

## 프로젝트 2

# E-commerce 기업 고객 이탈 예측 및 예방 전략

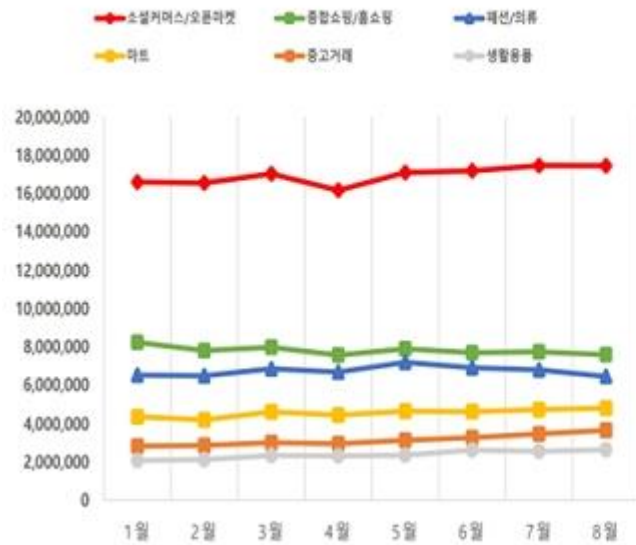
발표자 : 이새벽

Part 1

목 적

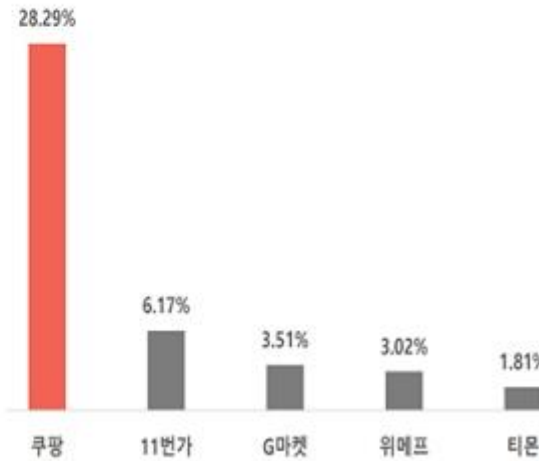
## < 목 적 >

[주요 쇼핑앱 카테고리 MAU 추이]



[주요 소셜커머스/오픈마켓 앱 8월 단독 사용률 비교]

\*단독 사용률: 전체 소셜커머스/오픈마켓 카테고리 내 해당앱 단독 사용 비율



igaworks MOBILEINDEX

[주요 소셜커머스/오픈마켓 앱 월별 이탈률 현황]

\*이탈자: 지난월 해당앱 사용자 중 해당월 미사용자



- 이커머스 시장 급격히 확대되면서 끊임없이 새로운 커머스 앱 생겨남
- 쿠팡 : 쇼핑 앱 사용률 1위, 이탈하지 않는 사용자도 1위
- 다른 기업 이탈률을 보면 최대 40%로 매우 높음
- 우리 회사는 고객 이탈률을 감소시키기 위해 어떤 노력을 해야하는가?

## < 목 적 >



☆ 목적 : 회사 이탈 예정 고객을 놓치지 않기 위한 전략 수립 ☆

1. 데이터 시각화
2. 전처리, 모델링

Part 2

데 이 터

## < 데이터 >

Data	
1. Customer ID	Unique customer ID (고객 ID)
2. Churn	Churn Flag (이탈 고객) / ☆ target
3. Tenure	Tenure of customer in organization (고객의 가입기간)
4. PreferredLoginDevice	Preferred login device of customer (고객이 선호하는 로그인)
5. CityTier	City tier (도시 등급)
6. WarehouseToHome	Distance in between warehouse to home of customer (창고에서 고객 집까지의 거리)
7. PreferredPaymentMode	Preferred payment method of customer (고객이 선호하는 결제수단)
8. Gender	Gender of customer (고객의 성별)
9. HourSpendOnApp	Number of hours spend on mobile application or website (모바일 앱 또는 웹사이트에서 보낸 시간)
10. NumberOfDeviceRegistered	Total number of deceives is registered on particular customer (등록된 장치 수)

- 이커머스 회사 데이터
- 5630 rows, 20 columns 구성
- 타겟 : 이탈 고객 (이탈 X - 0, 이탈 O - 1)

## < 데이터 >

Data	
11. PreferredOrderCat	Preferred order category of customer in last month (선호 주문 카테고리)
12. SatisfactionScore	Satisfactory score of customer on service (서비스 만족도 점수)
13. MaritalStatus	Marital status of customer (고객의 결혼 여부)
14. NumberOfAddress	Total number of added address on particular customer (주소 수)
15. Complain	Any complaint has been raised in last month (지난달 컴플레인)
16. OrderAmountHikeFromlastYear	Percentage increases in order from last year (전 년도 대비 주문량 인상 퍼센트)
17. CouponUsed	Total number of coupon has been used in last month (지난 달에 사용한 총 쿠폰 수)
18. OrderCount	Total number of orders has been places in last month (지난 달에 이루어진 총 주문 수)
19. DaySinceLastOrder	Day Since last order by customer (마지막 주문 일)
20. CashbackAmount	Average cashback in last month (지난 달 평균 캐쉬백)

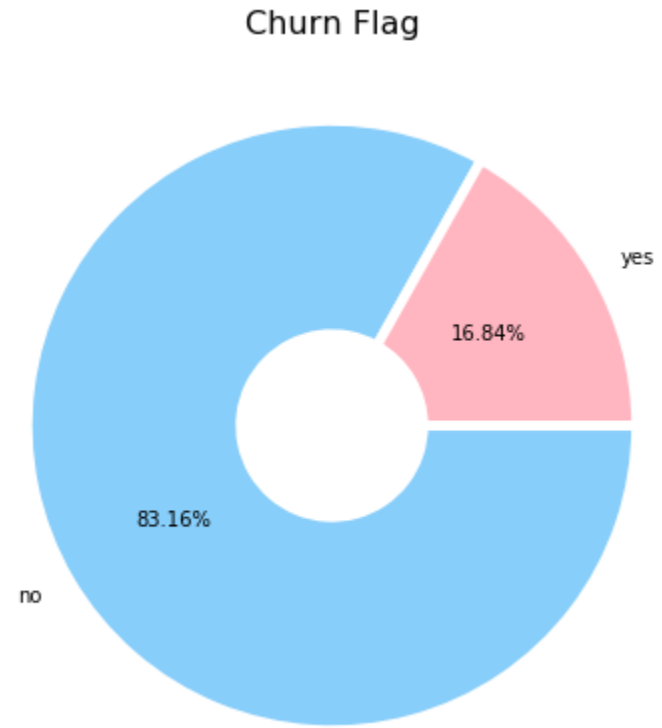
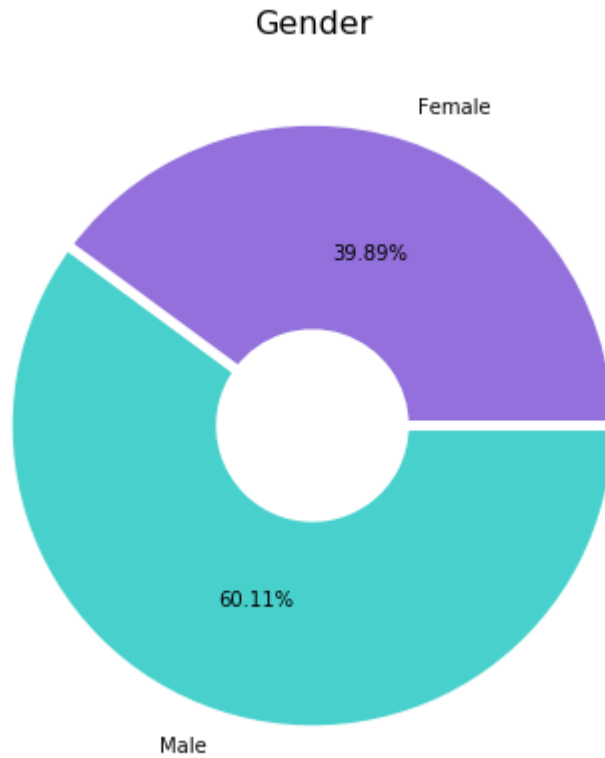
- 이커머스 회사 데이터
- 5630 rows, 20 columns 구성
- 타겟 : 이탈 플래그 (이탈 X - 0, 이탈 O - 1)

Part 3

E D A

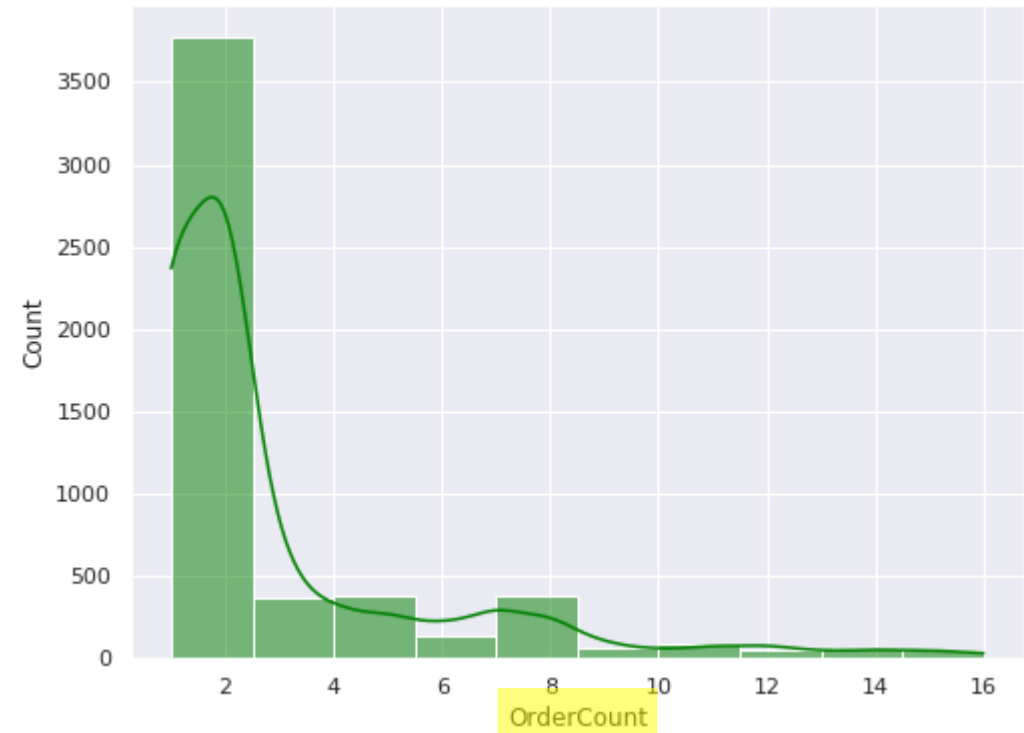
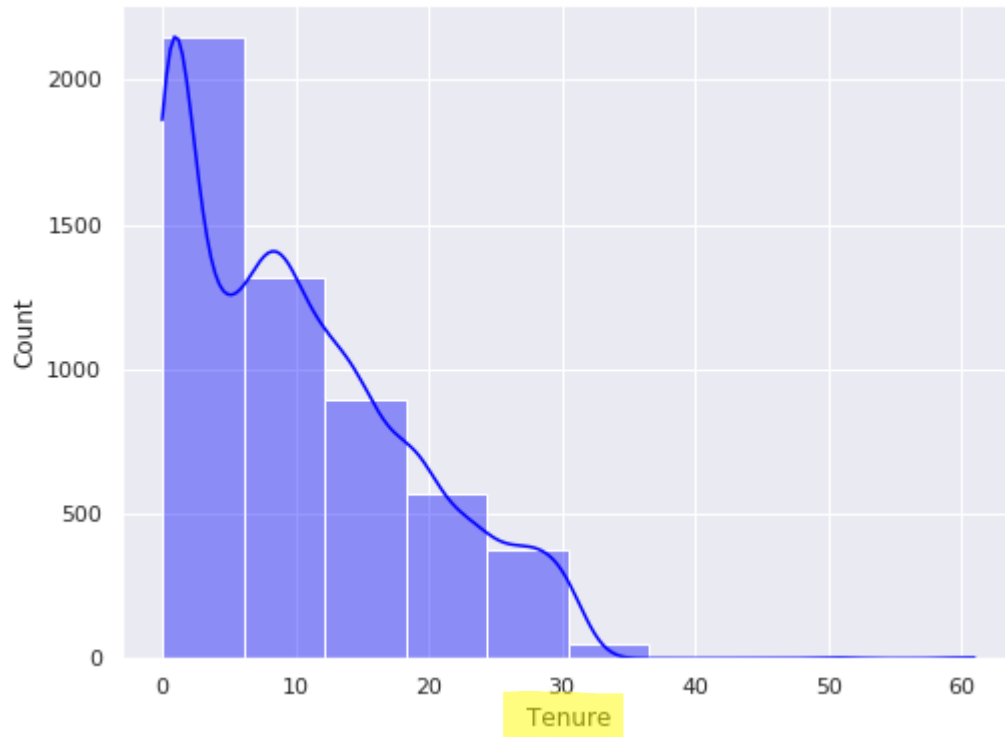


## < E D A >



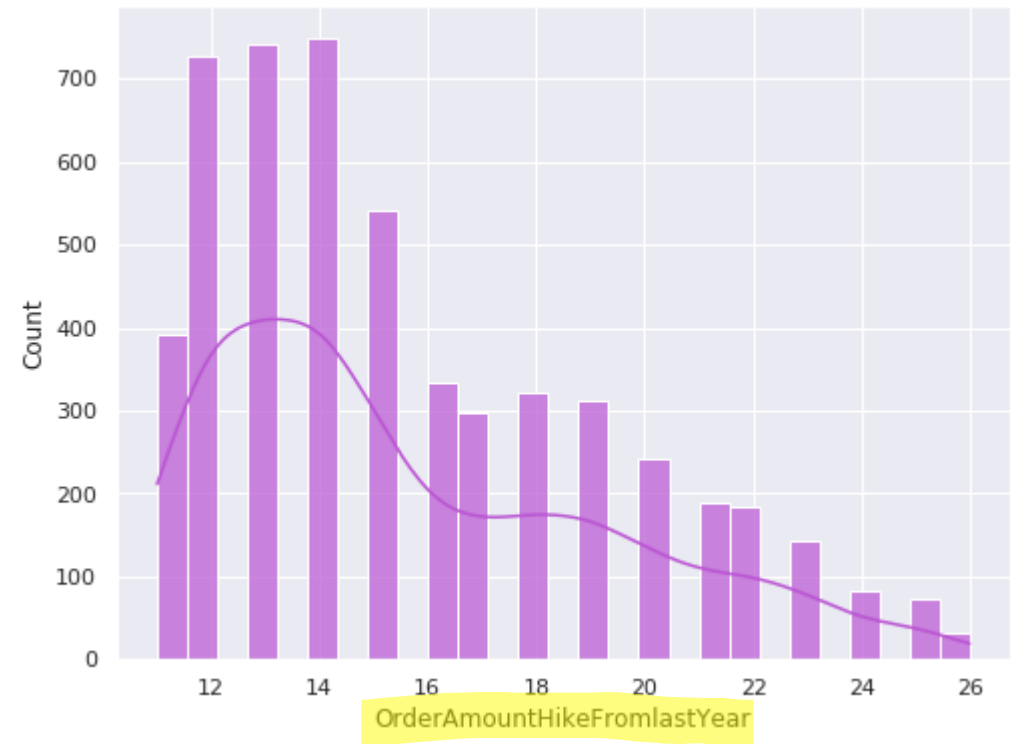
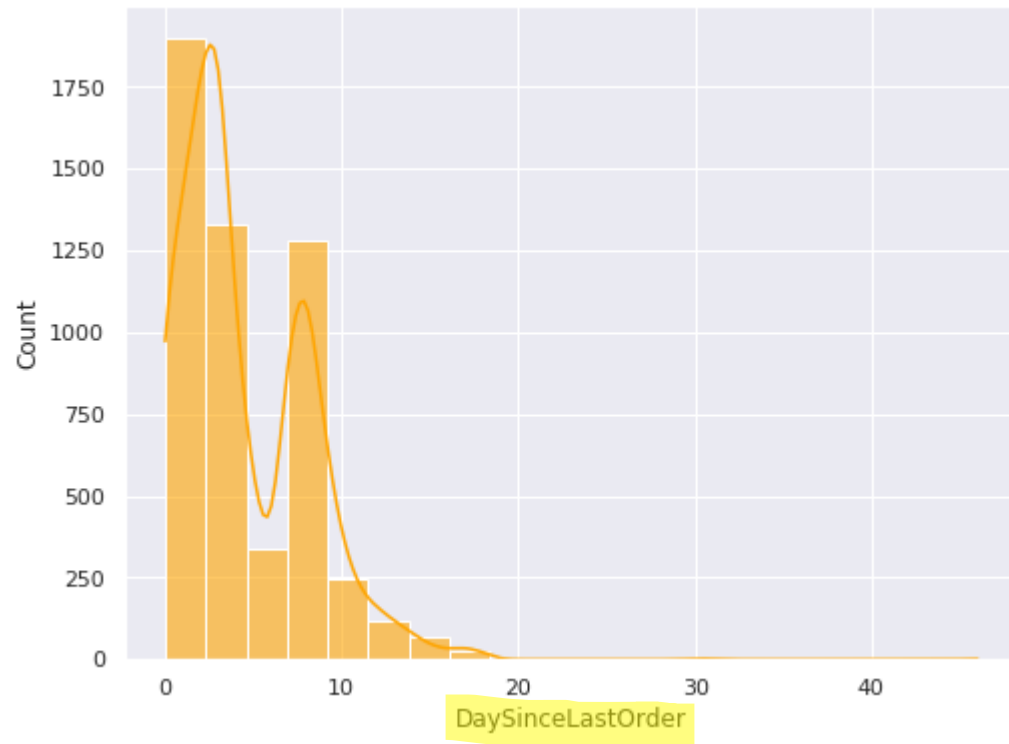
- 성별 : 남성 60%, 여성 40%
- 이탈하지 않은 고객 : 83%, 이탈한 고객 : 17%

## < E D A >



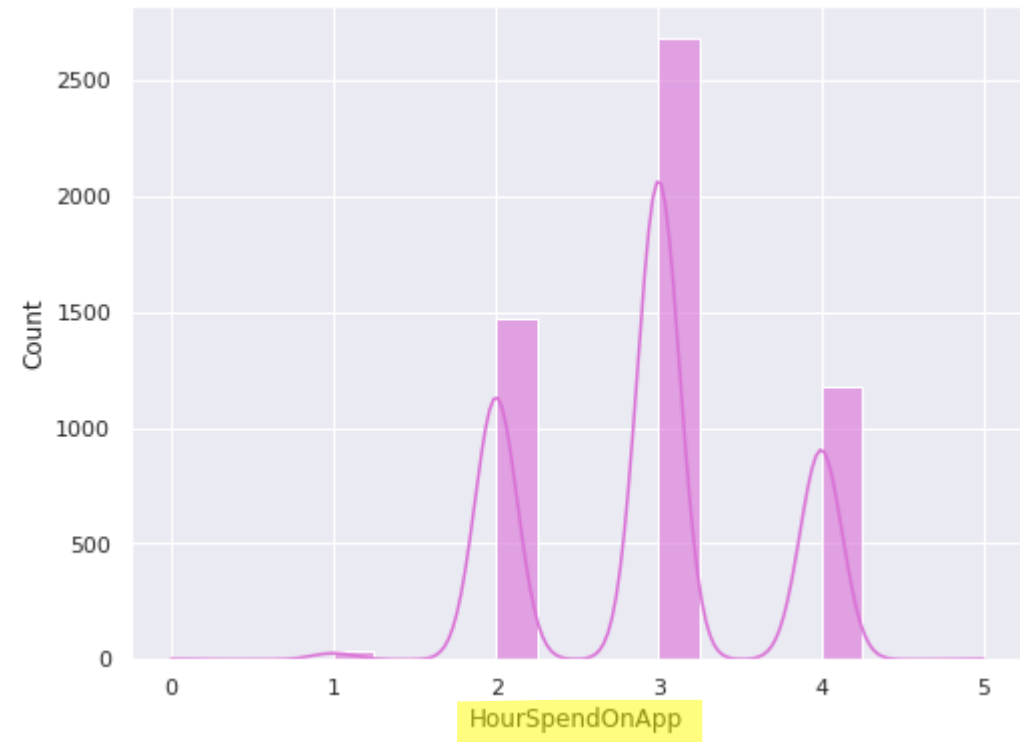
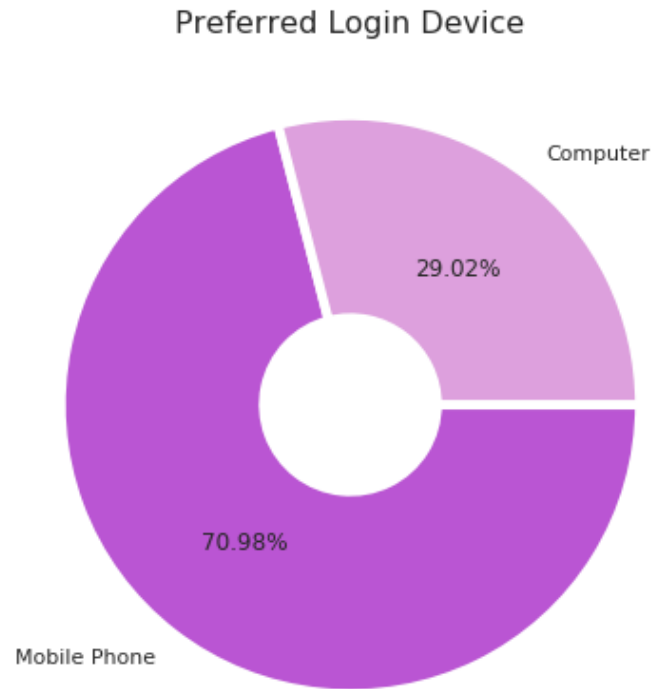
- 근무기간 : 0 ~ 5년이 2,000명 이상으로 가장 많았음
- 지난 달 총 주문 수 : 0 ~ 2번이 3,500명 이상으로 가장 많았음

## < E D A >



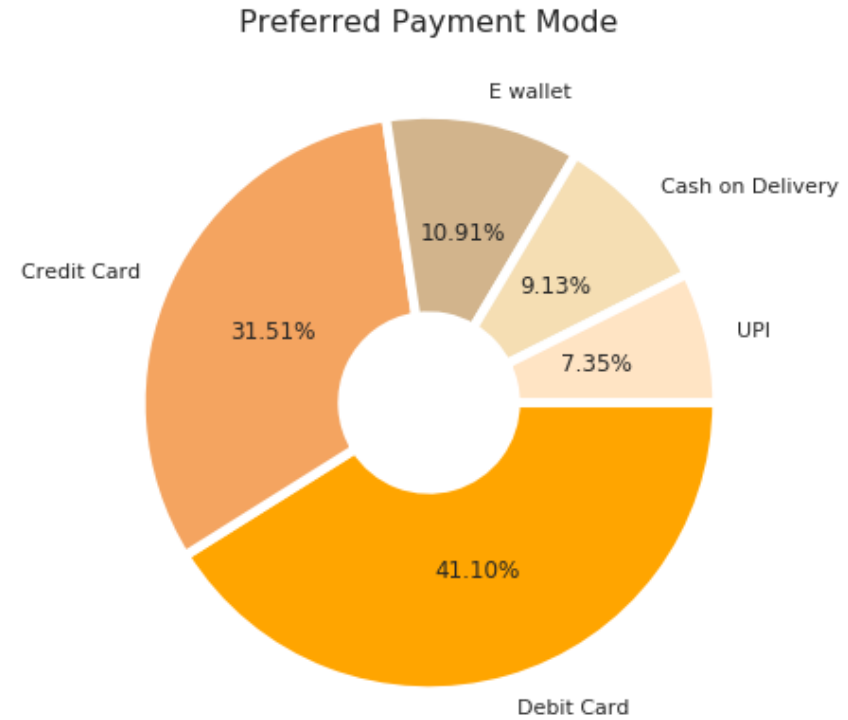
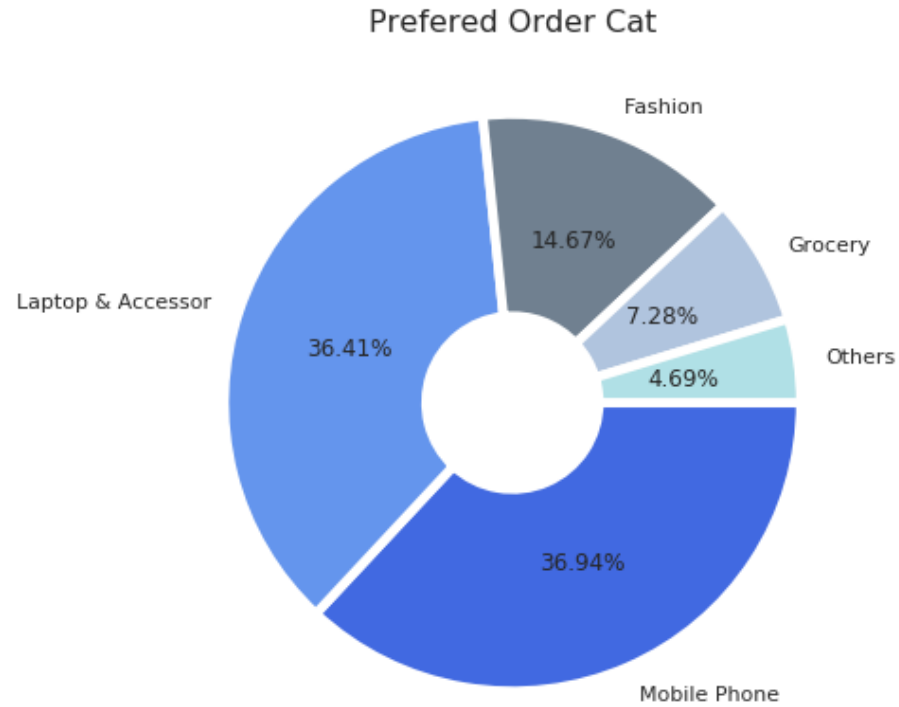
- 최근 주문 : 5일 이내가 3,000명 이상으로 가장 많았음
- 고객 주문 증가율 : 12 ~ 14가 2,219명으로 가장 많음

## < E D A >



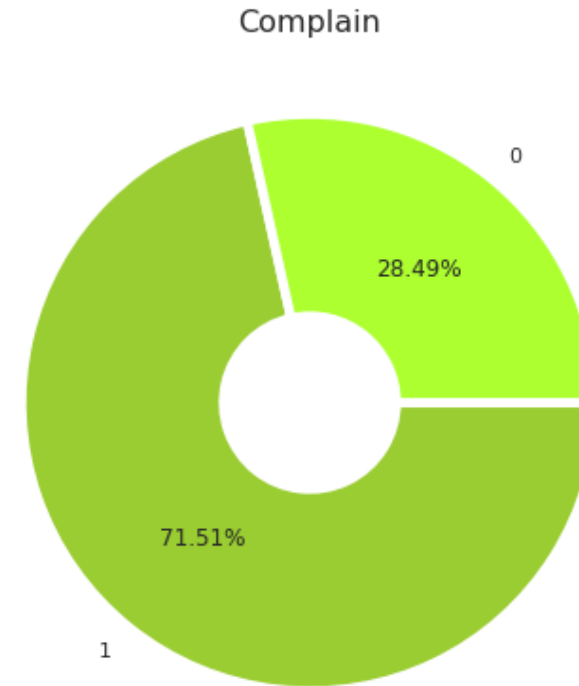
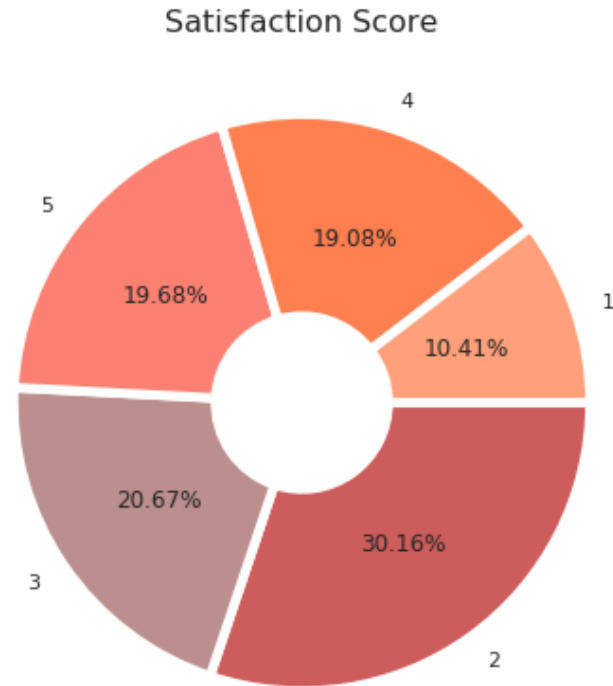
- 선호 로그인 디바이스 : 핸드폰이 71%, 컴퓨터가 29%
- 앱 사용 시간 : 3시간이 2,500명 이상으로 가장 많았음 (보통 사용 시간은 2 ~ 4시간 정도로 보임)

## < E D A >



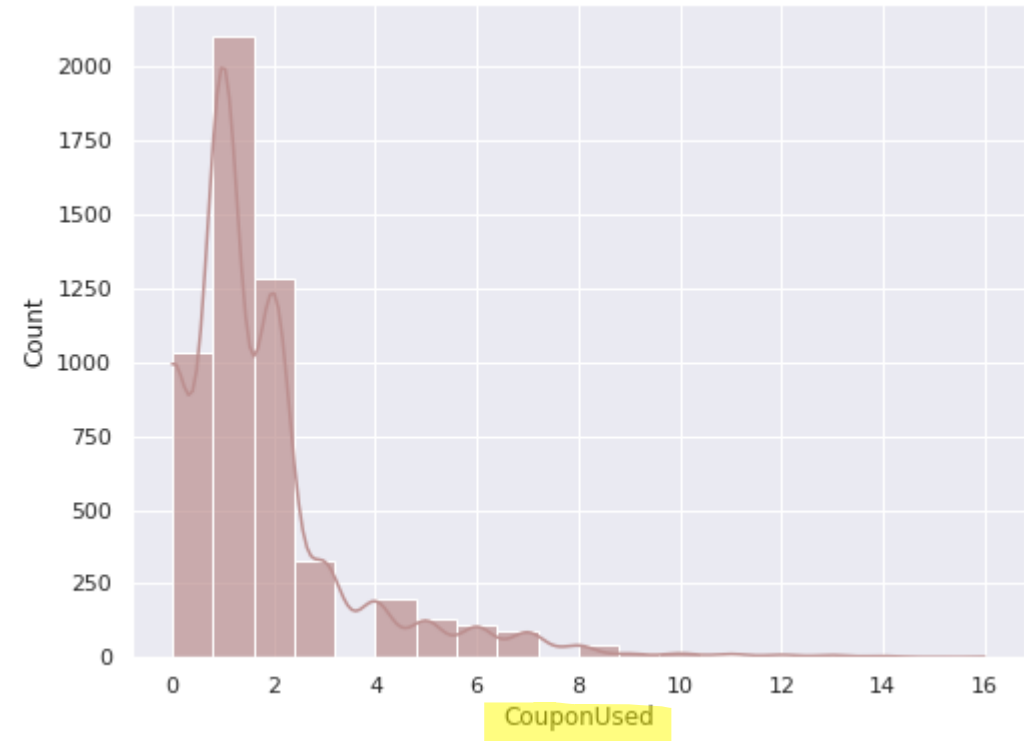
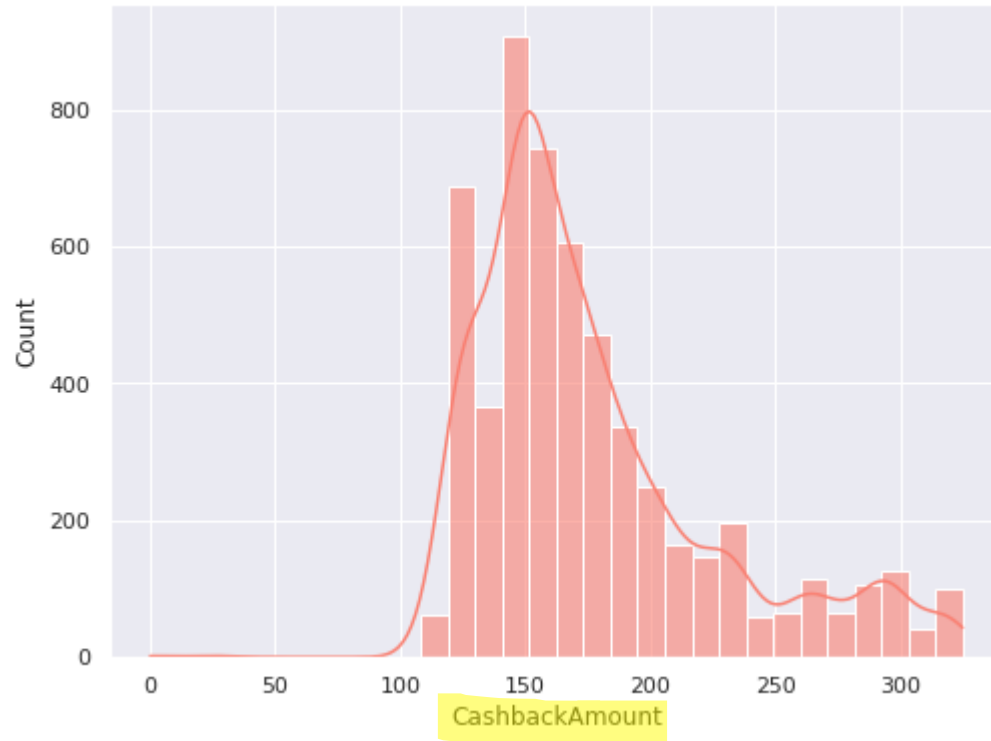
- 선호 구매 제품 : 핸드폰 37%, 랩탑 36%, 패션 15%, 식품 7%, 기타
- 선호 구매 수단 : Debit card(직불카드)와 Credit Card(신용카드)가 72%

## < E D A >



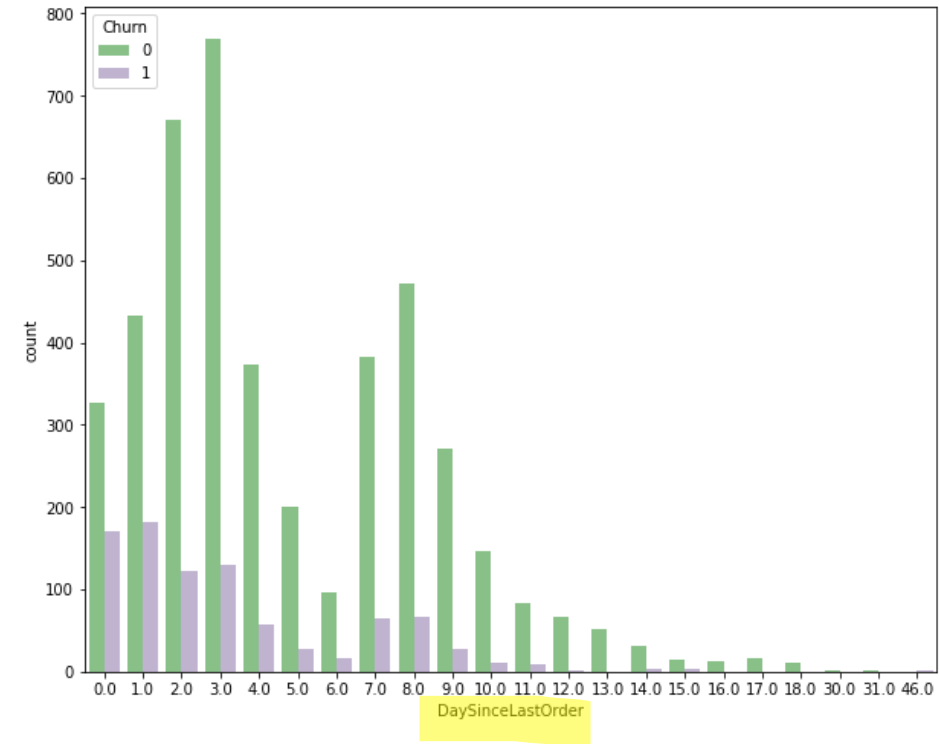
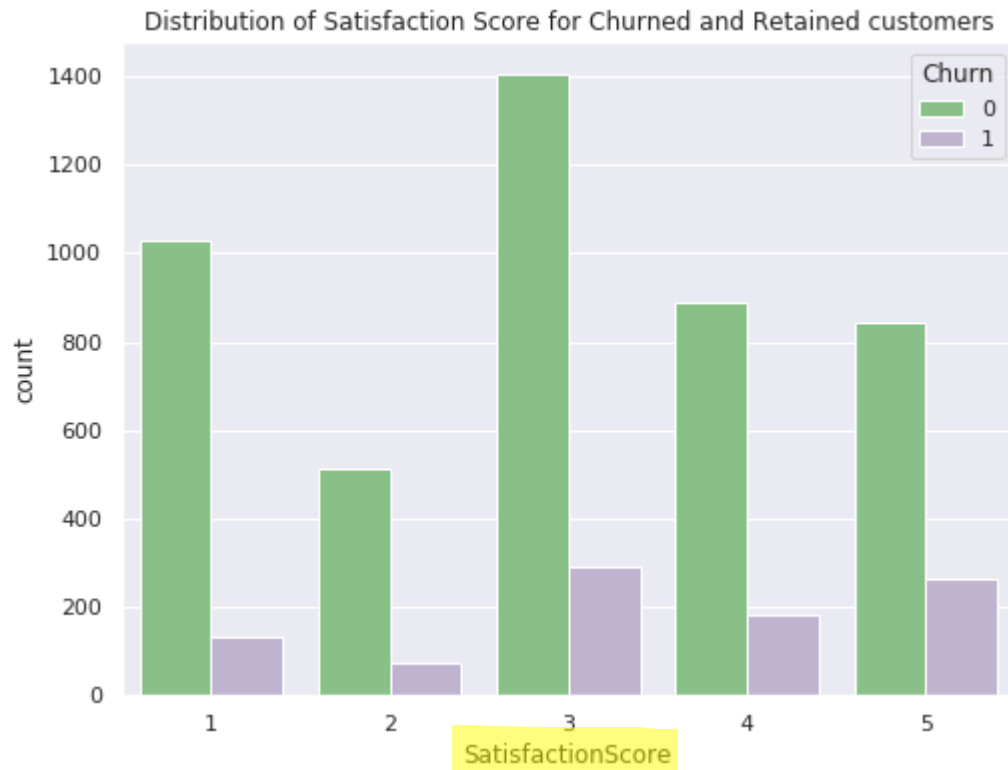
- 고객 만족도 : 2점 30%, 3점 21%, 5점 20%
- 컴플레인 : 72%가 컴플레인 X

## < E D A >



- 지난 달 쌓은 캐쉬백 : 150 포인트가 800명 이상으로 가장 많음
- 지난 달 사용 쿠폰 : 1개가 2,000명 이상으로 가장 많음

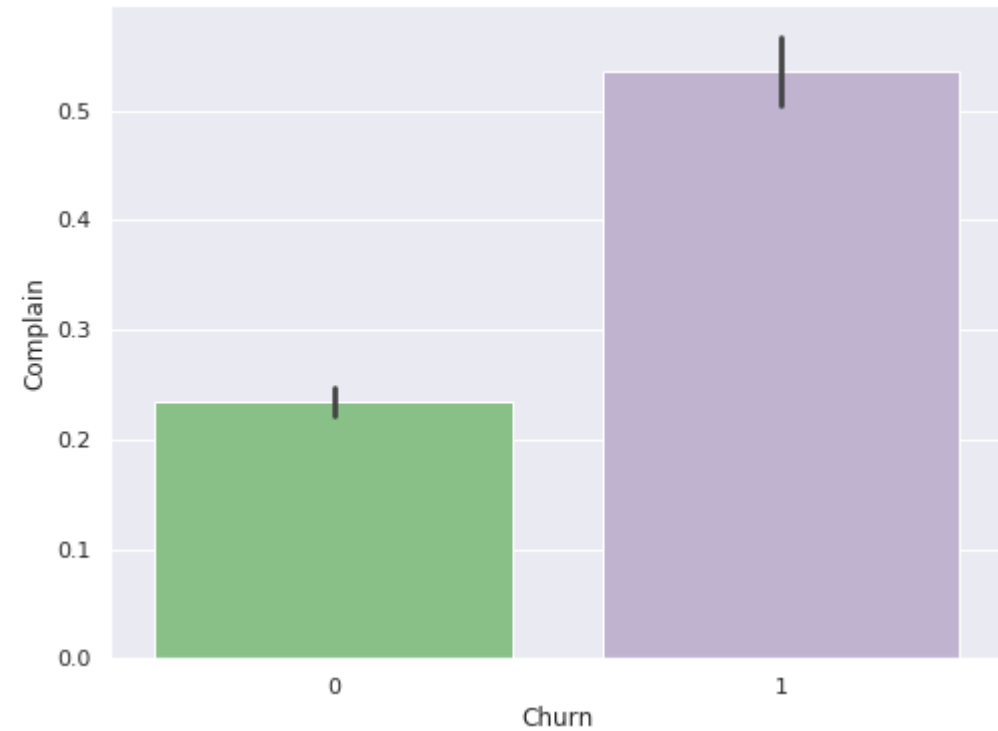
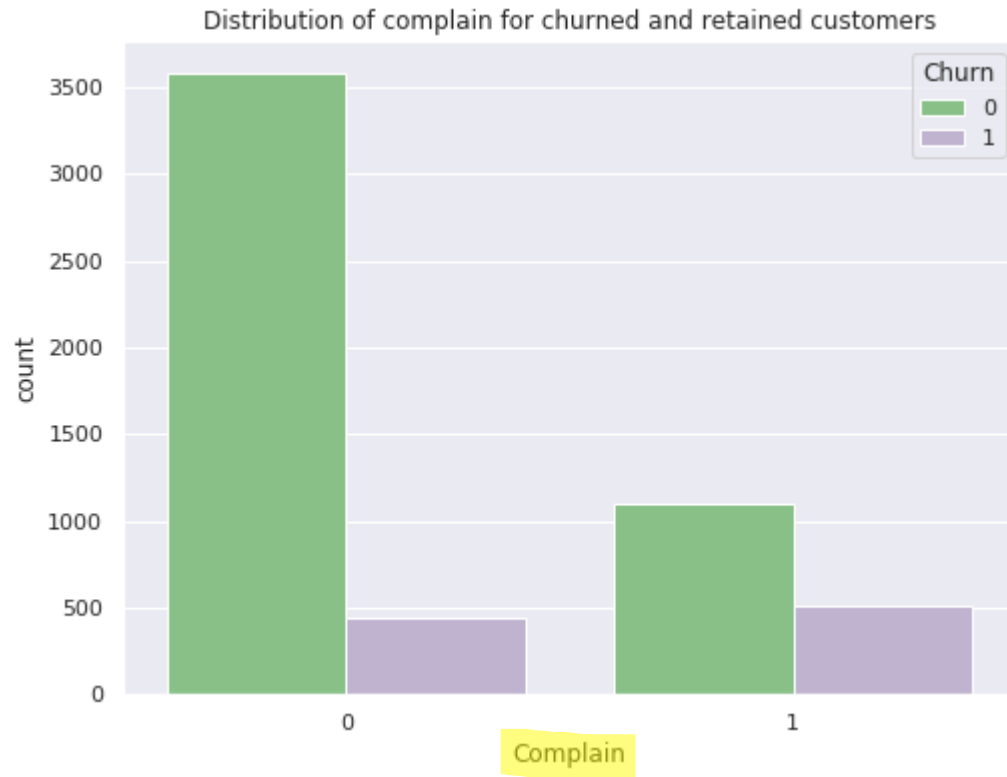
## < E D A >



- 만족도에 따른 고객 이탈률 : 영향을 미치긴 하지만 미미한 수준
- 최근 주문에 따른 고객 이탈률 : 최근 마지막으로 주문한 고객들이 그렇지 않은 고객보다 이탈률 차이가 남

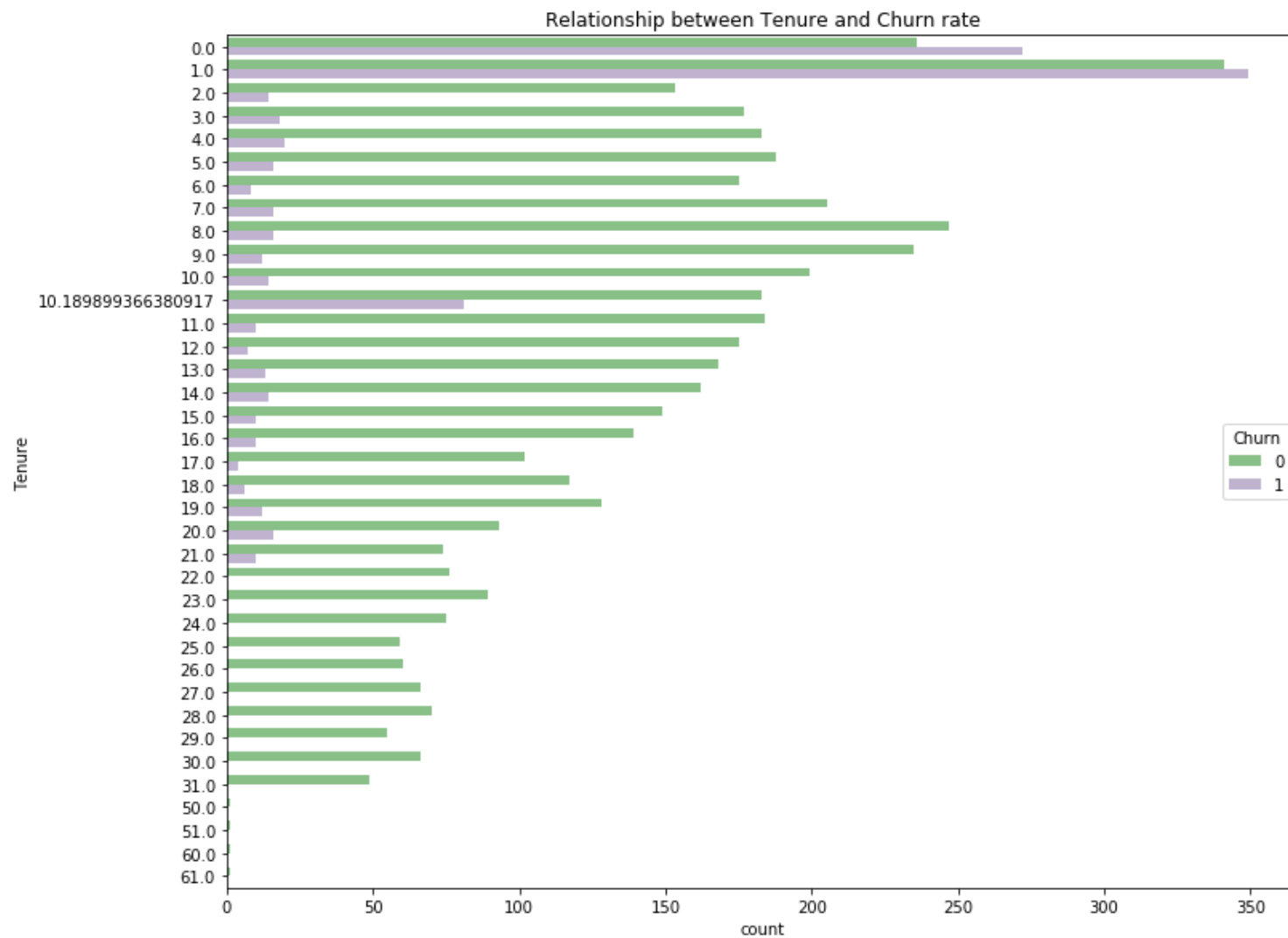


## < E D A >



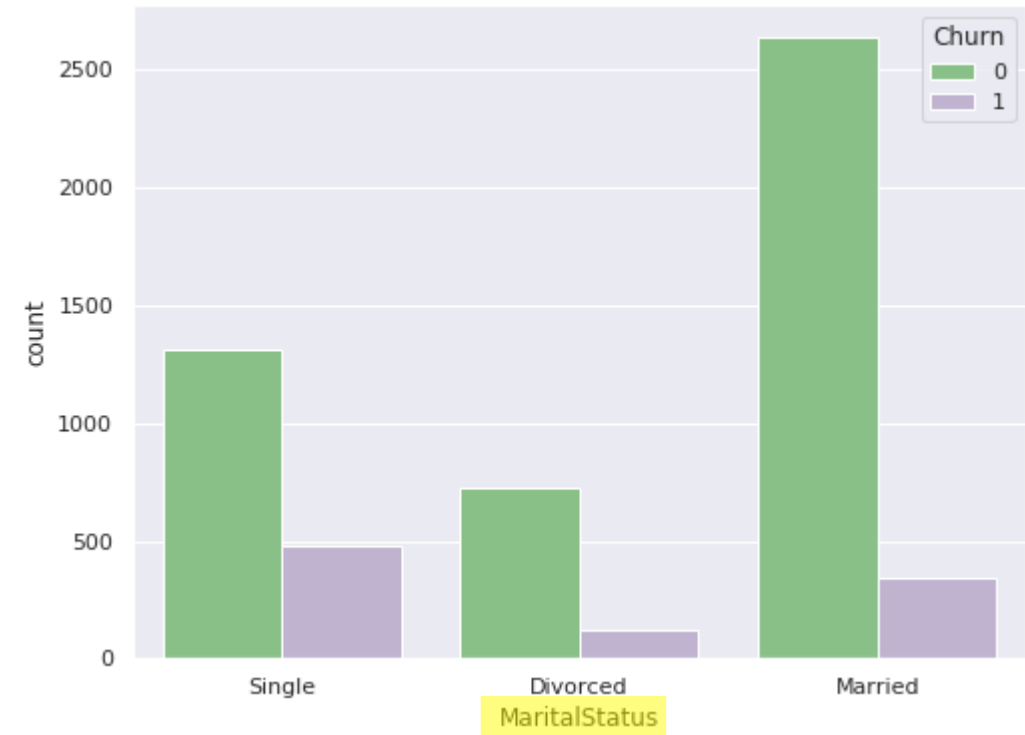
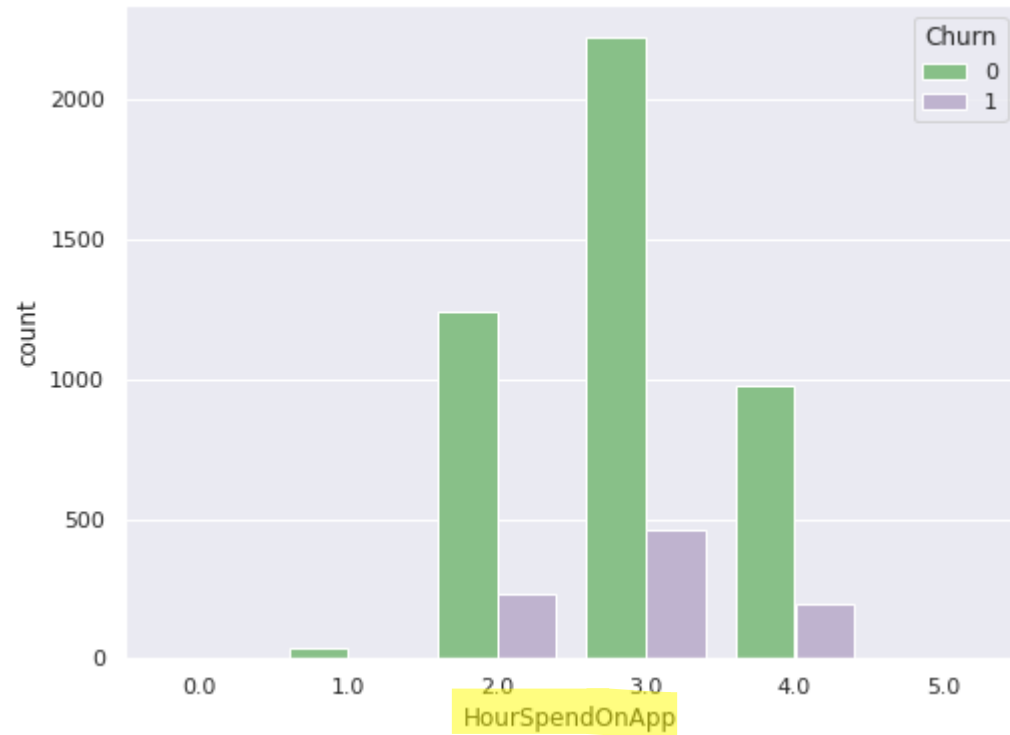
- 컴플레인에 따른 고객 이탈률 : 컴플레인을 건 쪽과 걸지 않은 쪽의 이탈률 차이가 큼
- 컴플레인으로 인한 이탈률(31.67%)은 컴플레인을 걸지 않은 고객의 전체 이탈률(10.93%)보다 3배 높음

## < E D A >

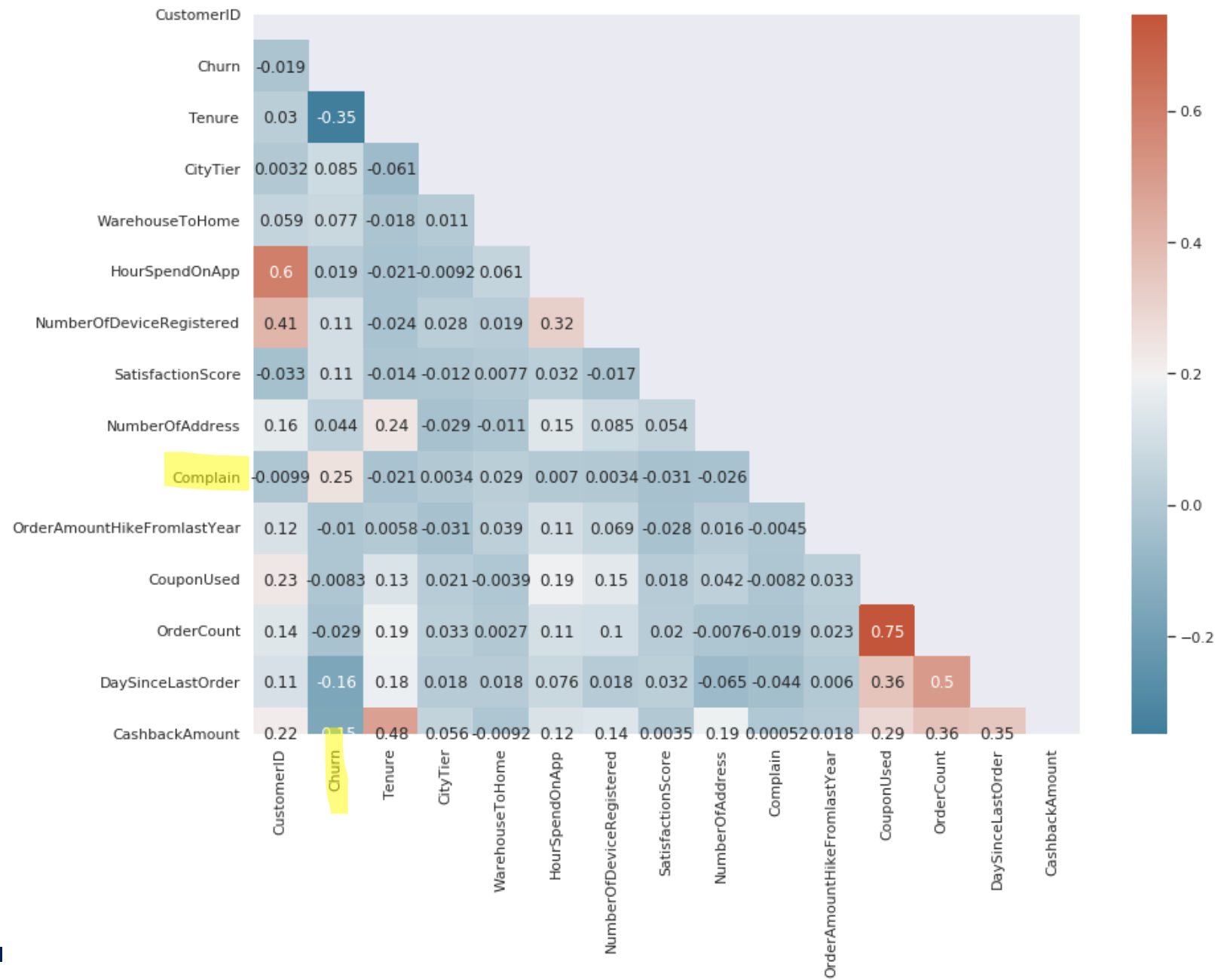


- 고객 가입 기간(개월)이 적을 수록 이탈률이 아주 크게 나타남

## < E D A >



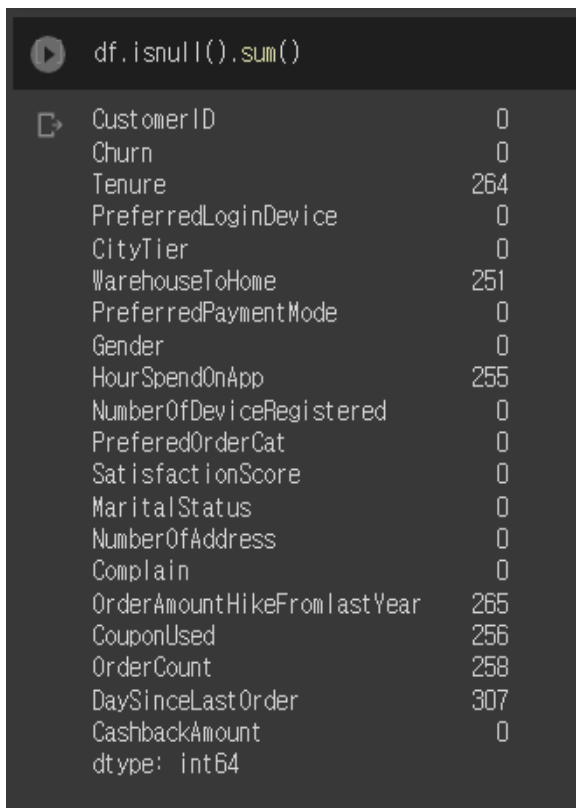
- 앱 사용시간에 따른 이탈률 : 3시간 정도 사용이 이탈률 가장 높음
- 결혼 여부에 따른 이탈률 : 싱글일수록 이탈률이 높음



## ☆ 정리 ☆

- 이탈률 : 이탈 X - 83%, 이탈 O - 17%
- 근무기간 : 0 ~ 5년이 2000개 이상으로 가장 많았음
- 지난 달 총 주문 수 : 0 ~ 2번이 3500명 이상으로 가장 많았음
- 최근 주문 : 5일 이내가 3000명 이상으로 가장 많았음
- 고객 주문 증가율 : 14가 750명으로 가장 많음
- 선호하는 로그인 디바이스 : Mobile Phone과 Phone 합쳐줌, 모바일 폰이 71%, 컴퓨터가 29%
- 선호하는 구매 : 1. 모바일 폰 37%, 랩탑 36%, 패션 15%, 식품 7%, 기타 5%
- 선호하는 구매 수단 : Debit card(직불카드) 41% Credit Card(신용카드) 32%, e-wallet 11%, cash on delivery 9%, UPI(유니온페이) 7%
- 고객만족도 : 1점(10%), 2점(30%), 3점(21%), 4점(19%), 5점(20%)
- 가입기간(개월) : 적을 수록 많이 이탈함
- 컴플레인 : 컴플레인 X - 72%, 컴플레인 O - 28%
- 전 달 캐쉬백 : 150 포인트가 800명 이상으로 가장 많음
- 쿠폰 사용 : 1개 사용이 2000명 이상으로 가장 많음
- 결혼 여부 : 싱글이 이탈률 더 높음
- 고객만족도는 이탈률에 영향 미침
- 이탈률에 가장 큰 영향을 미치는 변수는 가입 기간 및 컴플레인

## < E D A >



```
df.isnull().sum()
```

CustomerID	0
Churn	0
Tenure	264
PreferredLoginDevice	0
CityTier	0
WarehouseToHome	251
PreferredPaymentMode	0
Gender	0
HourSpendOnApp	255
NumberOfDeviceRegistered	0
PreferedOrderCat	0
SatisfactionScore	0
MaritalStatus	0
NumberOfAddress	0
Complain	0
OrderAmountHikeFromLastYear	265
CouponUsed	256
OrderCount	258
DaySinceLastOrder	307
CashbackAmount	0
dtype: int64	



```
[5] df.isnull().sum()
```

CustomerID	0
Churn	0
Tenure	0
PreferredLoginDevice	0
CityTier	0
WarehouseToHome	0
PreferredPaymentMode	0
Gender	0
HourSpendOnApp	0
NumberOfDeviceRegistered	0
PreferedOrderCat	0
SatisfactionScore	0
MaritalStatus	0
NumberOfAddress	0
Complain	0
OrderAmountHikeFromLastYear	0
CouponUsed	0
OrderCount	0
DaySinceLastOrder	0
CashbackAmount	0
dtype: int64	

- 총 데이터 수가 너무 적어 결측치 drop 할 수 없음
- 결측치를 0으로 채우면 분석에 영향 감
- → 각 변수의 평균값으로 결측치 처리 (+ 중복값 0)

## < E D A >

```
▼ - PreferredOrderCat
(Mobile Phone과 Mobile 합쳐줌)

[ ] # PreferredOrderCat
df['PreferredOrderCat'].value_counts()

Laptop & Accessory    2050
Mobile Phone         1271
Fashion              826
Mobile              809
Grocery              410
Others               264
Name: PreferredOrderCat, dtype: int64

[ ] # Mobile Phone과 Mobile 합쳐줌
df['PreferredOrderCat'] = df['PreferredOrderCat'].replace('Mobile', 'Mobile Phone')

df['PreferredOrderCat'].value_counts()

Mobile Phone         2080
Laptop & Accessory    2050
Fashion              826
Grocery              410
Others               264
Name: PreferredOrderCat, dtype: int64
```

```
▼ - PreferredPaymentMode
(CC와 Credit Card 합쳐줌) (COD와 Cash on Delivery 합쳐줌)

[ ] df['PreferredPaymentMode'].value_counts()

Debit Card          2314
Credit Card         1501
E wallet            614
UPI                 414
COD                 365
CC                  273
Cash on Delivery    149
Name: PreferredPaymentMode, dtype: int64

[ ] # CC와 Credit Card 합쳐줌
df['PreferredPaymentMode'] = df['PreferredPaymentMode'].replace('CC', 'Credit Card')

df['PreferredPaymentMode'].value_counts()

Debit Card          2314
Credit Card         1774
E wallet            614
UPI                 414
COD                 365
Cash on Delivery    149
Name: PreferredPaymentMode, dtype: int64

[ ] # COD와 Cash on Delivery 합쳐줌
df['PreferredPaymentMode'] = df['PreferredPaymentMode'].replace('COD', 'Cash on Delivery')

df['PreferredPaymentMode'].value_counts()

Debit Card          2314
Credit Card         1774
E wallet            614
Cash on Delivery    514
UPI                 414
Name: PreferredPaymentMode, dtype: int64
```

- 같은 이름 합쳐줌
- Mobile phone - Phone
- CC - Credit Card / COD - Cash On Delivery

Part 4

## 모 델 링



## < 모델링 >

```
[6] from sklearn.model_selection import train_test_split
    X = df.drop('Churn', axis=1)
    y = df['Churn']

    # 테스트 사이즈 20%
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

[7] # train, test 데이터가 잘 나뉘었는지 확인

    X_train.shape, X_test.shape, y_train.shape, y_test.shape

((4504, 19), (1126, 19), (4504,), (1126,))
```

- Train 80%, test 20%
- 원핫인코딩 후 데이터 → 4504개 샘플, 35개의 컬럼으로 구성

## < 모델링 >

```
# 기준 모델

from sklearn.metrics import accuracy_score
from sklearn.metrics import r2_score, mean_absolute_error

base = y_train.mode()[0]
# 타겟 샘플 수 만큼 0이 담긴 리스트를 만듦. 기준모델로 예측
baseline = len(y_train) * [base]

baseline_acc = accuracy_score(y_train, baseline)
print(f'기준모델 정확도: {baseline_acc.round(2)}')

#baseline_r2 = r2_score(y_train, baseline)
#print(f'기준모델의 r2_score: {baseline_r2}')

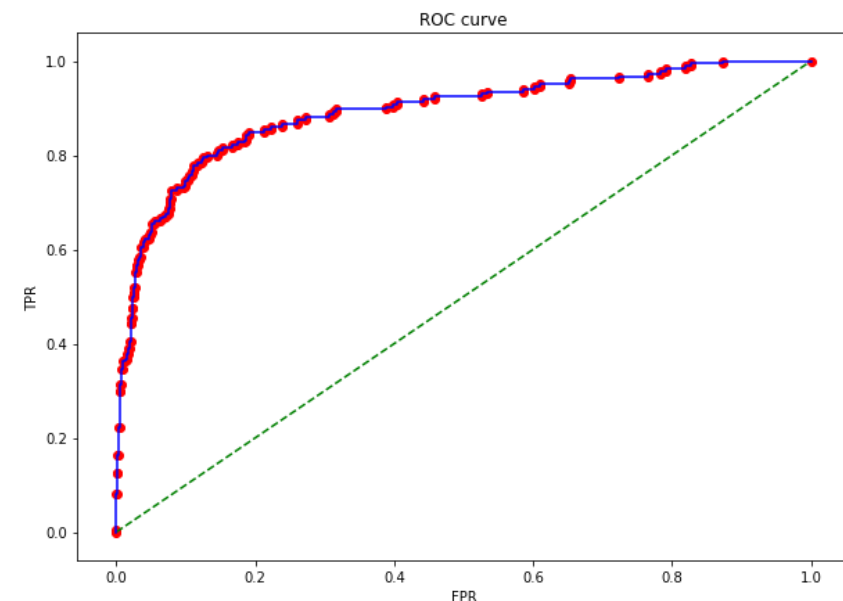
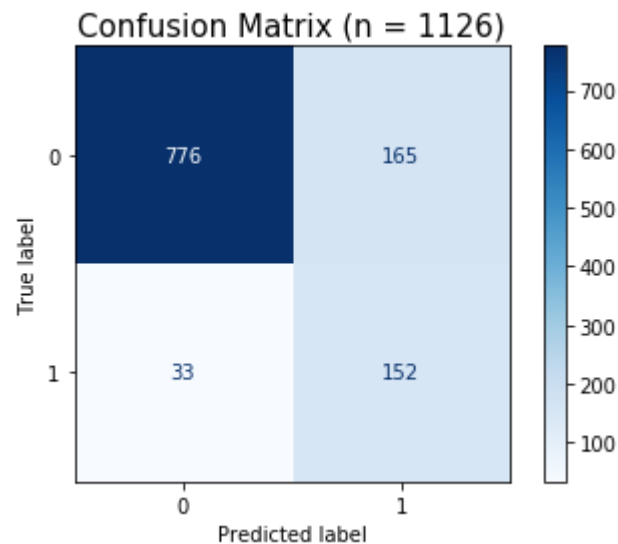
baseline_mae = mean_absolute_error(y_train, baseline)
print(f'기준모델의 mae : {baseline_mae}')
```

기준모델 정확도: 0.83  
기준모델의 mae : 0.169404973357016

- 기준모델 만듦
- 기준모델 정확도 : 0.83

## < 로지스틱 회귀 모델 >

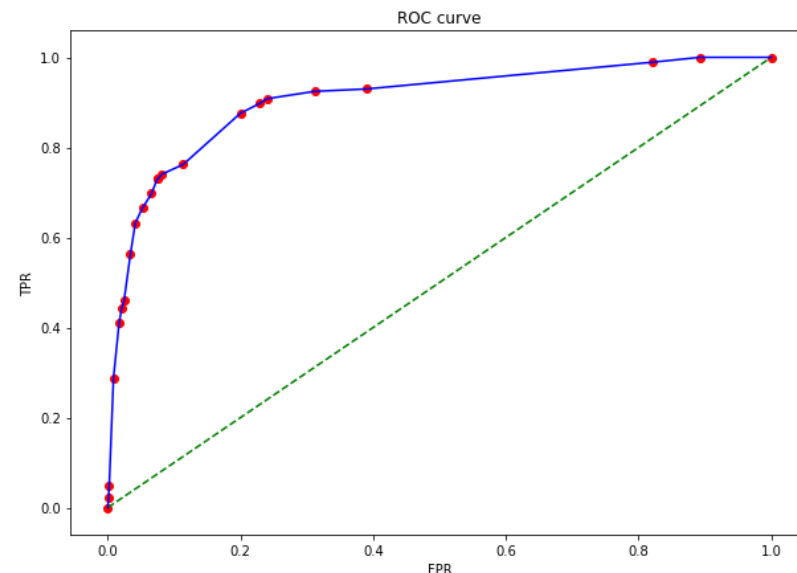
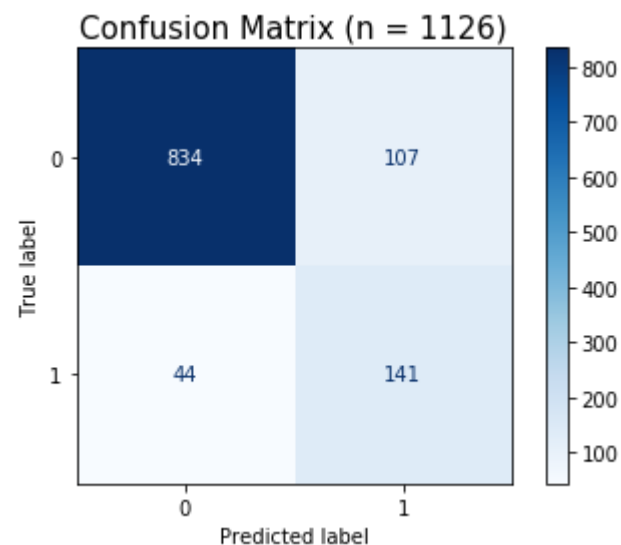
	precision	recall	f1-score	support
0	0.96	0.82	0.89	941
1	0.48	0.82	0.61	185
accuracy			0.82	1126
macro avg	0.72	0.82	0.75	1126
weighted avg	0.88	0.82	0.84	1126



- 로지스틱 회귀 모델 정확도 : 0.82
- Train 정확도 : 0.84 / Test 정확도 : 0.85
- class\_weight="balanced" <- 가중치 줘서 과적합 해결
- AUC : 0.89
- Test score for Logistic Regression: 0.6055776892430279 (0.6)
- Training score for Logistic Regression: 0.5886757136172204 (0.59)

## < 결정 트리 >

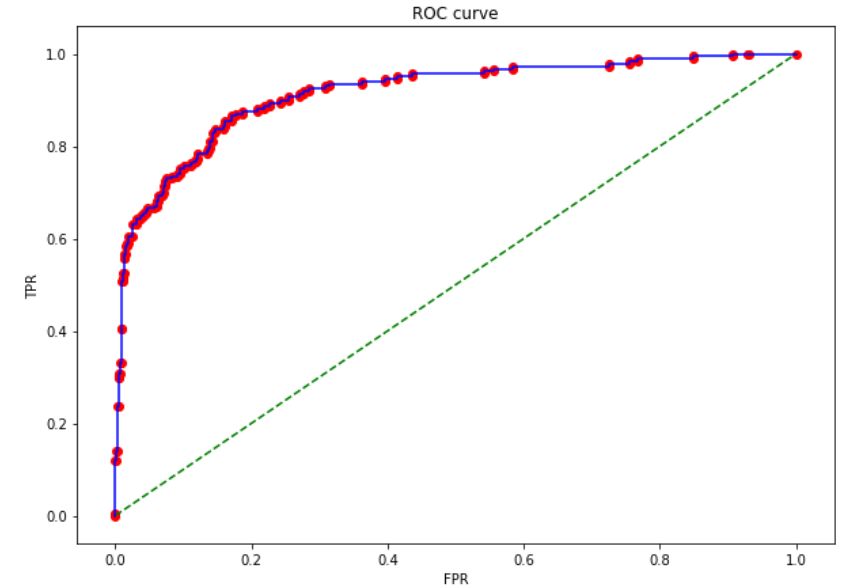
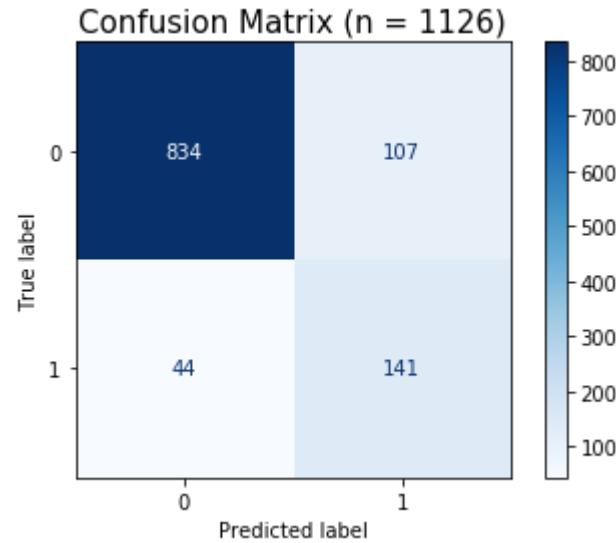
	precision	recall	f1-score	support
0	0.95	0.89	0.92	941
1	0.57	0.76	0.65	185
accuracy			0.87	1126
macro avg	0.76	0.82	0.78	1126
weighted avg	0.89	0.87	0.87	1126



- 결정 트리 모델 정확도 : 0.87
- Train 정확도 : 1.0 / Test 정확도 : 0.96
- max\_depth=6, class\_weight="balanced" <- 가중치 줘서 과적합 해결
- AUC : 0.90
- Test score for DecisionTree: 0.651270207852194 (0.65)
- Training score for DecisionTree: 0.6889742183214481 (0.69)

## < 랜덤 포레스트 >

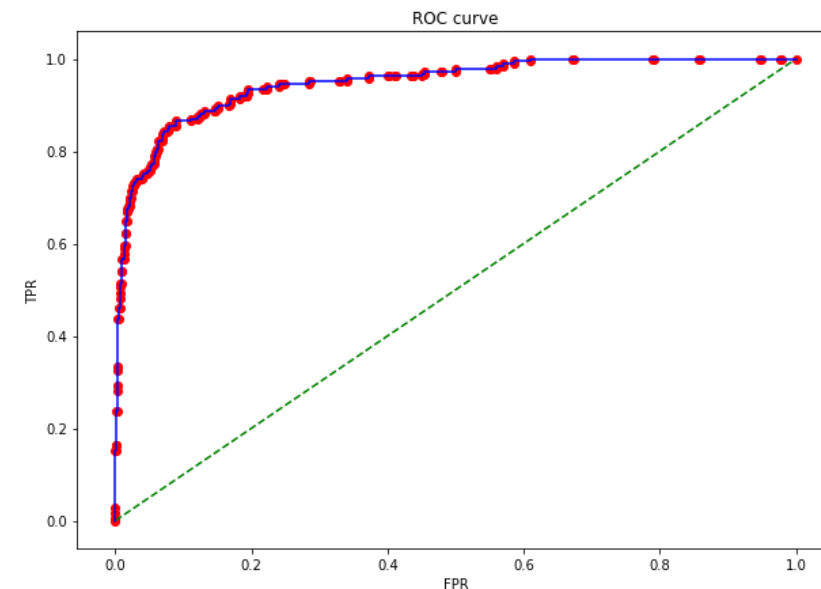
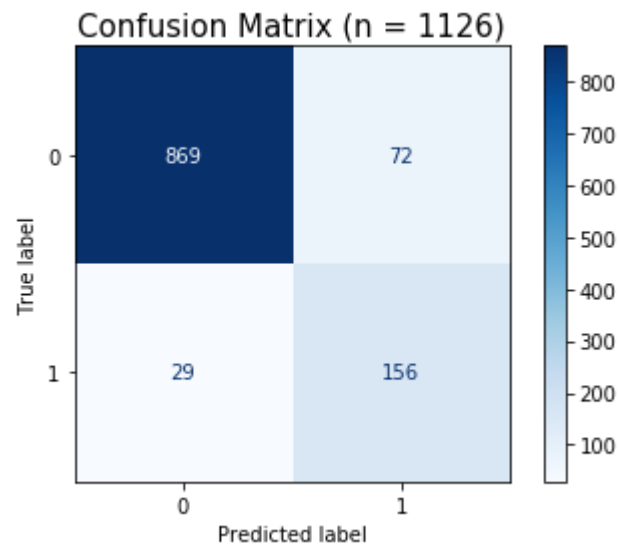
	precision	recall	f1-score	support
0	0.95	0.89	0.92	941
1	0.57	0.76	0.65	185
accuracy			0.87	1126
macro avg	0.76	0.82	0.78	1126
weighted avg	0.89	0.87	0.87	1126



- 랜덤 포레스트 모델 정확도 : 0.87
- Train 정확도 : 1.0 / Test 정확도 : 0.97
- max\_depth=6, class\_weight="balanced" <- 가중치 줘서 과적합 해결
- AUC : 0.92
- Test score for Random Forest: 0.651270207852194 (0.65)
- Training score for Random Forest: 0.7018498367791076 (0.7)

## < XGBoost >

	precision	recall	f1-score	support
0	0.97	0.92	0.95	941
1	0.68	0.84	0.76	185
accuracy			0.91	1126
macro avg	0.83	0.88	0.85	1126
weighted avg	0.92	0.91	0.91	1126

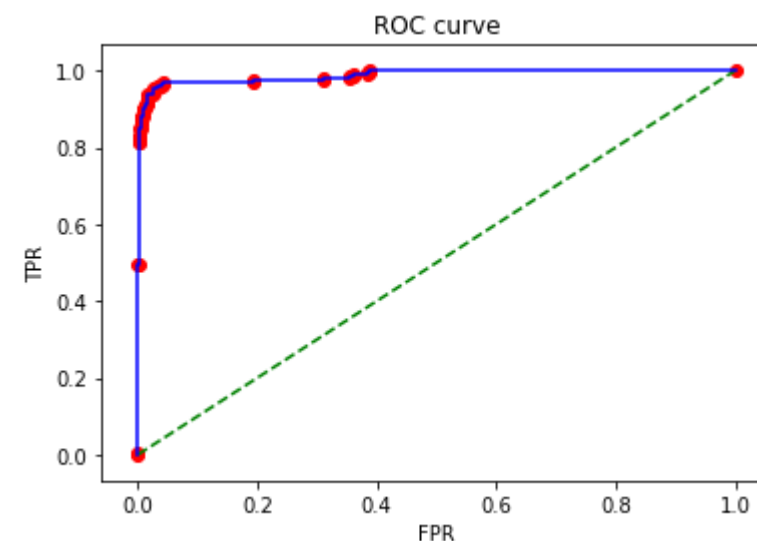
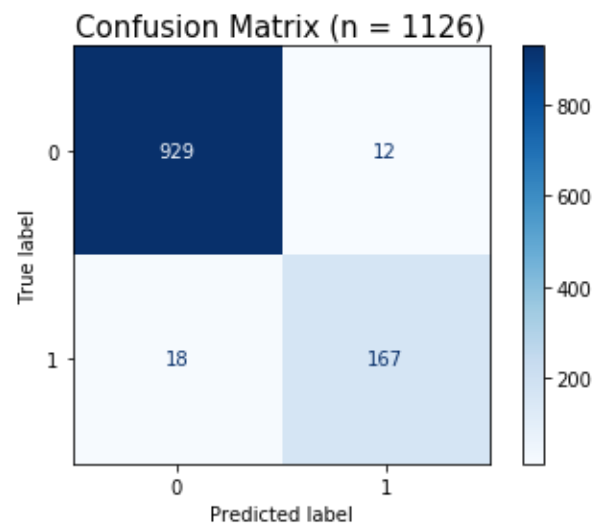


- XGBoost 모델 정확도 : 0.91
- Train 정확도 : 0.93 / Test 정확도 : 0.92
- AUC : 0.95
- Test score for XGBoost : 0.7554479418886199 (0.76)
- Training score for XGBoost : 0.8145620022753128 (0.81)

## < RandomSearchCV – XGBoost >

```
최적 하이퍼파라미터: {'simpleimputer__strategy': 'median', 'xgbclassifier__colsample_bytree': 0.5499874579090014, 'xgbclassifier__max_depth': 8, 'xgbclassifier__min_child_weight': 4}  
최적 AUC: 0.9696323928186744
```

	precision	recall	f1-score	support
0	0.98	0.99	0.98	941
1	0.93	0.90	0.92	185
accuracy			0.97	1126
macro avg	0.96	0.94	0.95	1126
weighted avg	0.97	0.97	0.97	1126



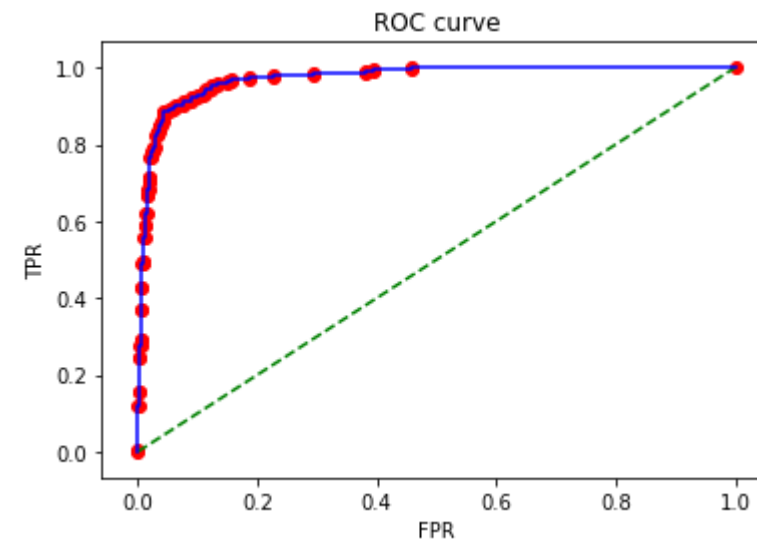
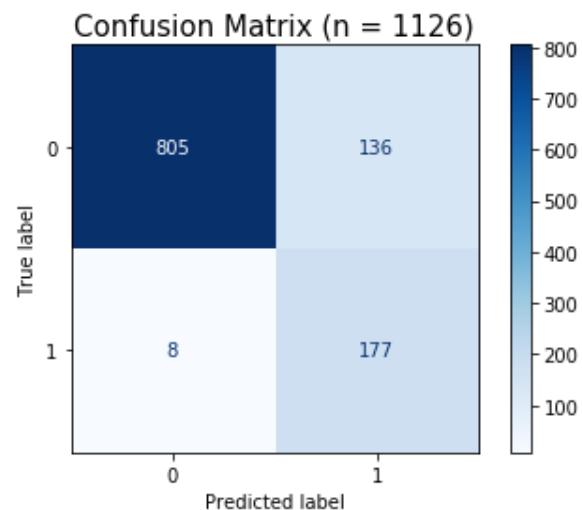
- 최적 하이퍼파라미터 찾기 위해 RandomSearchCV 사용
- XGBoost 모델 정확도 : 0.97
- AUC : 0.99
- Test score for XGBoost: 0.9175824175824175 (0.92)
- Training score for XGBoost: 0.9980379332897319 (1.0)
- → 과적합된 것을 볼 수 있음 → 데이터 수가 너무 불균형해서 그런가? → 언더샘플링

## < Undersampling – XGBoost >

```
[96] # Undersampling 시 동일한 비율로 샘플링  
y_train_sampled.value_counts()
```

```
0    763  
1    763  
Name: Churn, dtype: int64
```

	precision	recall	f1-score	support
0	0.99	0.86	0.92	941
1	0.57	0.96	0.71	185
accuracy			0.87	1126
macro avg	0.78	0.91	0.81	1126
weighted avg	0.92	0.87	0.88	1126



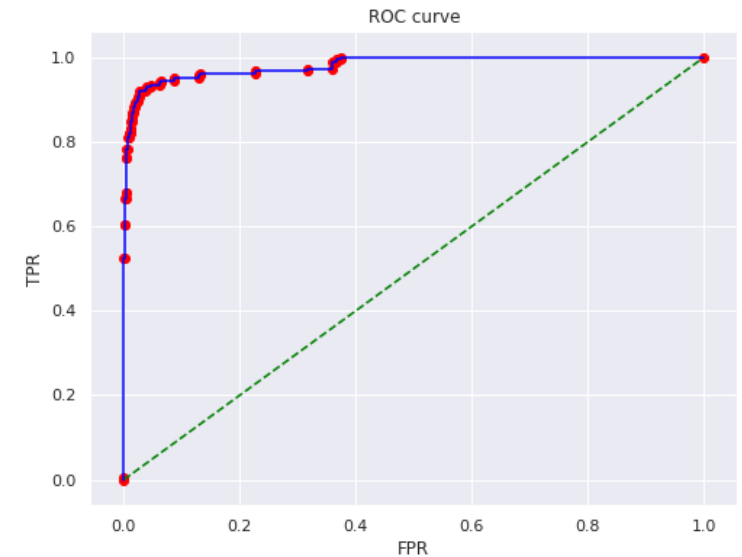
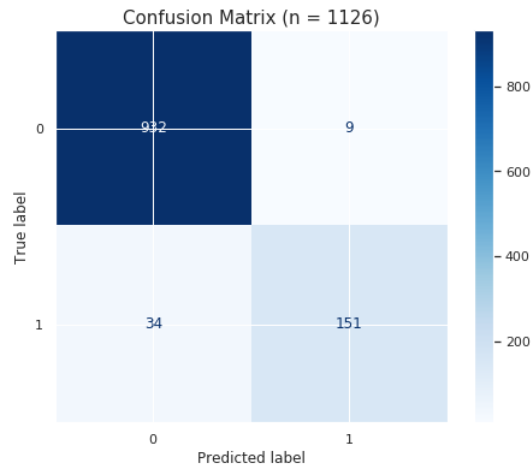
- XGBoost 모델 정확도 : 0.87
- AUC : 0.97
- Test score for XGBoost: 0.7108433734939759 (0.71)
- Training score for XGBoost: 0.7660642570281123 (0.77)
- → 기존 XGBoost test score와 비교하면 학습에 사용된 데이터 수가 크게 감소되어 더 낮은 성능 결과를 반환한 것으로 보임
- 기존 데이터 수가 너무 적어서 오버샘플링으로 다시 학습시켜보고자 함



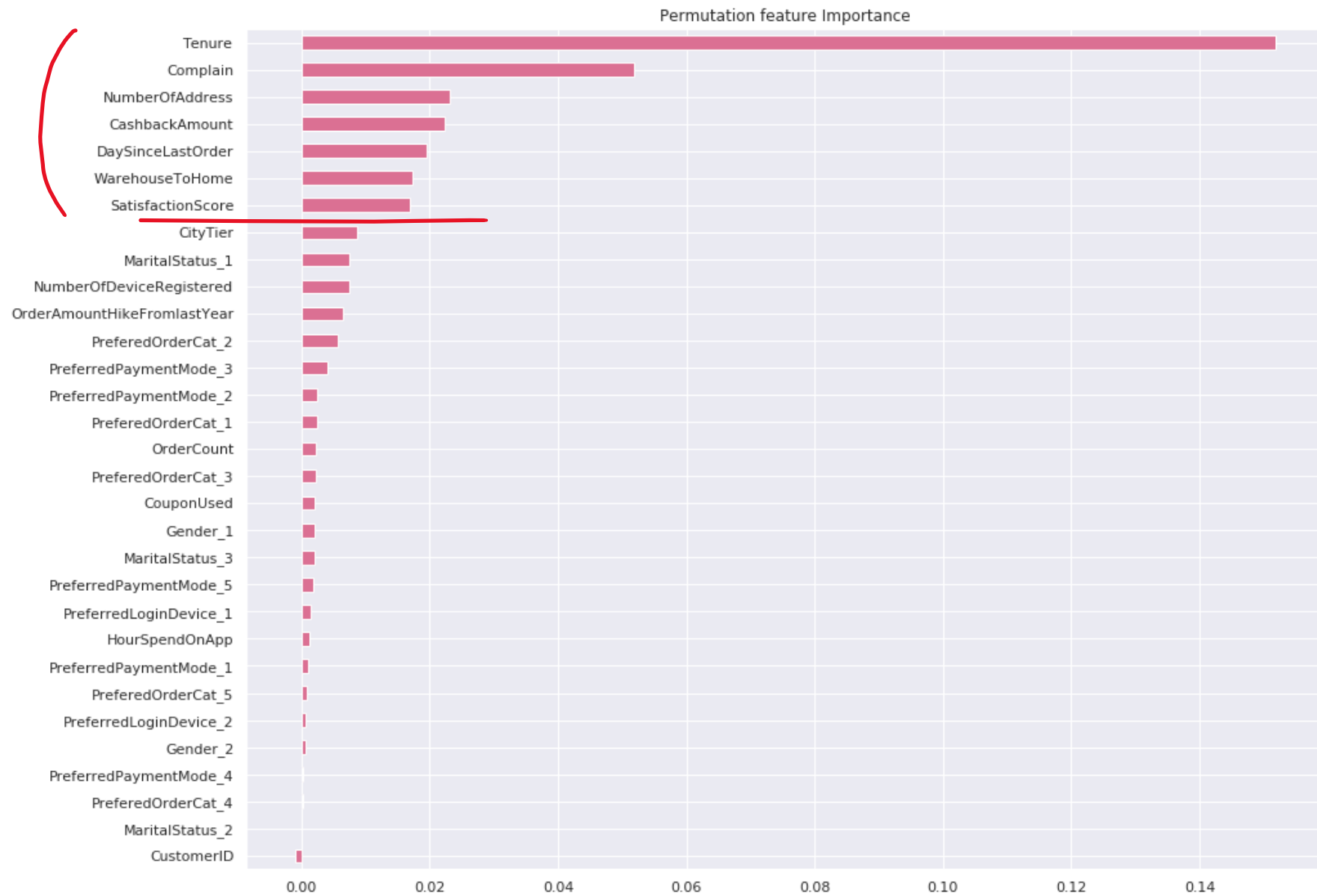
## < Oversampling - XGBoost >

```
# oversampling 시 동일한 비율로 샘플링  
y_train_sampled.value_counts()  
  
0    3741  
1    3741  
Name: Churn, dtype: int64
```

	precision	recall	f1-score	support
0	0.96	0.99	0.98	941
1	0.94	0.82	0.88	185
accuracy			0.96	1126
macro avg	0.95	0.90	0.93	1126
weighted avg	0.96	0.96	0.96	1126



- XGBoost 모델 정확도 : 0.96
- AUC : 0.98
- Test score for XGBoost: 0.8753623188405797 (0.89)
- Training score for XGBoost: 0.998032786885246 (0.99)
- ☆ 최종 모델



Part 5

결론

## < 결 론 >

- 이커머스 회사 이탈 예정 고객을 예방하고자 데이터 분석 진행
- 타겟 : 이탈 고객 (이탈하지 않은 고객 - 83%, 이탈한 고객 17%)
- 컴플레인에 따른 고객 이탈률 : 컴플레인을 건 쪽과 걸지 않은 쪽의 이탈률 차이가 큼 --> 컴플레인을 건 고객의 이탈률이 3배 더 높음
- 고객 가입 기간이 적을수록 이탈률이 매우 크게 나타남
- 즉, 이탈률에 가장 큰 영향을 미치는 변수는 가입 기간과 컴플레인인 것을 알 수 있음
- 중복값 X
- 결측치 각 변수 평균값으로 처리
- train data 80%, test data 20%
- 원핫인코딩 후 데이터 : 4504 행, 35 컬럼

## < 결 론 >

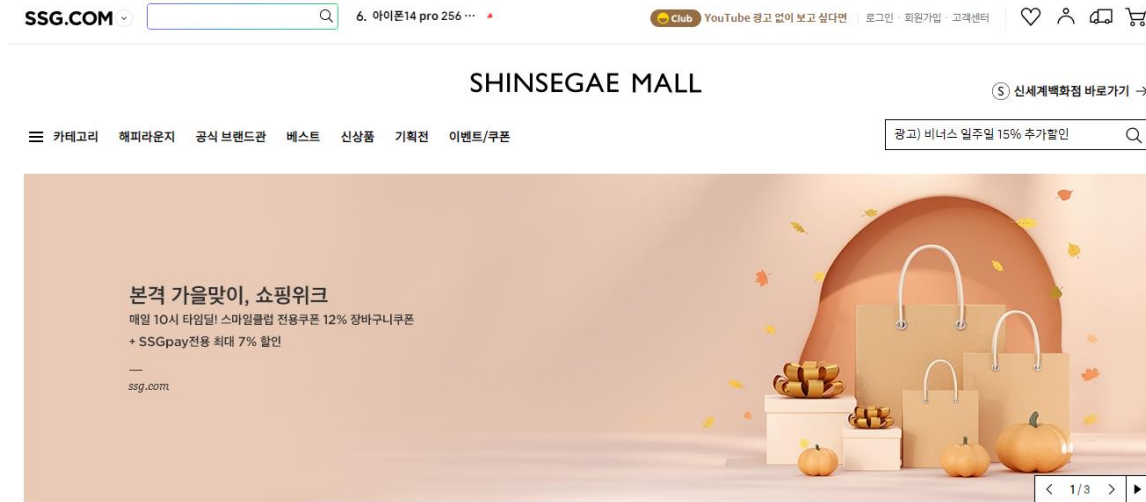
- 기준모델 정확도 : 0.83 / 로지스틱 회귀 정확도 : 0.82 / 결정트리 정확도 : 0.87 / 랜덤 포레스트 정확도 : 0.87 / XGBoost 정확도 : 0.87
- 로지스틱 회귀 test score : 0.6
- 결정트리 test score : 0.65
- 랜덤 포레스트 test score : 0.65
- XGBoost → 최적 하이퍼파라미터 찾기 위해 RandomSearchCV → 데이터 불균형 막기 위해 Undersampling → 데이터 수가 너무 감소되는 현상이 발생하여 다시 Oversampling
- 최종 XGBoost test score : 0.89
- permutation feature importance(순열 특성 중요도) : 1. Tenure / 2. Complain / 3. Number of address / 4. Cashback amount / 5. Day since last order / 6. Warehouse to home / 7. Satisfaction score

## < 이탈 예방 전략 >



- ✓ **Tenure**
- 가입 기간이 짧은 고객(~2개월)은 그 이상 가입 고객보다 이탈 가능성이 매우 높음
- 2개월 미만 고객에게 **가입 환영 쿠폰**을 지급하여 구매 유도 전략 (회사 유지 위해 최소 주문 금액, 최대 할인 금액 설정)
- ✓ **Complain**
- 컴플레인으로 인한 이탈률(31.67%)은 컴플레인을 걸지 않은 고객의 전체 이탈률(10.93%)보다 3배 높음
- **컴플레인 마케팅** 도입 : 컴플레인 고객을 매달 확인하여 고객의 불만 사항을 개선하고, 올바른 지적을 해준 고객 중 추첨을 통해 상품권을 증정하여 고객 충성도를 높이는 전략

## < 이탈 예방 전략 >



- ✓ Number Of Address
  - 주소를 입력하지 않은 고객은 주소를 입력한 고객보다 이탈 가능성이 더 높음
  - 회원가입 할 때 주소를 입력하게 유도하고, 로그인 하지 않은 고객은 이벤트를 미리보기만 가능하게 (참여하지 못하게) 하여 회원가입 유도하는 전략
- ✓ Cashback Amount
  - 지난 달 캐쉬백이 많이 쌓이지 않은 고객은 그렇지 않은 고객보다 이탈 가능성이 높음
  - 핸드폰, 랩탑 등 가격대가 높은 인기 상품을 구매하면 10% 캐쉬백이 쌓이는 기간 한정 이벤트 (ex. 새학기 기간) 통해 구매를 유도하고, 포인트를 평소보다 많이 쌓게 함으로써 회원 탈퇴를 방지하는 전략

감 사 합 니 다