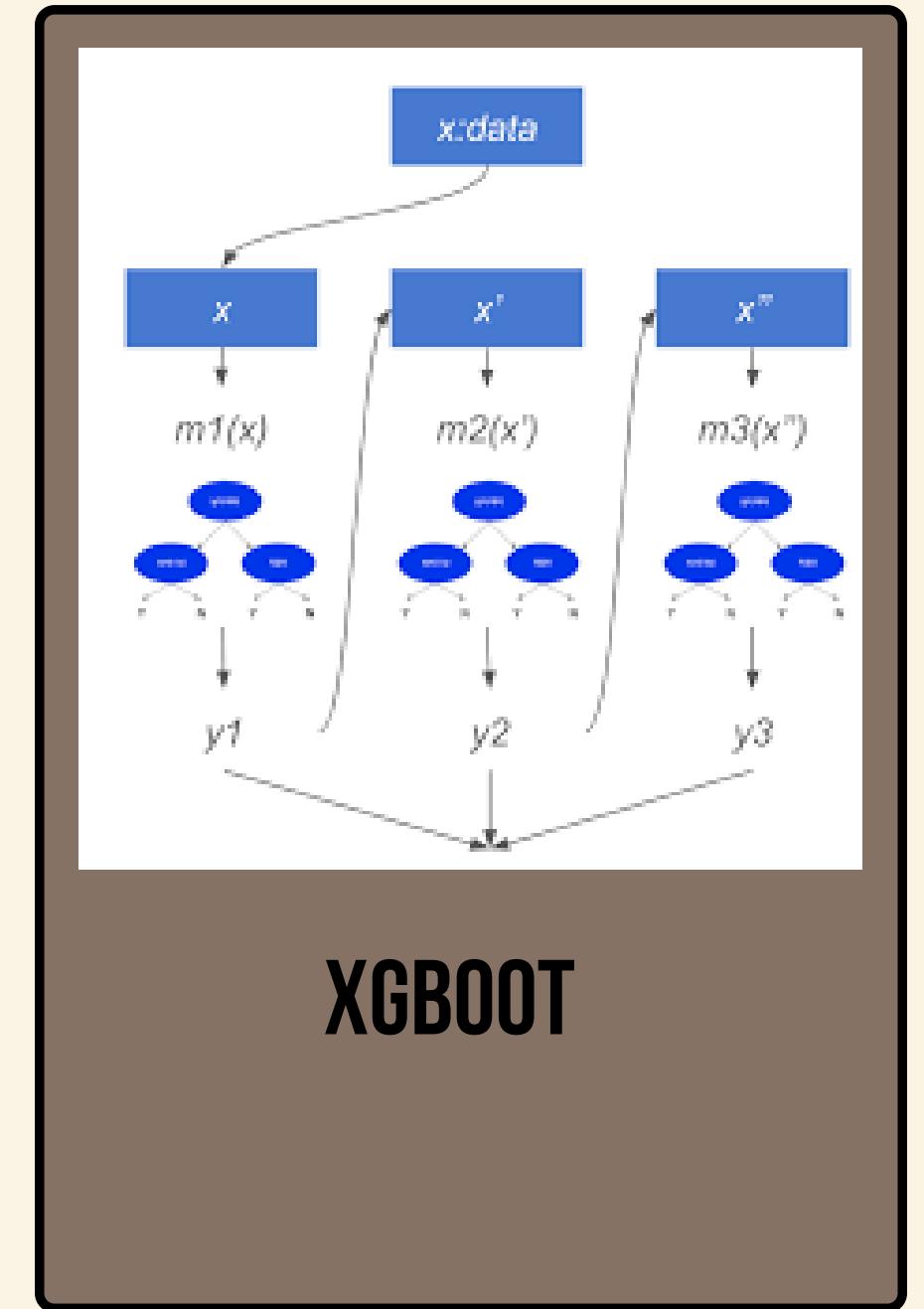
MODELING : AUC 기준으로 선택



VS



VS

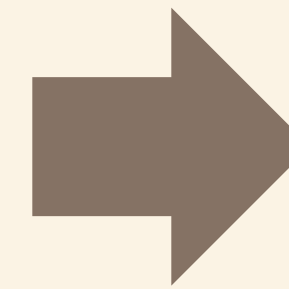


기준모델

- churn 특성 타겟으로 분리
- 학습 / 검증 / 평가 데이터로 분리 8 : 2 비율
- 타겟 데이터의 최빈값 기준으로 기준모델 **AUC** 점수 먼저 계산

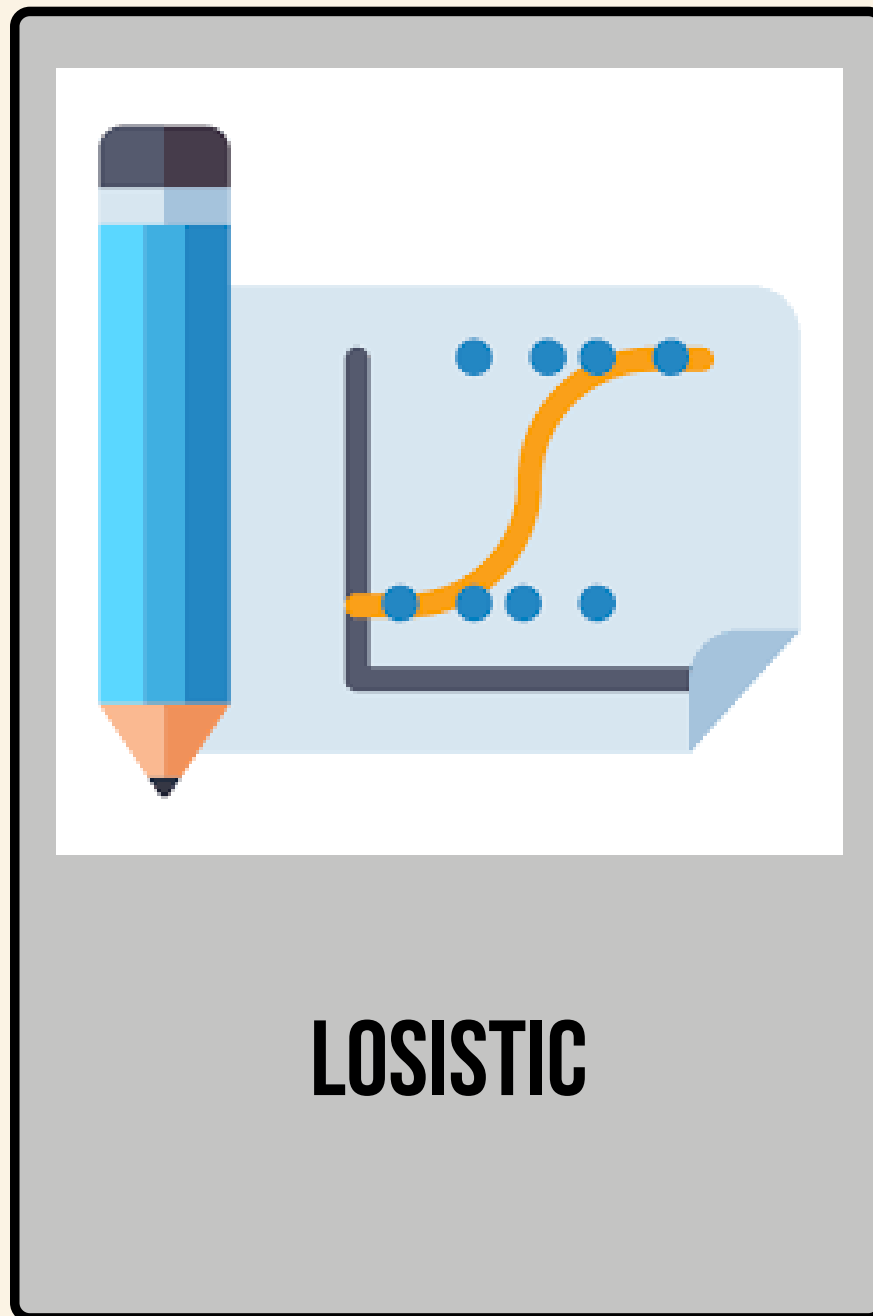
```
1 # 기준모델
2
3 base = y_train.mode()[0]
4 baseline = [base] * len(y_train)
5 auc_baseline = roc_auc_score(y_train, baseline)
6 print(f'기준모델 AUC score : {auc_baseline}')
```

```
기준모델 AUC score : 0.5
```



**기준모델
AUC 점수
0.5**

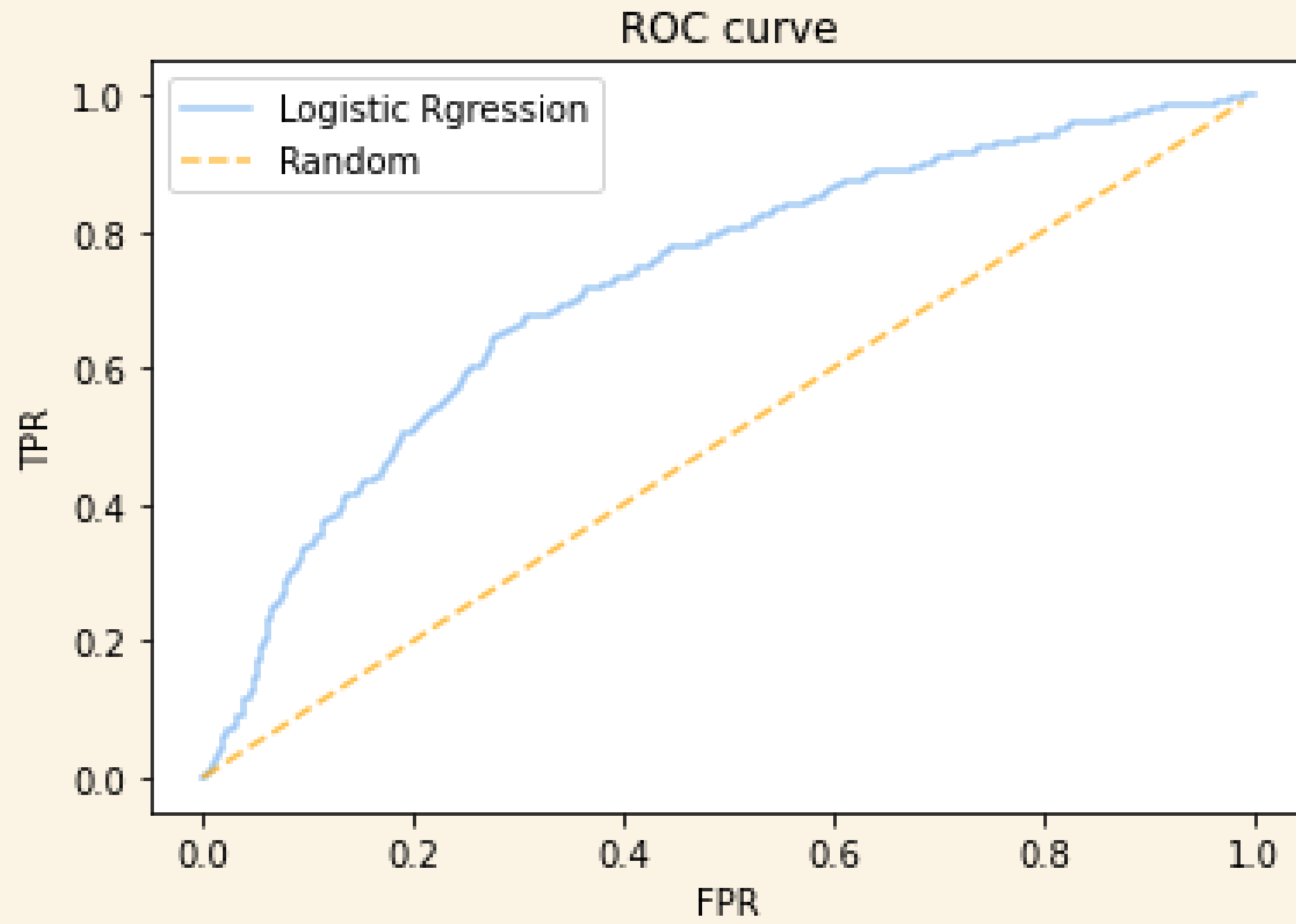
LOGISTIC



```
1 # 학습
2 from sklearn.datasets import make_classification
3 from sklearn.linear_model import LinearRegression, LogisticRegression
4
5 logistic = LogisticRegression(class_weight='balanced')
6
7 logistic.fit(X_train_ode, y_train)
8 y_val_pred = logistic.predict(X_val_ode)
```

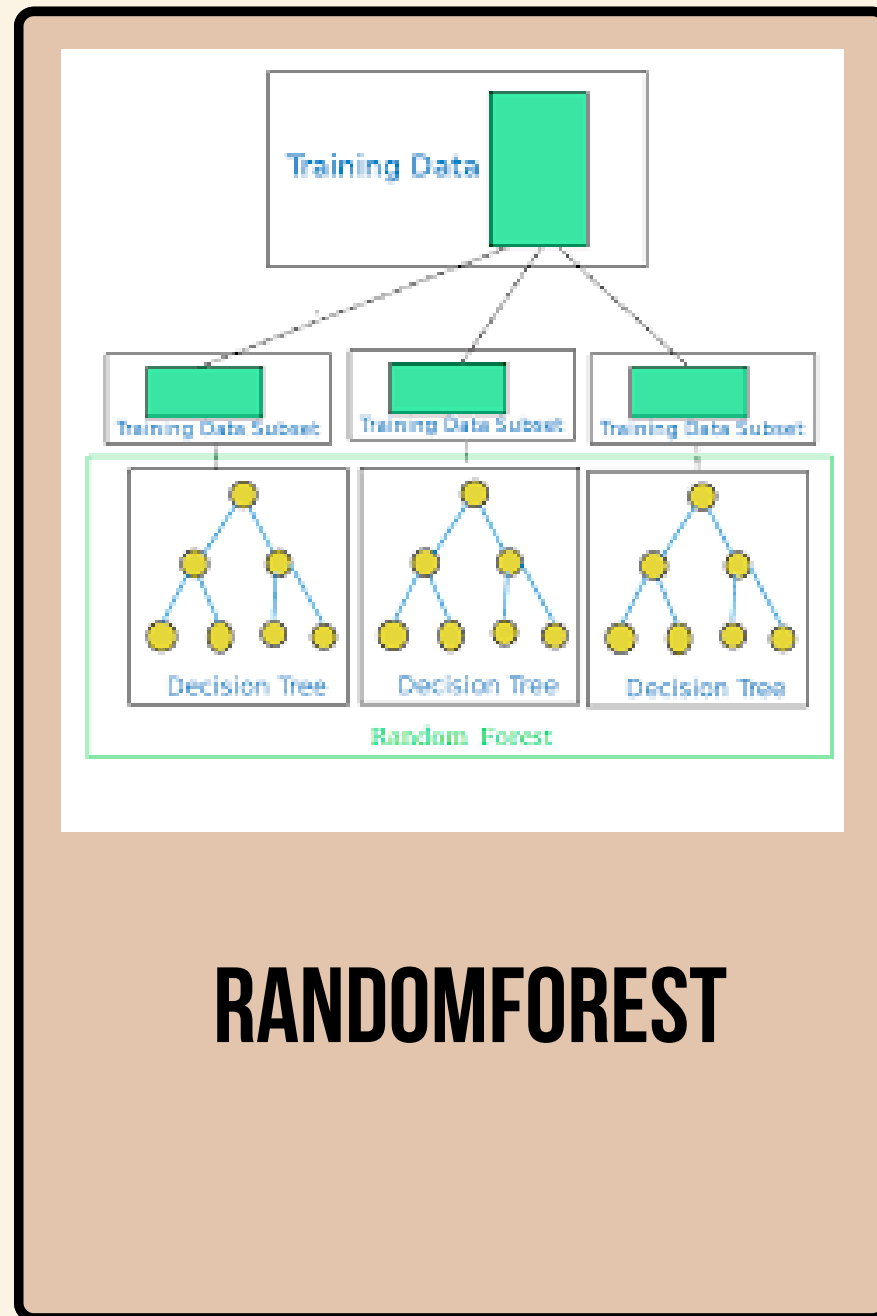
- 타겟 데이터가 불균형 -> class_weight = 'balance' 조정
- 학습 데이터로 학습 검증 데이터로 먼저 성능 확인
- 검증 데이터 AUC 점수 : 0.696

LOGISTIC



- 평가 데이터 ROC curve
- 평가 데이터 **AUC 점수**
: **0.723**

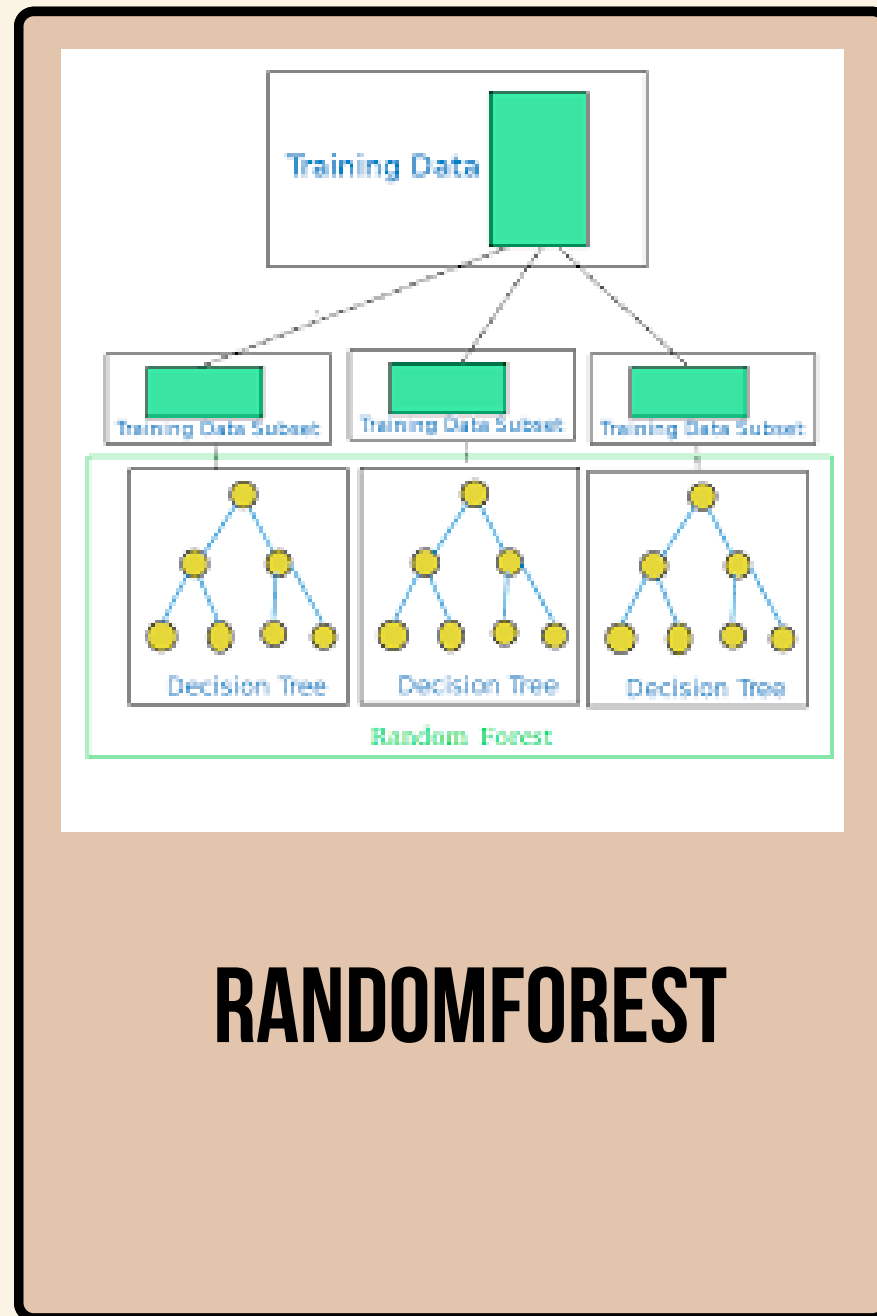
RANDOM FOREST -> RANDOMIZED SEARCH CV



```
3 pipe_rf = make_pipeline(  
4     OrdinalEncoder(),  
5     SimpleImputer(),  
6     RandomForestClassifier(bootstrap=True,  
7                             random_state=42,  
8                             oob_score=True,  
9                             n_jobs=-1,  
0                             class_weight= 'balanced'),  
1 )
```

- 타겟 데이터가 불균형 -> class_weight = 'balance' 조정
- randomized search cv
교차검증을 통해 최적 하이퍼파라메타 설정 -> 과적합 방지

RANDOM FOREST -> RANDOMIZED SEARCH CV

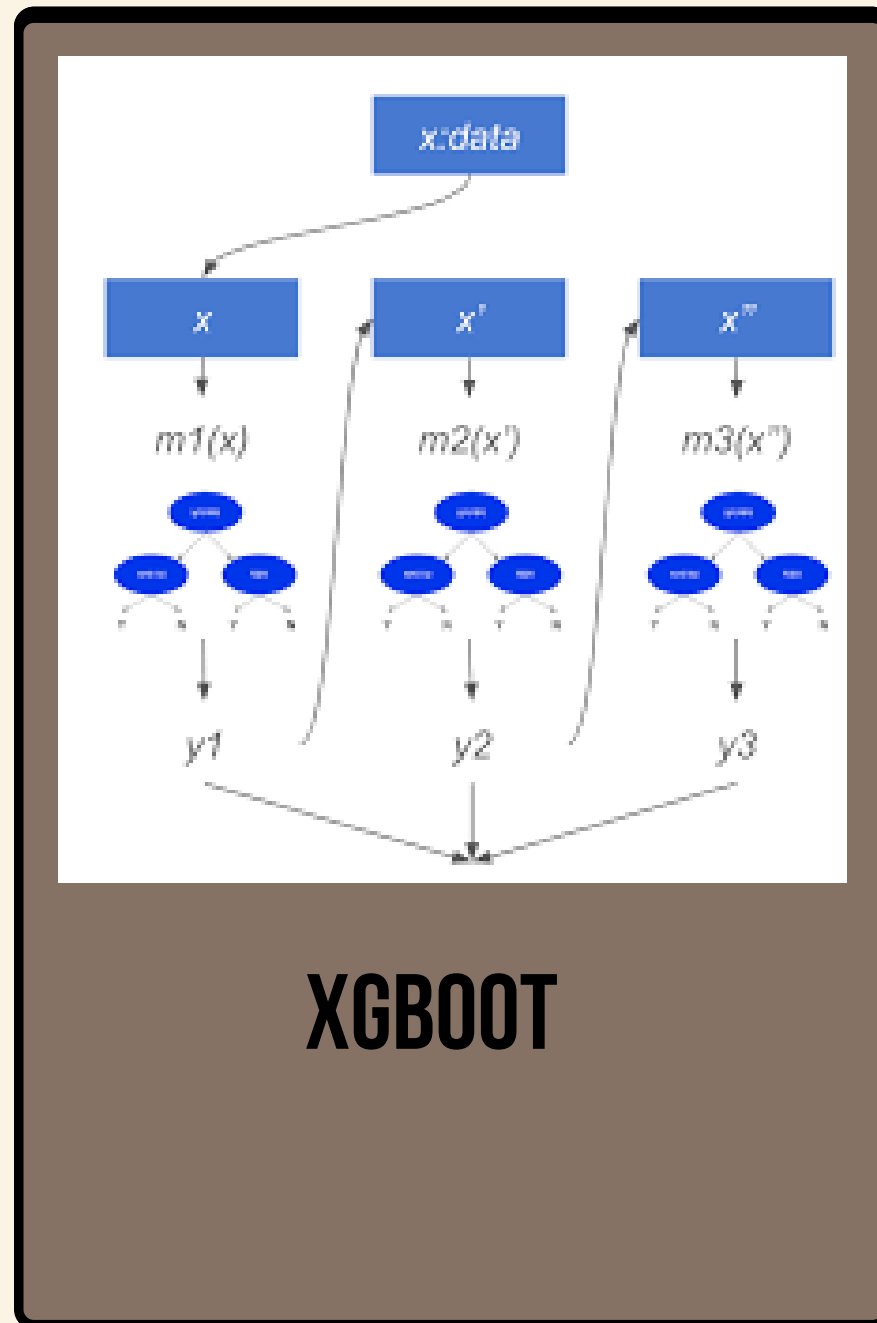


```
1 cv_score = cross_val_score(pipe_rf, X_test, y_test,  
2 |     cv=10, n_jobs=-1, scoring='roc_auc').mean()  
3 print("Random Forest Model Test AUC Score:", round(cv_score, 2))
```

Random Forest Model Test AUC Score: 0.86

- 최적의 파라미터를 적용한 후 재학습
- 검증 데이터 AUC 점수 : 0.85
- 평가 데이터 **AUC 점수 : 0.86**

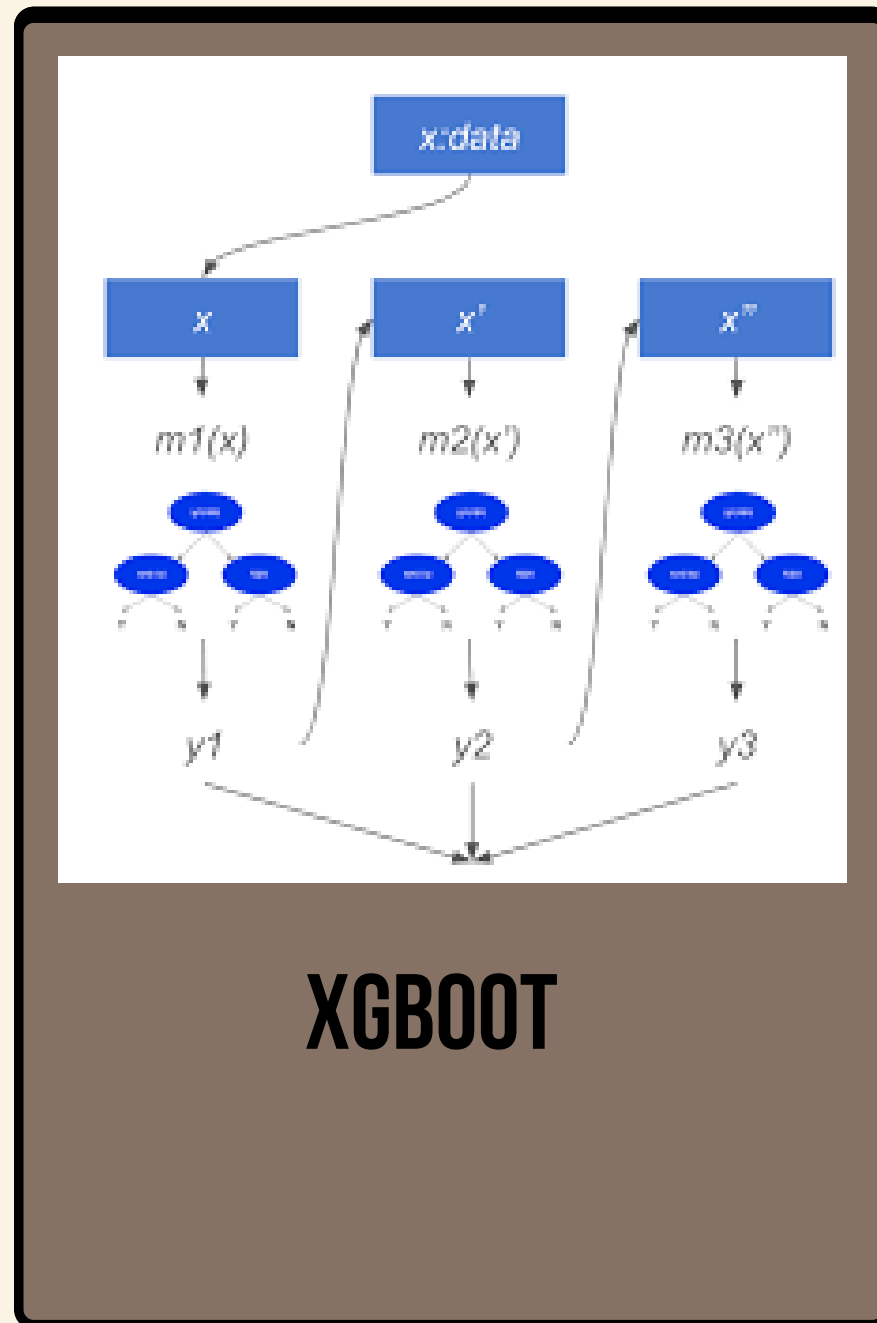
XGBOOST -> RANDOMIZED SEARCH CV



```
1 pipe_xgb = make_pipeline(  
2     OrdinalEncoder(),  
3     SimpleImputer(),  
4     XGBClassifier(  
5         objective="binary:logistic",  
6         eval_metric="auc",  
7         n_estimators=10000,  
8         random_state=42,  
9         n_jobs=-1,  
10        scale_pos_weight=(y_val == 0).sum() / (y_val == 1).sum()),
```

- 타겟 데이터가 불균형 -> scale_pos_weight 조정
- randomized search cv
교차검증을 통해 최적 하이퍼파라메타 설정

XGBOOST -> EARLY STOPPING



```
watchlist = [(X_train_ode, y_train), (X_val_ode, y_val)]

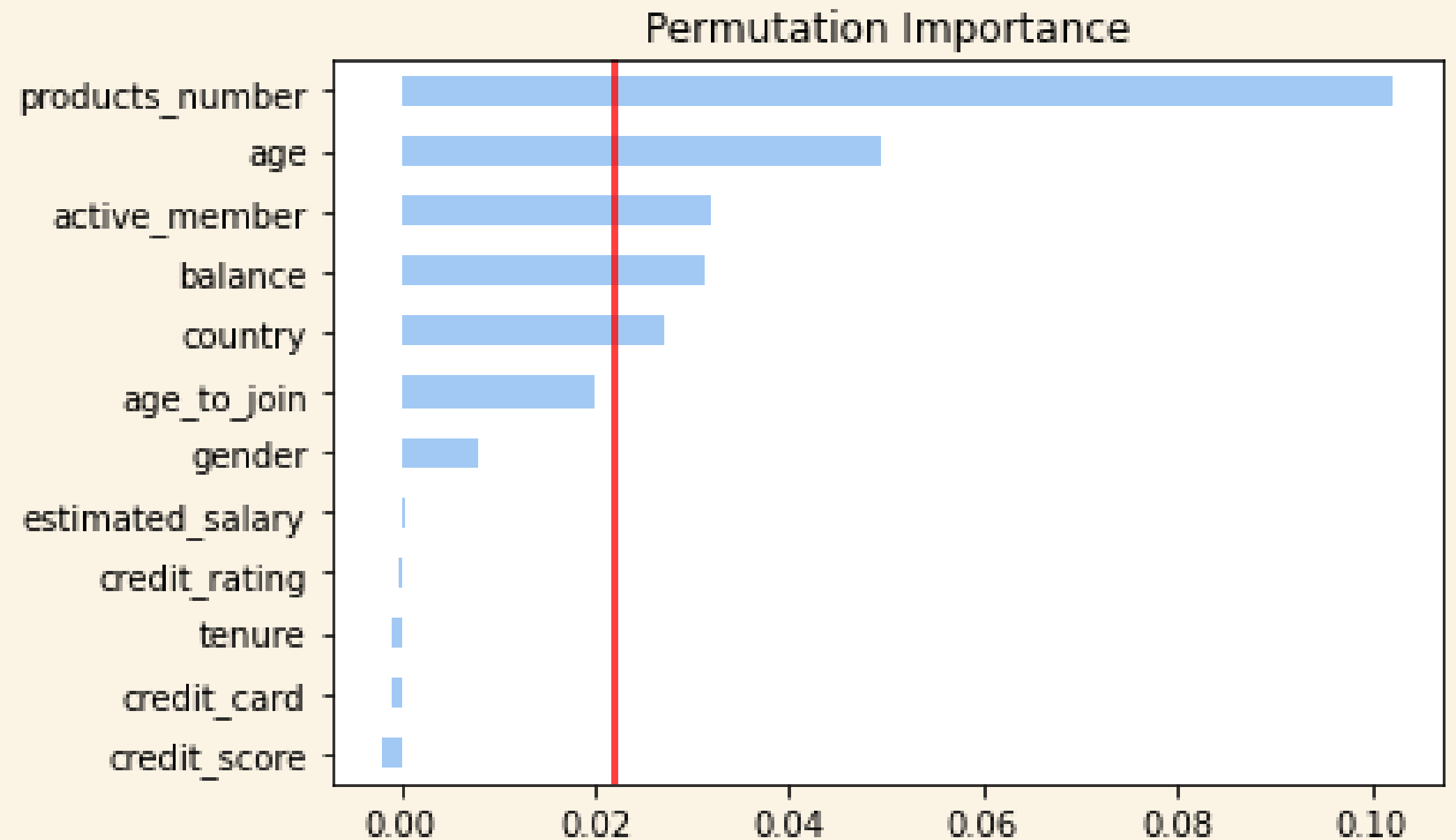
model_xgb.fit(
    X_train_ode,
    y_train,
    eval_set=watchlist,
    early_stopping_rounds=50,
```

- early stopping 을 통해 성능이 더 이상 증가하지 않을 때 학습을 중지 -> 과적합 방지하면서 정규화
- 평가 데이터 **AUC 점수 : 0.85**

특성중요도 : PERMUTATION IMPORTANCE

PERMUTATION IMPORTANCE

- 각 특성을 무작위 배열
- 결과비교
- 특성 중요도 계산
- XGBOOST 모델

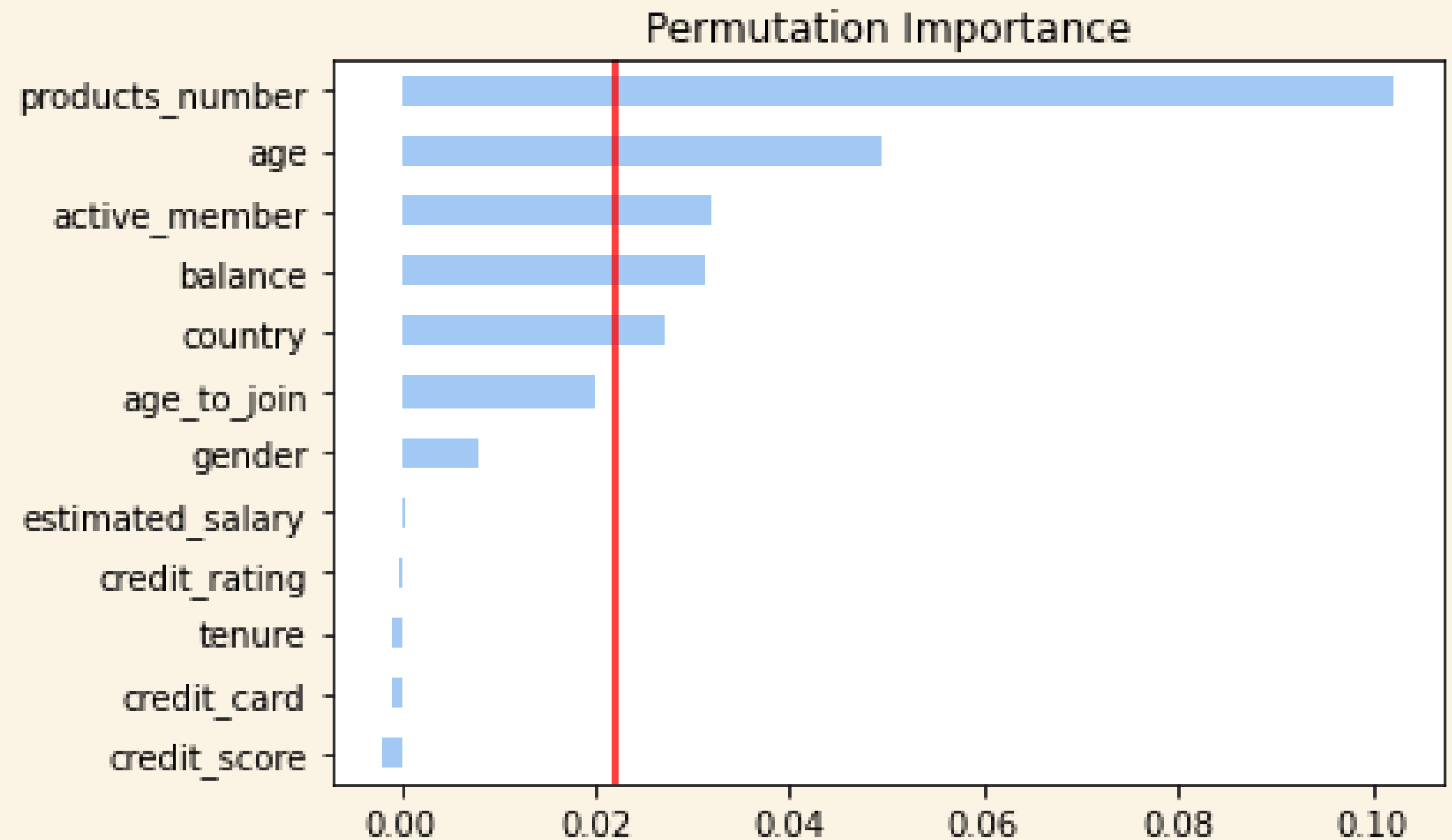


특성중요도 : PERMUTATION IMPORTANCE

PERMUTATION IMPORTANCE

- 각 특성을 무작위 배열
- 결과비교
- 특성 중요도 계산
- XGBOOST 모델

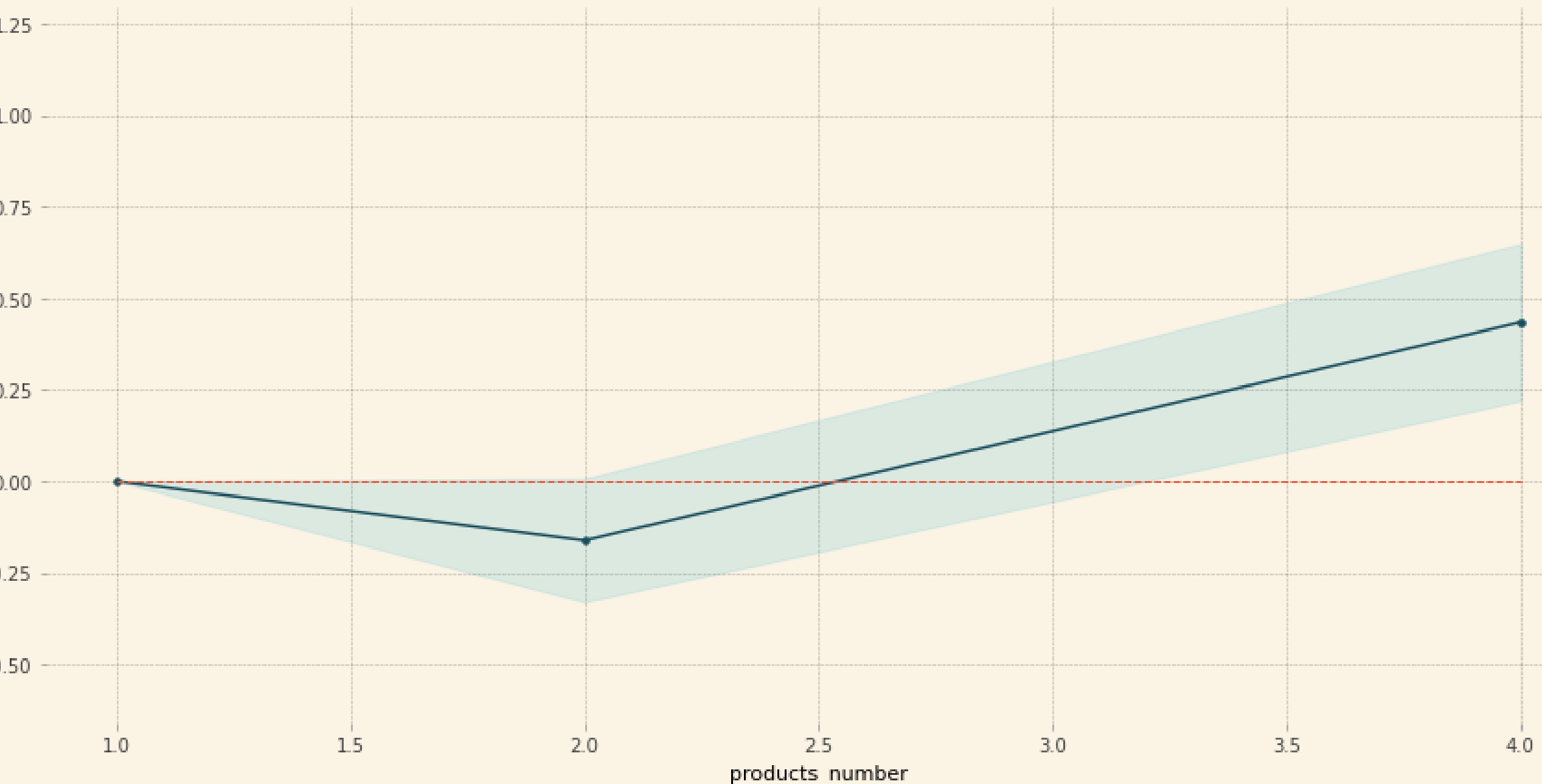
- products_number
- age
- active_member



특성중요도 : PERMUTATION IMPORTANCE

PDP for feature "products_number"

Number of unique grid points: 3



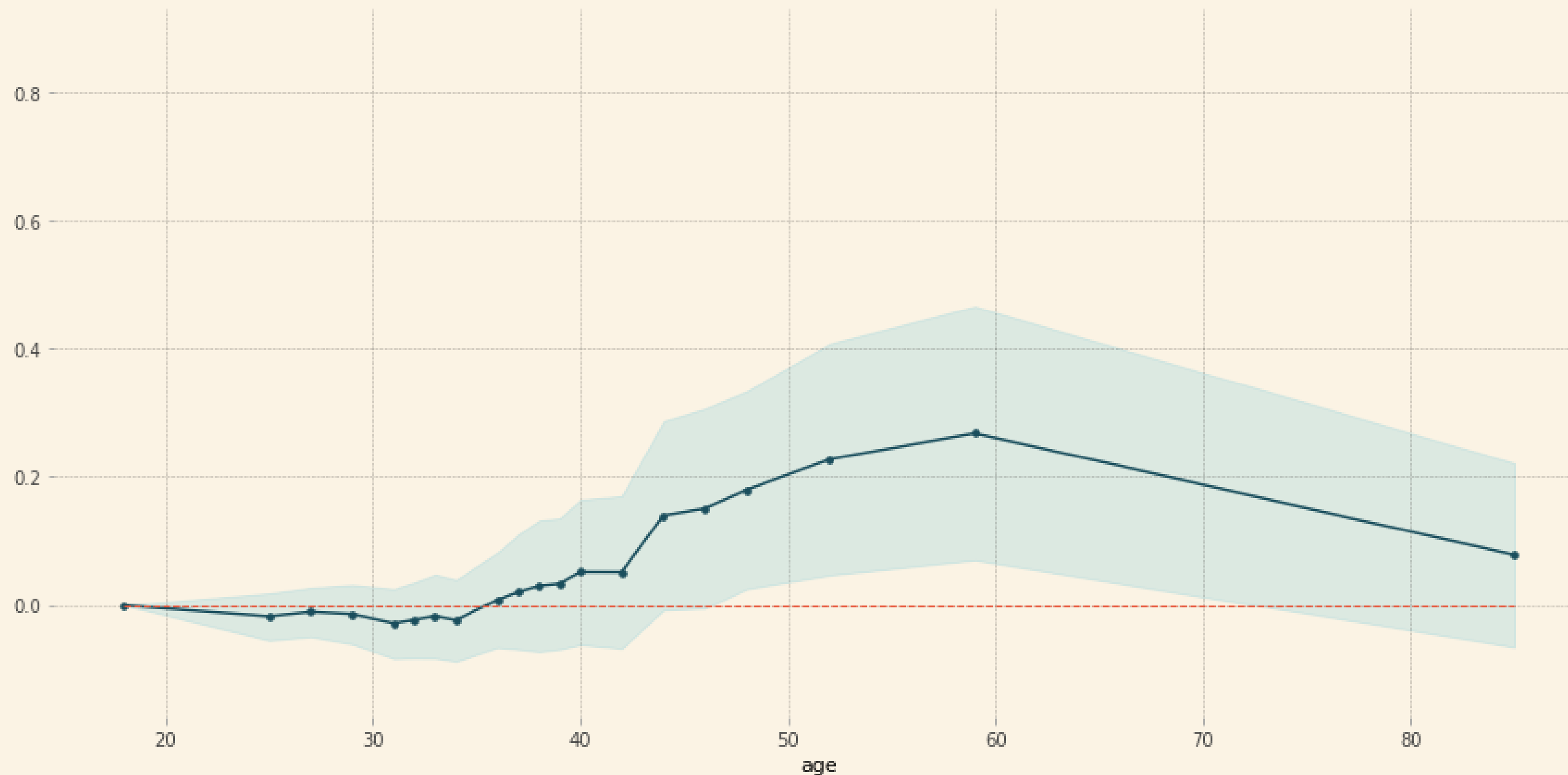
• 2 상품 계약고객 -> 이탈율 ↓

• 3,4 상품 계약고객 -> 이탈율 ↑

특성중요도 : PERMUTATION IMPORTANCE

PDP for feature "age"

Number of unique grid points: 20

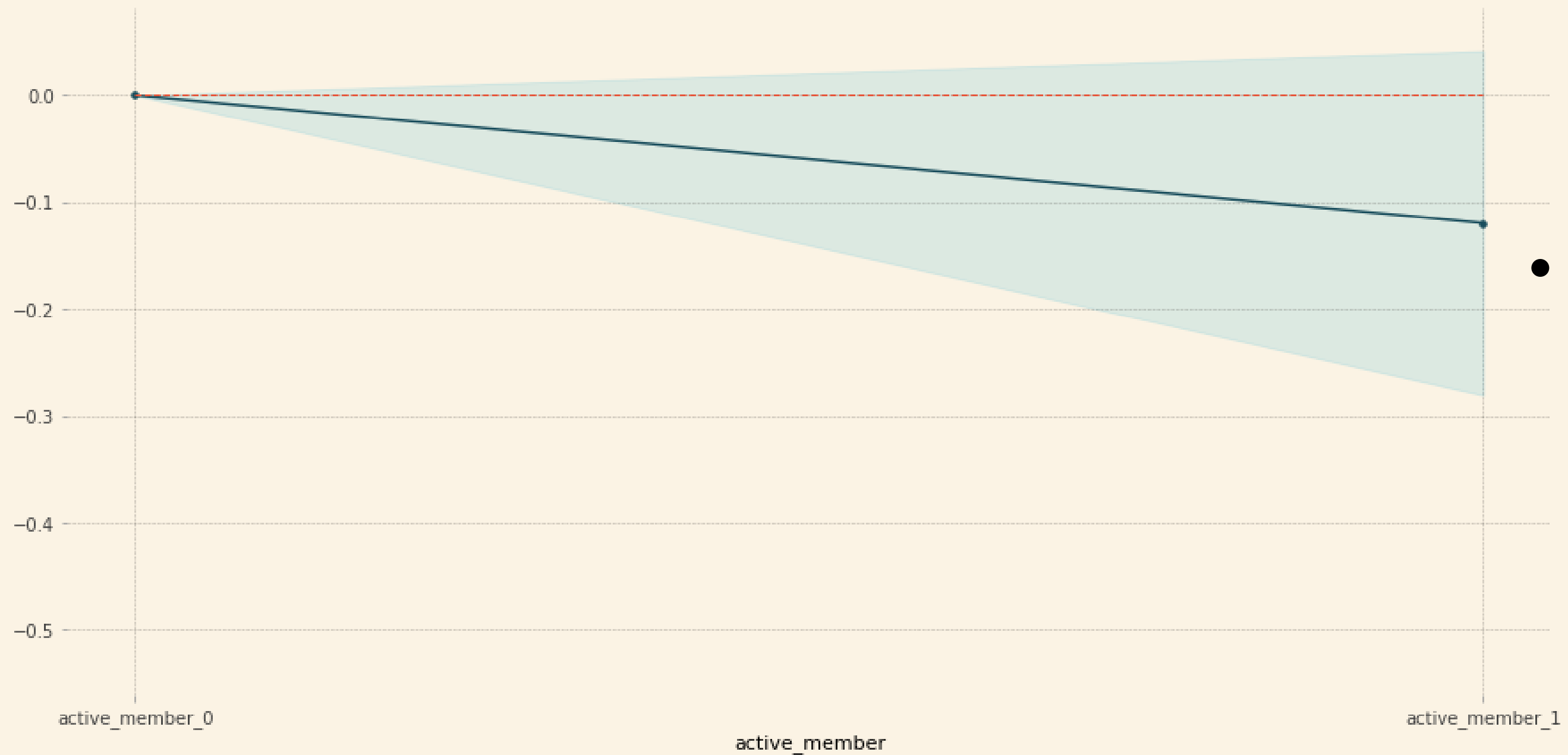


• 40대 중반 이후부터 이탈율 ↑

특성중요도 : PERMUTATION IMPORTANCE

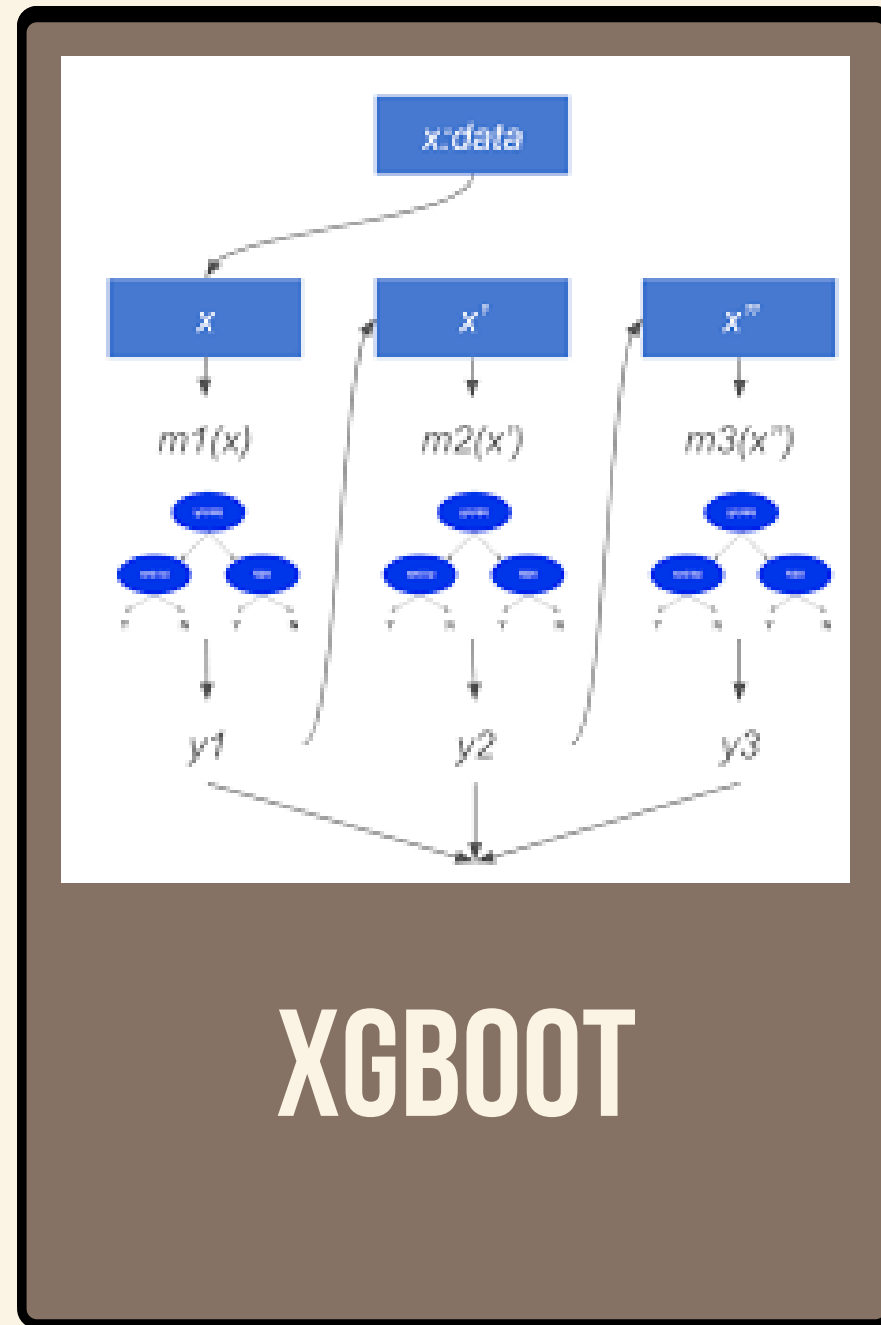
PDP for feature "active_member"

Number of unique grid points: 2



- 정회원 가입 고객→ 이탈율 ↓

해석 및 결론



[계약 상품 종류]

- 2 상품 계약고객 -> 이탈율 ↓
- 3,4 상품 계약고객 -> 이탈율 ↑

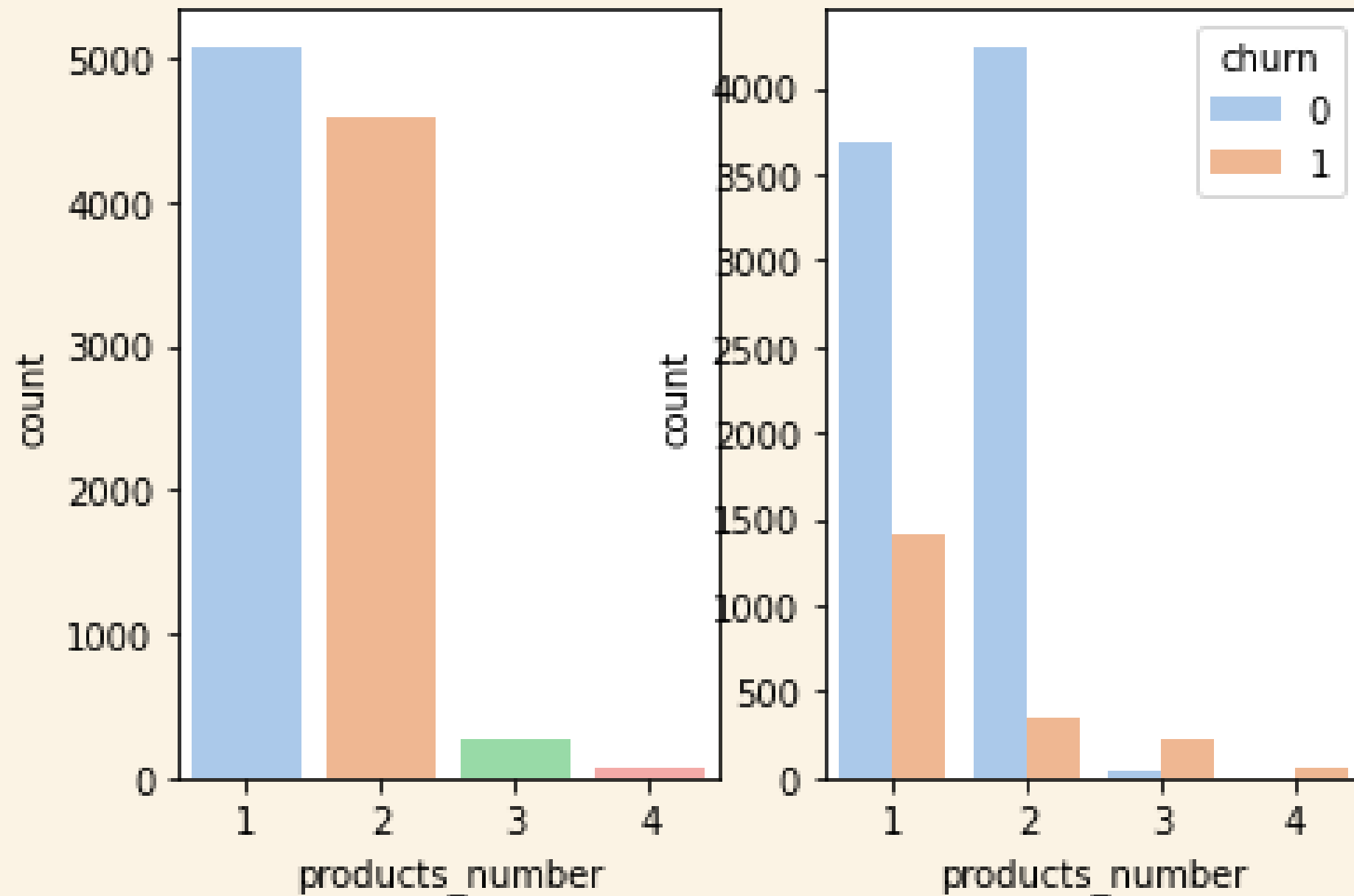
[고객 나이]

- 40대 중반 이후부터 이탈율 ↑

[정회원 여부]

- 정회원 가입 고객-> 이탈율 ↓

해석 및 결론



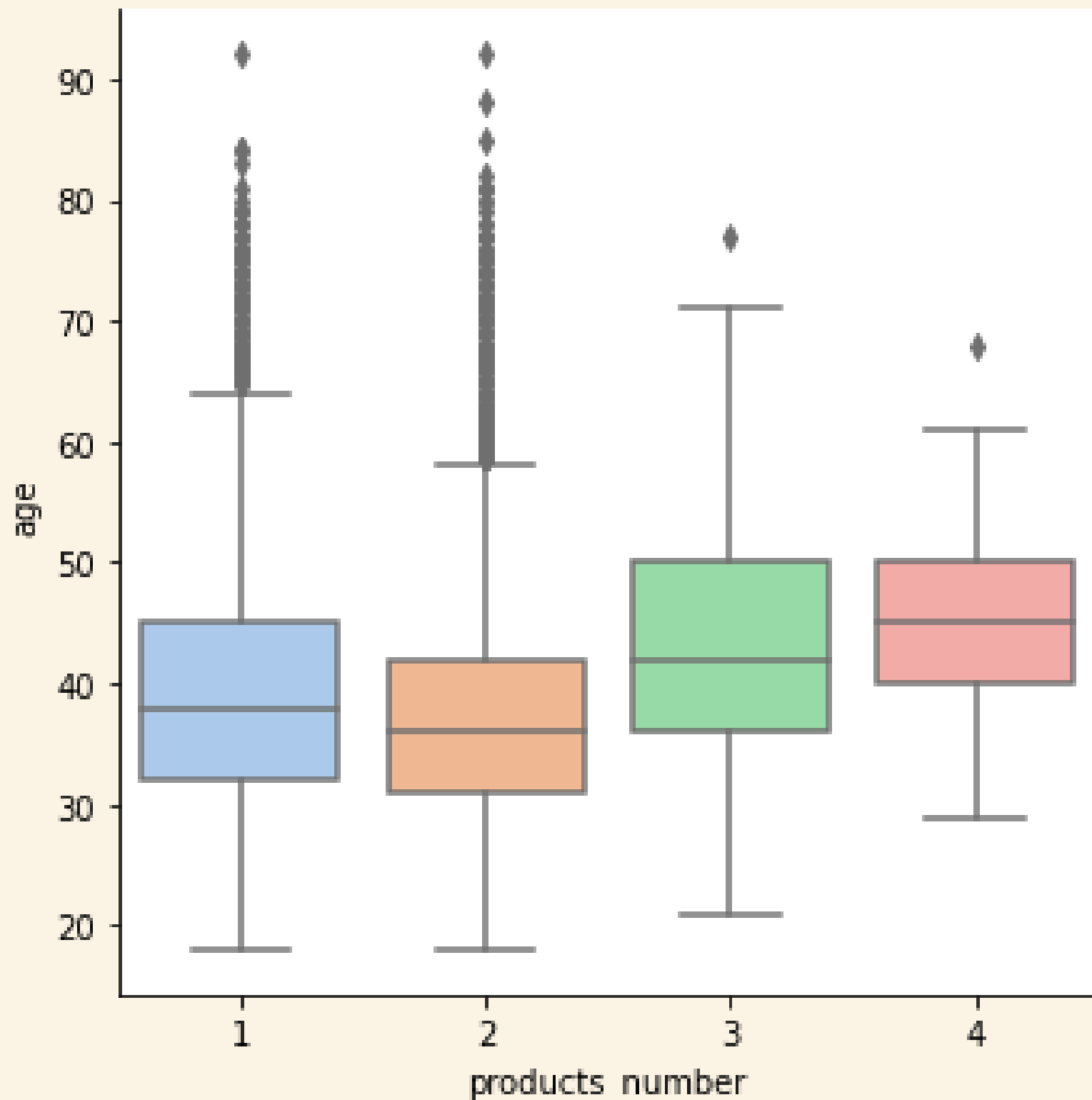
[계약 상품 종류]

- 1, 2 상품은 인기와 유지비율이 준수
- 3, 4 상품은 가입률이 적고 이탈 고객이 유지비율 보다 높으므로 고객만족도가 낮음
- 3, 4 상품 폐기 -> 유지비 절약

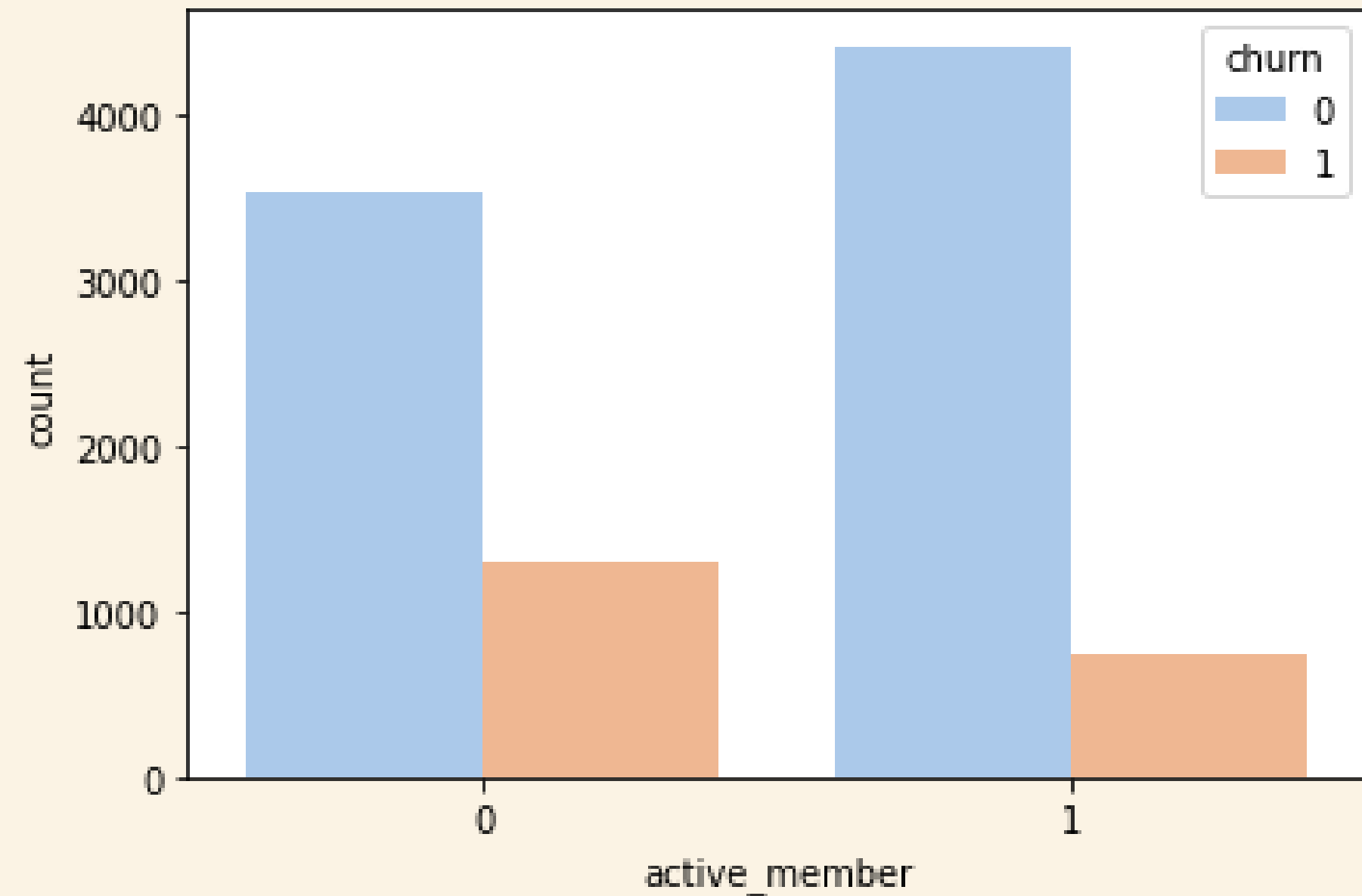
해석 및 결론

[계약 상품 종류, 나이]

- 3, 4 상품의 가입 고객의 나이가 **4,50대 중장년층**
- 중장년층 대상의 새로운 상품 개발이 필요



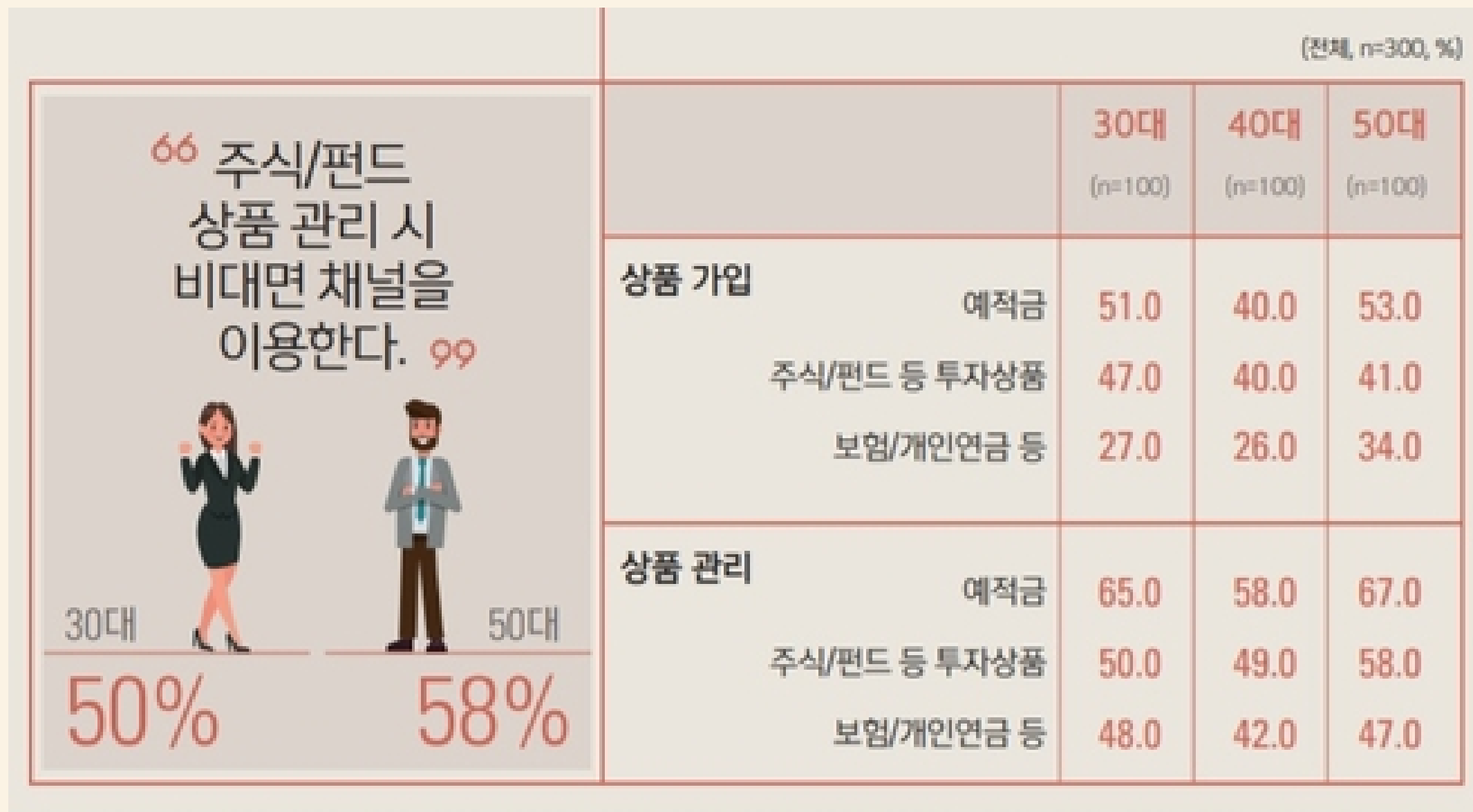
해석 및 결론



[정회원 여부]

- 고객의 정회원 가입을 유도할 필요

해석 및 결론



[중장년층도 쓰는 모바일 금융]

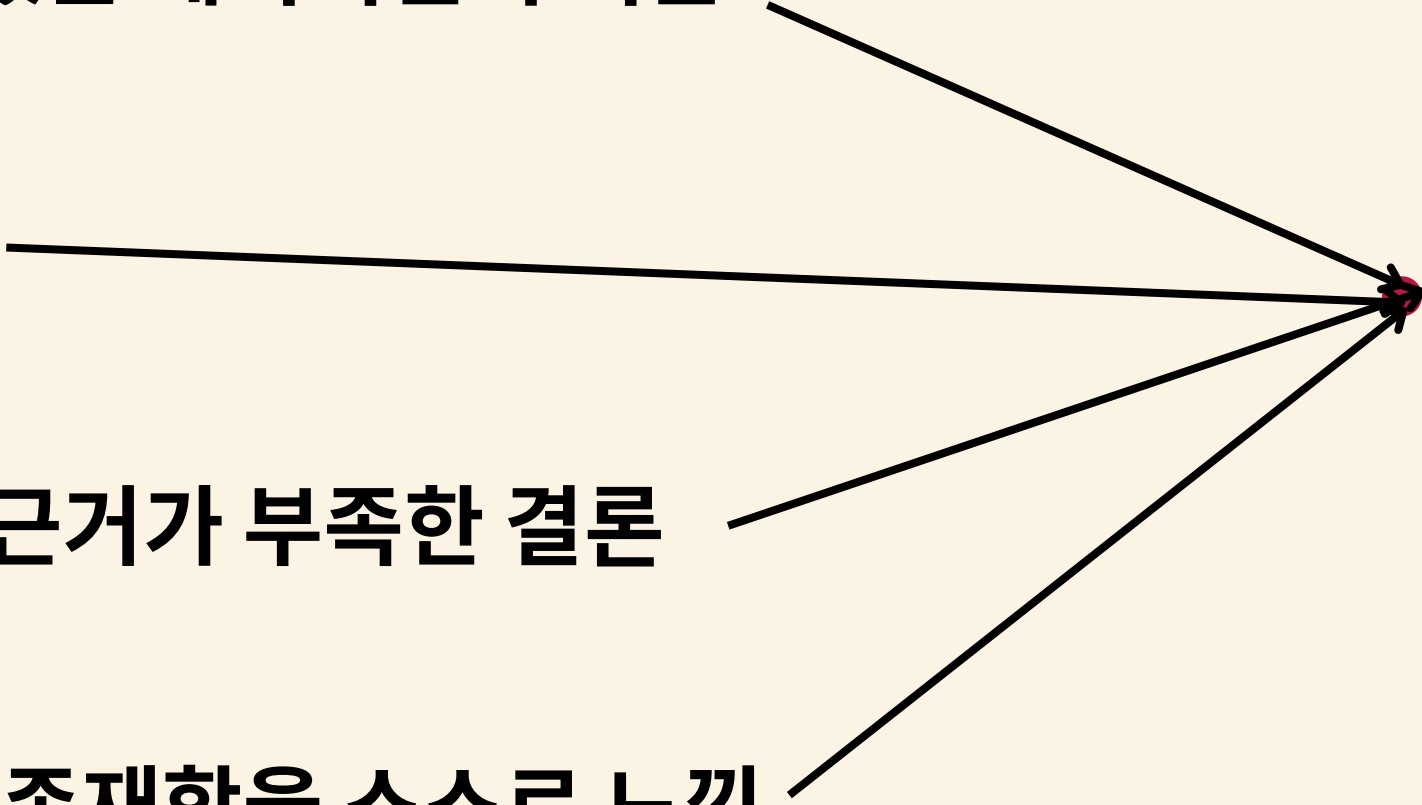
- 50대 이상 고객의 모바일 금융 사용 ↑
- 금융 정보를 모바일로 획득
- 3, 4 보완 상품 개발
- 50대 이상, 정회원이 아닌 고객에게 앱 알리를 통해 정회원 가입 시 혜택이 주어지는 새로운 상품 알리 발송

한계점

- 신뢰성있는 데이터인지 의문
- 모델링
- 확실한 근거가 부족한 결론
- 등 다수 존재함을 스스로 느낌

한계점

- 신뢰성있는 데이터인지 의문
- 모델링
- 확실한 근거가 부족한 결론
- 등 다수 존재함을 스스로 느낌



손철살인 피드백
부탁드리겠습니다.

THANK YOU :)

AIB_15_박준영
