여행상품 신청 여부 예측

22-2 SookTat Project

Team2
고나경 임주영 조민영
정재원 최호경

01 주제 및 데이터 소개

02 EDA

03 데이터 전처리

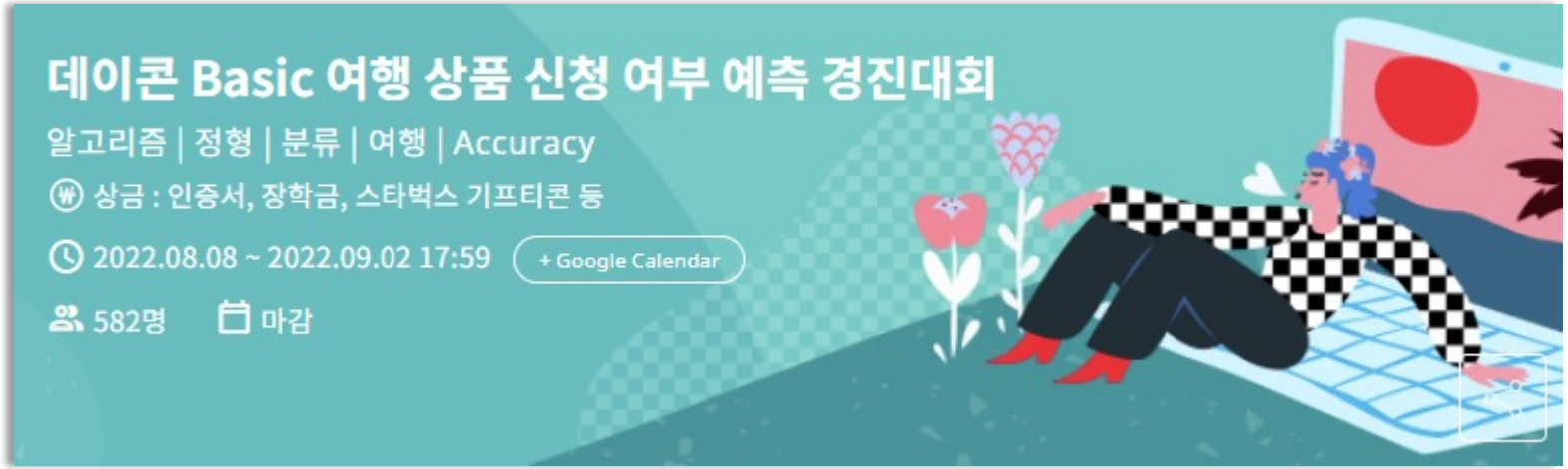
04 Modeling

01 주제 및 데이터 소개

주제

: 여행 상품 신청 여부 예측

https://dacon.io/competitions/official/235959/overview/description



- 나이, 성별, 월 수입 등의 고객 데이터를 이용해 여행 상품 신청 여부를 예측

데이터 소개

: train.csv

	id	Age	TypeofContact	CityTier	DurationOfPitch	Occupation	Gender	NumberOfPersonVisiting	NumberOfFollowups	ProductPitched	PreferredPropertyStar	MaritalStatus	NumberOfTrips	Passport	PitchSatisfactionScore	OwnCar	NumberOfChildrenVisiting	Designation	MonthlyIncome	ProdTaken
0	1	28.0	Company Invited	1	10.0	Small Business	Male	3	4.0	Basic	3.0	Married	3.0	0	1	0	1.0	Executive	20384.0	0
1	2	34.0	Self Enquiry	3	NaN	Small Business	Female	2	4.0	Deluxe	4.0	Single	1.0	1	5	1	0.0	Manager	19599.0	1
2	3	45.0	Company Invited	1	NaN	Salaried	Male	2	3.0	Deluxe	4.0	Married	2.0	0	4	1	0.0	Manager	NaN	0
3	4	29.0	Company Invited	1	7.0	Small Business	Male	3	5.0	Basic	4.0	Married	3.0	0	4	0	1.0	Executive	21274.0	1
4	5	42.0	Self Enquiry	3	6.0	Salaried	Male	2	3.0	Deluxe	3.0	Divorced	2.0	0	3	1	0.0	Manager	19907.0	0
5	6	32.0	Self Enquiry	1	29.0	Small Business	Male	4	4.0	Deluxe	3.0	Divorced	3.0	1	5	1	1.0	Manager	24857.0	1
6	7	43.0	Company Invited	3	8.0	Salaried	Male	3	3.0	Deluxe	3.0	Married	2.0	0	3	1	2.0	Manager	20675.0	0
7	8	32.0	Self Enquiry	3	20.0	Small Business	Male	4	5.0	Deluxe	5.0	Married	7.0	1	1	1	1.0	Manager	20980.0	1
8	9	36.0	Company Invited	3	NaN	Small Business	Female	2	1.0	Deluxe	5.0	Divorced	3.0	0	1	1	0.0	Manager	19639.0	0
9	10	34.0	Self Enquiry	1	7.0	Salaried	Male	4	4.0	Basic	3.0	Unmarried	3.0	1	3	1	1.0	Executive	21364.0	1
10	11	35.0	Company Invited	1	14.0	Salaried	Male	4	6.0	Deluxe	3.0	Unmarried	3.0	1	5	1	1.0	Manager	24752.0	1
11	12	31.0	Self Enquiry	1	9.0	Small Business	Fe Male	3	5.0	Deluxe	3.0	Unmarried	7.0	1	2	1	2.0	Manager	25555.0	0
12	13	49.0	Company Invited	3	14.0	Small Business	Female	4	4.0	Basic	3.0	Married	4.0	1	4	1	2.0	Executive	21333.0	1
13	14	NaN	Self Enquiry	3	6.0	Small Business	Male	2	1.0	Deluxe	5.0	Married	2.0	0	4	0	0.0	Manager	NaN	0
14	15	52.0	Company Invited	3	16.0	Salaried	Male	3	4.0	King	NaN	Married	6.0	1	4	1	2.0	VP	38525.0	0
15	16	28.0	Self Enquiry	1	15.0	Small Business	Male	3	3.0	Basic	3.0	Married	2.0	0	1	1	0.0	Executive	17070.0	0
16	17	28.0	Self Enquiry	1	23.0	Large Business	Male	2	4.0	Basic	3.0	Married	6.0	0	3	0	1.0	Executive	17367.0	1
17	18	33.0	Self Enquiry	1	9.0	Large Business	Male	4	4.0	Basic	5.0	Single	3.0	0	1	1	2.0	Executive	21117.0	0
18	19	36.0	Self Enquiry	1	8.0	Salaried	Female	3	3.0	Basic	3.0	Married	5.0	0	5	1	0.0	Executive	17543.0	0
19	20	22.0	Self Enquiry	1	21.0	Small Business	Female	2	3.0	Basic	3.0	Single	2.0	0	1	1	1.0	Executive	17871.0	0
20	21	33.0	Self Enquiry	1	20.0	Small Business	Female	3	3.0	Basic	4.0	Married	2.0	0	5	1	1.0	Executive	17756.0	0

1955 rows × 20 columns

데이터 소개

: test.csv

	id	Age	TypeofContact	CityTier	DurationOfPitch	Occupation	Gender	NumberOfPersonVisiting	NumberOfFollowups	ProductPitched	PreferredPropertyStar	MaritalStatus	NumberOfTrips	Passport	PitchSatisfactionScore	OwnCar	NumberOfChildrenVisiting	Designation	MonthlyIncome
0	1	32.0	Company Invited	3	NaN	Small Business	Male	2	5.0	Deluxe	3.0	Married	1.0	0	2	0	1.0	Manager	19668.0
1	2	46.0	Self Enquiry	2	11.0	Small Business	Male	3	NaN	Deluxe	4.0	Married	1.0	1	5	0	1.0	Manager	20021.0
2	3	37.0	Self Enquiry	3	22.0	Small Business	Male	3	4.0	Deluxe	3.0	Married	5.0	0	5	1	0.0	Manager	21334.0
3	4	43.0	Self Enquiry	1	36.0	Small Business	Male	3	6.0	Deluxe	3.0	Unmarried	6.0	0	3	1	2.0	Manager	22950.0
4	5	25.0	Self Enquiry	3	7.0	Large Business	Female	4	4.0	Basic	4.0	Unmarried	3.0	1	4	1	3.0	Executive	21880.0
5	6	40.0	Self Enquiry	1	22.0	Salaried	Female	2	3.0	Standard	3.0	Unmarried	7.0	1	4	1	0.0	Senior Manager	22945.0
6	7	55.0	Company Invited	1	8.0	Salaried	Male	3	3.0	Standard	4.0	Divorced	4.0	0	2	1	1.0	Senior Manager	25976.0
7	8	24.0	Self Enquiry	1	6.0	Small Business	Male	3	3.0	Basic	3.0	Married	3.0	1	3	0	2.0	Executive	17293.0
8	9	38.0	Self Enquiry	1	29.0	Salaried	Male	2	3.0	Deluxe	3.0	Married	1.0	0	3	0	0.0	Manager	20745.0
9	10	33.0	Self Enquiry	1	9.0	Large Business	Male	3	5.0	Deluxe	5.0	Single	6.0	0	4	0	2.0	Manager	20854.0
10	11	55.0	Self Enquiry	1	12.0	Small Business	Male	3	4.0	King	5.0	Divorced	NaN	0	4	1	1.0	VP	38084.0
11	12	47.0	Self Enquiry	1	7.0	Small Business	Male	3	4.0	King	NaN	Married	2.0	0	5	1	2.0	VP	38305.0
12	13	30.0	Company Invited	1	9.0	Small Business	Female	3	3.0	Basic	3.0	Married	2.0	0	3	1	1.0	Executive	17083.0
13	14	40.0	Self Enquiry	1	13.0	Small Business	Male	4	4.0	Basic	5.0	Divorced	2.0	1	2	1	2.0	Executive	21082.0
14	15	52.0	Self Enquiry	3	17.0	Salaried	Female	4	4.0	Standard	4.0	Married	7.0	0	1	1	3.0	Senior Manager	31820.0
15	16	20.0	Self Enquiry	1	9.0	Salaried	Male	2	4.0	Basic	3.0	Single	2.0	0	3	0	1.0	Executive	18033.0
16	17	38.0	Self Enquiry	1	15.0	Salaried	Female	3	3.0	Basic	3.0	Single	2.0	0	2	1	1.0	Executive	17288.0
17	18	37.0	Self Enquiry	3	17.0	Small Business	Male	3	5.0	Standard	5.0	Married	2.0	0	5	0	1.0	Senior Manager	25772.0
18	19	47.0	Self Enquiry	3	7.0	Small Business	Female	4	4.0	Standard	5.0	Married	3.0	0	1	1	3.0	Senior Manager	29131.0
19	20	31.0	Company Invited	1	10.0	Small Business	Female	4	4.0	Basic	3.0	Married	3.0	0	3	1	2.0	Executive	20761.0
20	21	NaN	Self Enquiry	1	8.0	Small Business	Male	2	5.0	Basic	3.0	Married	6.0	1	3	1	1.0	Executive	18464.0

2933 rows × 19 columns

데이터 소개

: 변수

변수	의미
id	샘플 아이디
Age	나이
TypeofContact	고객의 제품 인지 방법
CityTier	주거 중인 도시의 등급
DurationOfPitch	영업 사원이 고객에게 제공하는 프레젠테이션 기간
Occupation	직업
Gender	성별
NumberOf PersonVisiting	고객과 함께 여행을 계획 중인 총 인원
NumberOf Followups	영업 사원의 프레젠테이션 후 이루어진 후속 조치 수
ProductPitched	영업 사원이 제시한 상품

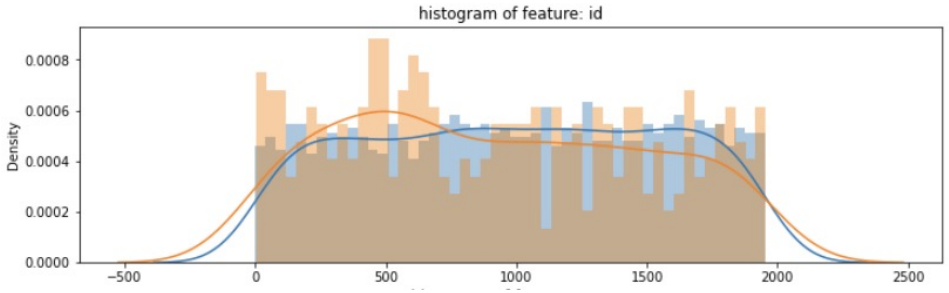
변수	의미
PreferredPropertyStar	선호 호텔 숙박업소 등급
MaritalStatus	결혼여부
NumberOfTrips	평균 연간 여행 횟수
Passport	여권 보유 여부
PitchSatisfactionScore	영업 사원의 프레젠테이션 만족도
OwnCar	자동차 보유 여부
NumberOf ChildrenVisiting	함께 여행을 계획 중인 5세 미만의 어린이 수
Designation	(직업의) 직급
MonthlyIncome	월 급여
ProdTaken	여행 패키지 신청 여부

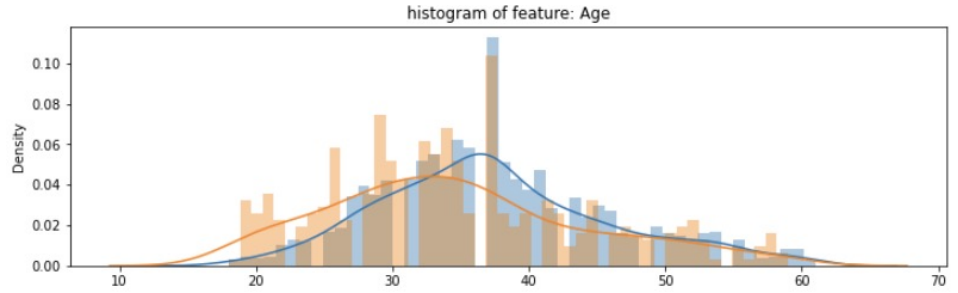
02 EDA

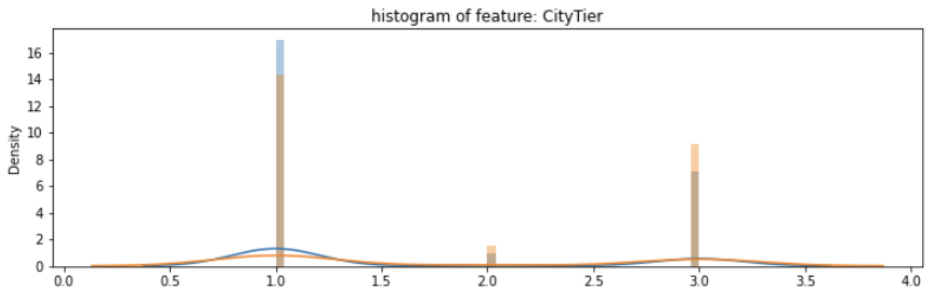
EDA

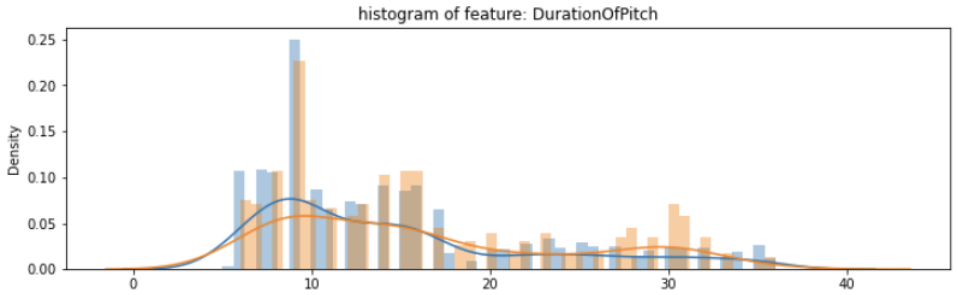
: Protaken의 값 (0,1)에 따른 분포 확인

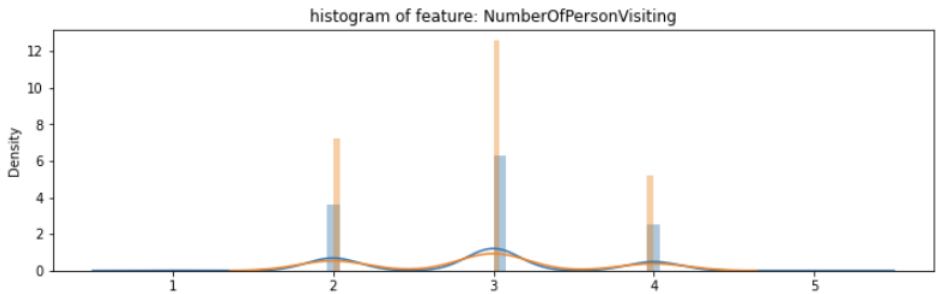
- 0의 값을 가지나 1의 값을 가지나 비슷한 분포를 가지고 있다는 것을 확인함.

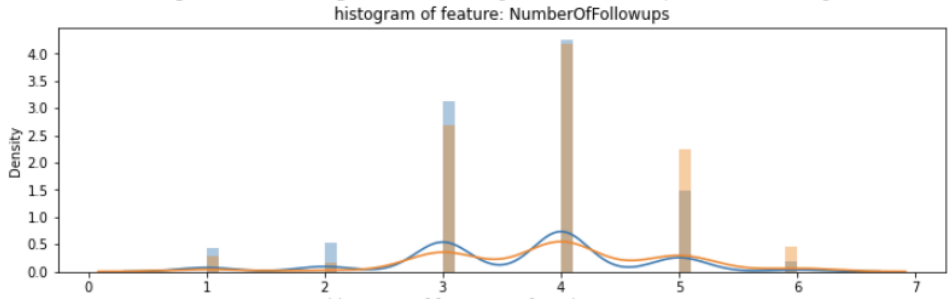










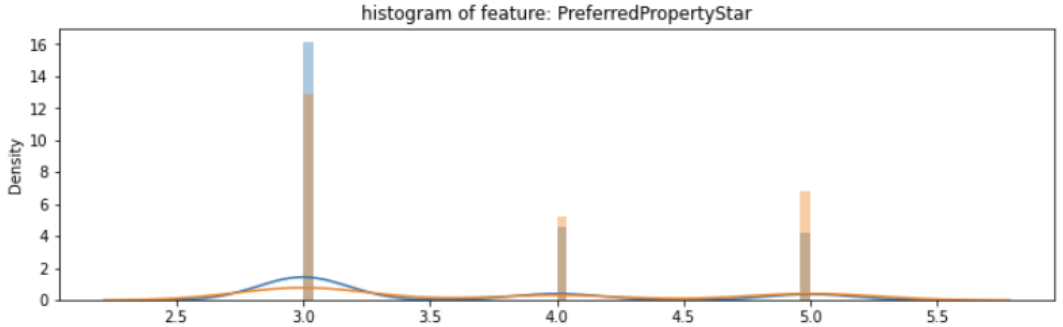


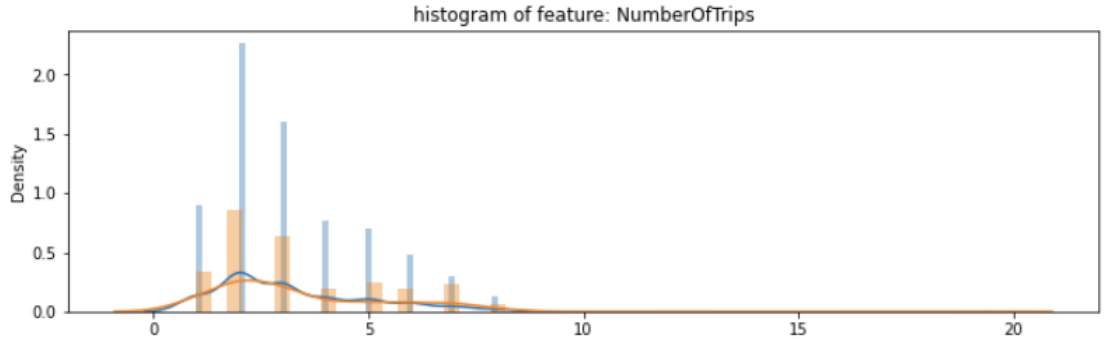
● : 1 (여행 패키지 신청 o)
● : 0 (여행 패키지 신청 x)

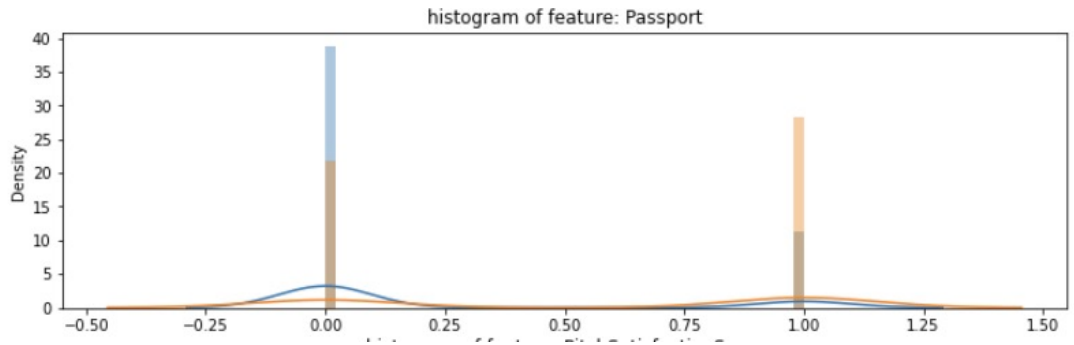
EDA

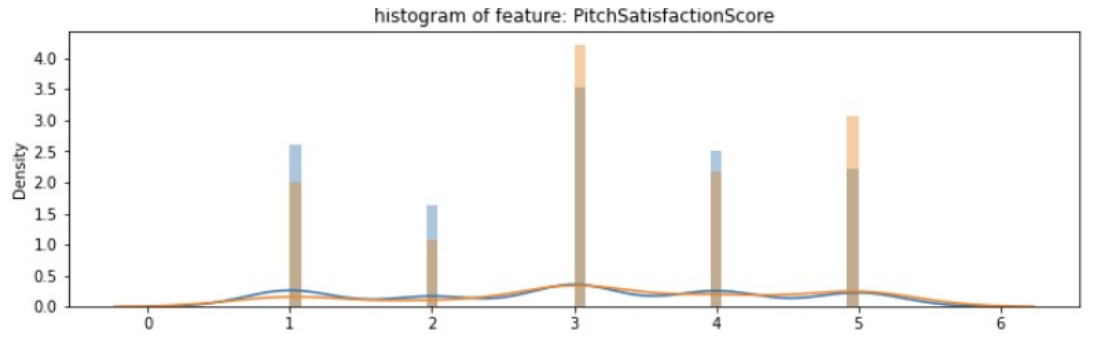
: Protaken의 값 (0,1)에 따른 분포 확인

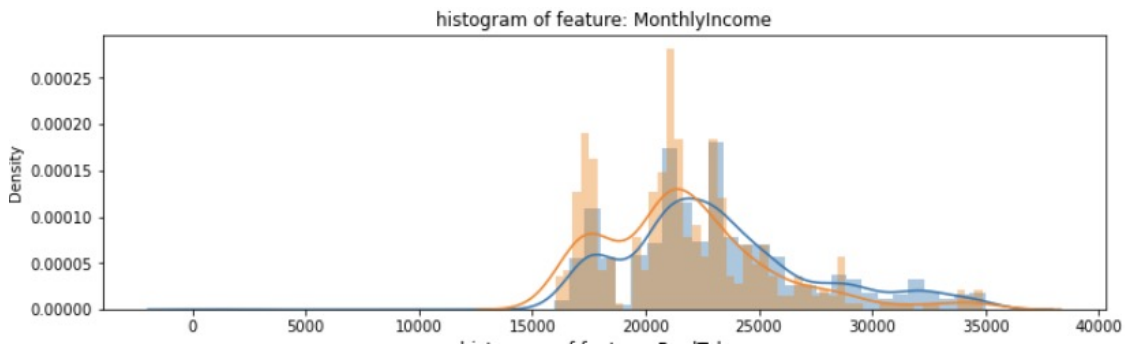
● : 1 (여행 패키지 신청 o)
● : 0 (여행 패키지 신청 x)

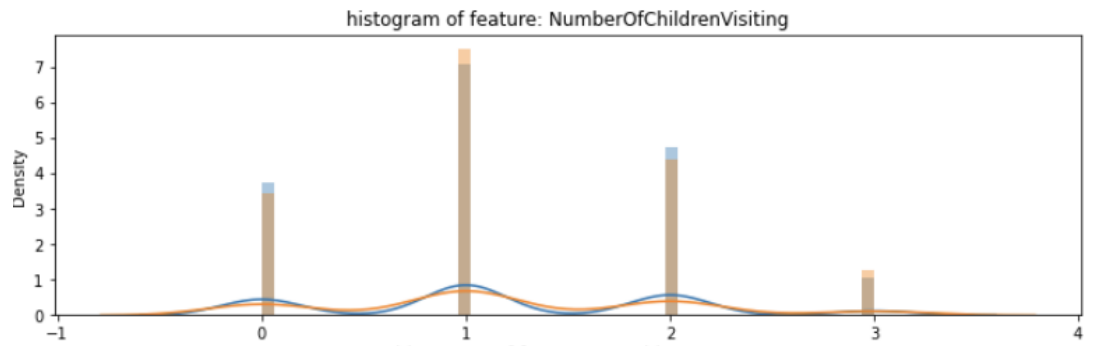






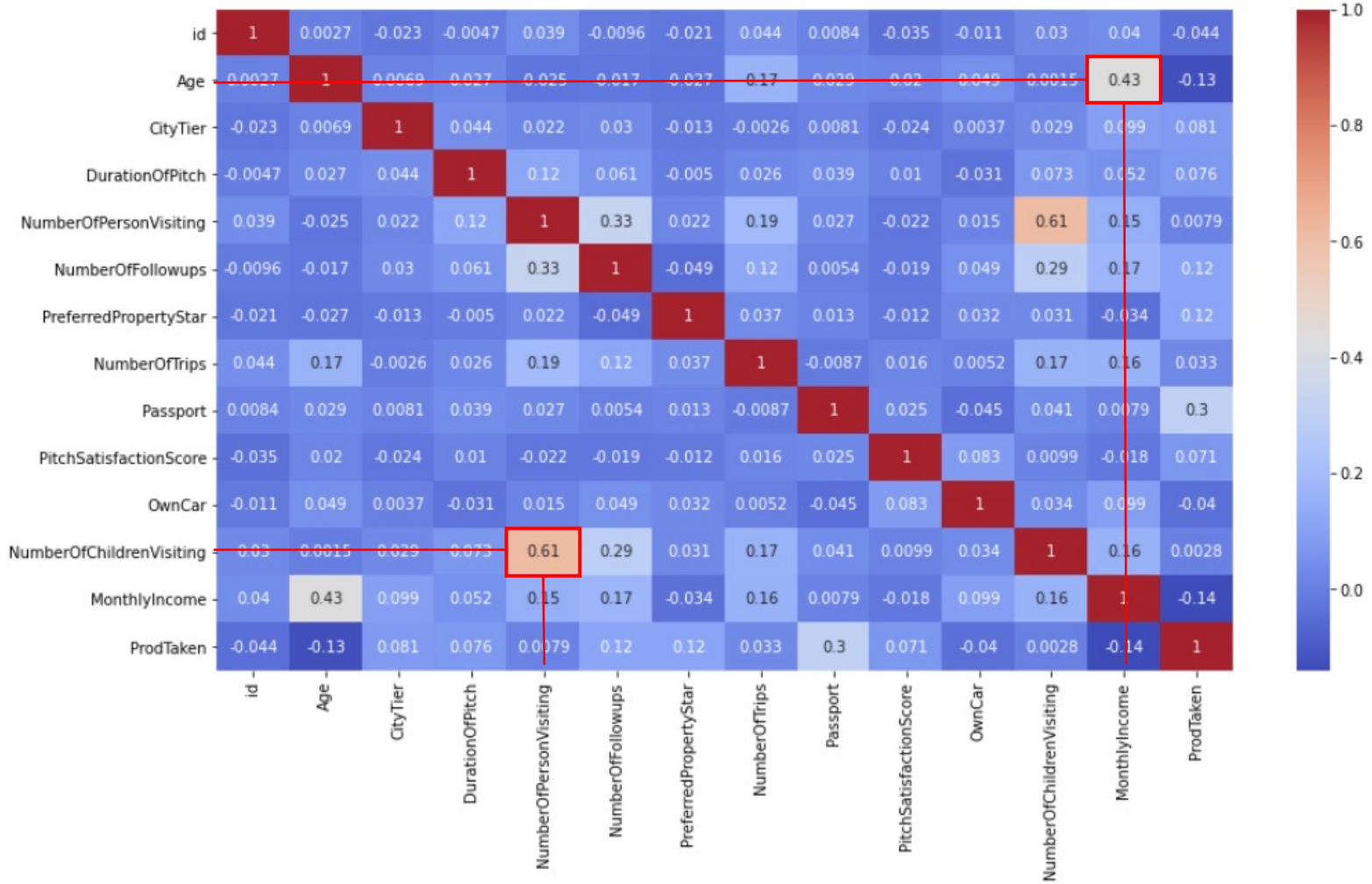






EDA

: 상관관계 파악


id 1 0.0027 -0.023 -0.0047 0.039 -0.0096 -0.021 0.044 0.0084 -0.035 -0.011 0.03 0.04 -0.044
Age 0.0027 1 0.0069 0.027 0.025 0.017 0.027 0.17 0.029 0.02 0.049 0.0015 0.43 -0.13
CityTier -0.023 0.0069 1 0.044 0.022 0.03 -0.013 -0.0026 0.0081 -0.024 0.0037 0.029 0.099 0.081
DurationOfPitch -0.0047 0.027 0.044 1 0.12 0.061 -0.005 0.026 0.039 0.01 -0.031 0.073 0.052 0.076
NumberOfPersonVisiting 0.039 -0.025 0.022 0.12 1 0.33 0.022 0.19 0.027 -0.022 0.015 0.61 0.15 0.0079
NumberOfFollowups -0.0096 -0.017 0.03 0.061 0.33 1 -0.049 0.12 0.0054 -0.019 0.049 0.29 0.17 0.12
PreferredPropertyStar -0.021 -0.027 -0.013 -0.005 0.022 -0.049 1 0.037 0.013 -0.012 0.032 0.031 -0.034 0.12
NumberOfTrips 0.044 0.17 -0.0026 0.026 0.19 0.12 0.037 1 -0.0087 0.016 0.0052 0.17 0.16 0.033
Passport 0.0084 0.029 0.0081 0.039 0.027 0.0054 0.013 -0.0087 1 0.025 -0.045 0.041 0.0079 0.3
PitchSatisfactionScore -0.035 0.02 -0.024 0.01 -0.022 -0.019 -0.012 0.016 0.025 1 0.083 0.0099 -0.018 0.071
OwnCar -0.011 0.049 0.0037 -0.031 0.015 0.049 0.032 0.0052 -0.045 0.083 1 0.034 0.099 -0.04
NumberOfChildrenVisiting 0.03 0.0015 0.029 0.073 0.61 0.29 0.031 0.17 0.041 0.0099 0.034 1 0.16 0.0028
MonthlyIncome 0.04 0.43 0.099 0.052 0.15 0.17 -0.034 0.16 0.0079 -0.018 0.099 0.16 1 -0.14
ProdTaken -0.044 -0.13 0.081 0.076 0.0079 0.12 0.12 0.033 0.3 0.071 -0.04 0.0028 -0.14 1
id
Age
CityTier
DurationOfPitch
NumberOfPersonVisiting
NumberOfFollowups
PreferredPropertyStar
NumberOfTrips
Passport
PitchSatisfactionScore
OwnCar
NumberOfChildrenVisiting
MonthlyIncome
ProdTaken
1.0
0.8
0.6
0.4
0.2
0.0

03 데이터 전처리

데이터 전처리

: Gender 변수

Gender 변수에서 ‘Female’에 해당하는 데이터와
‘Fe Male’에 해당하는 데이터 존재

```
train['Gender'].value_counts()
```

```
Male       1207
Female      692
Fe Male      56
Name: Gender, dtype: int64
```

‘Fe Male’ 데이터와 ‘Female’ 데이터 병합

```
# Fe Male을 Female로 변경

train.loc[train['Gender']=='Fe Male','Gender']= 'Female'
```

```
train['Gender'].value_counts()
```

```
Male      1207
Female     748
Name: Gender, dtype: int64
```

데이터 전처리

: 중복 데이터

중복 데이터

```
train.duplicated()
```

```
0       False
1       False
2       False
3       False
4       False
        ...
1950    False
1951    False
1952    False
1953    False
1954    False
Length: 1955, dtype: bool
```

```
# 중복 데이터 없음

train[train.duplicated()]
```

id	Age	TypeofContact	CityTier	DurationOfPitch	Occupation	Gender	NumberOfPersonVisiting	NumberOfFollowups	ProductPitched	PreferredPropertyStar

→ 중복 데이터 존재하지 않음

데이터 전처리

: 결측치 확인

결측치

```
# 결측치 확인

train.isnull().sum()
```

```
id                          0
Age                        94
TypeofContact              10
CityTier                    0
DurationOfPitch           102
Occupation                  0
Gender                      0
NumberOfPersonVisiting      0
NumberOfFollowups          13
ProductPitched              0
PreferredPropertyStar      10
MaritalStatus               0
NumberOfTrips              57
Passport                    0
PitchSatisfactionScore      0
OwnCar                      0
NumberOfChildrenVisiting   27
Designation                 0
MonthlyIncome             100
ProdTaken                   0
dtype: int64
```

변수	자료형	결측치 수
Age	Float	94
DurationOfPitch	Float	102
NumberOfFollowups	Float	13
PreferredPropertyStar	Float	10
NumberOfTrips	Float	57
NumberOfChildrenVisiting	Float	27
MonthlyIncome	Float	100
TypeofContact	Object	10

처리 방법
평균값 37로 대체
최빈값 9로 대체
최빈값 4로 대체
최빈값 3으로 대체
평균값 3으로 대체
최빈값 1로 대체
평균값으로 대체
‘Unknown’으로 대체

데이터 전처리

: 이상치

```
def outlier_iqr(data, column):

    # lower, upper 글로벌 변수 선언하기
    global lower, upper

    # 4분위수 기준 지정하기
    q25, q75 = np.quantile(data[column], 0.25), np.quantile(data[column], 0.75)

    # IQR 계산하기
    iqr = q75 - q25

    # outlier cutoff 계산하기
    cut_off = iqr * 1.5

    # lower와 upper bound 값 구하기
    lower, upper = q25 - cut_off, q75 + cut_off

    print('IQR은',iqr, '이다.')
    print('lower bound 값은', lower, '이다.')
    print('upper bound 값은', upper, '이다.')

    # 1사분위와 4사분위 데이터
    data1 = data[data[column] > upper]
    data2 = data[data[column] < lower]

    # 이상치 총 개수
    return print('총 이상치 수', data1.shape[0] + data2.shape[0])
```

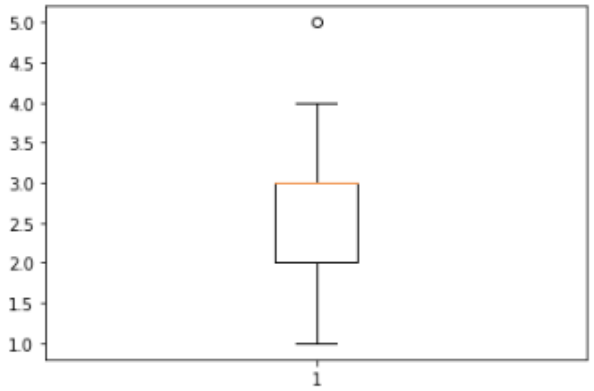
```
: # 결측치 처리한 데이터로 이상치 확인

for i in range(len(col_num)):
    print()
    print('변수명: %s' %col_num[i])
    outlier_iqr(train_im, col_num[i])
```

```
변수명: id
IQR은 974.0 이다.
lower bound 값은 -970.0 이다.
upper bound 값은 2926.0 이다.
총 이상치 수 0

변수명: Age
IQR은 12.0 이다.
lower bound 값은 13.0 이다.
upper bound 값은 61.0 이다.
총 이상치 수 0

변수명: CityTier
IQR은 2.0 이다.
lower bound 값은 -2.0 이다.
upper bound 값은 6.0 이다.
총 이상치 수 0
```

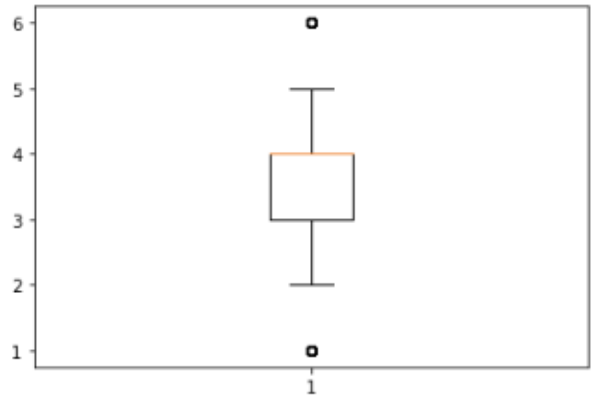
데이터 전처리

: 이상치

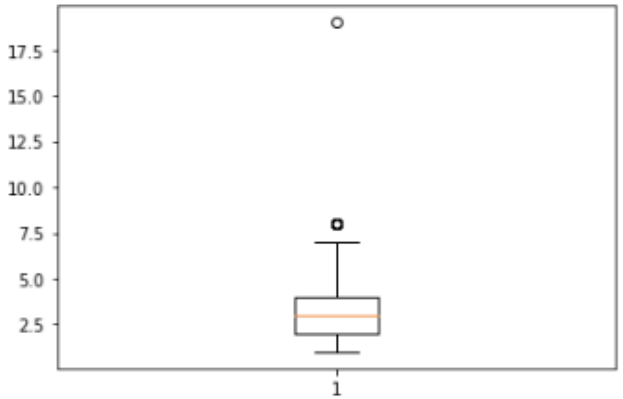
1) NumberOfPersonVisiting



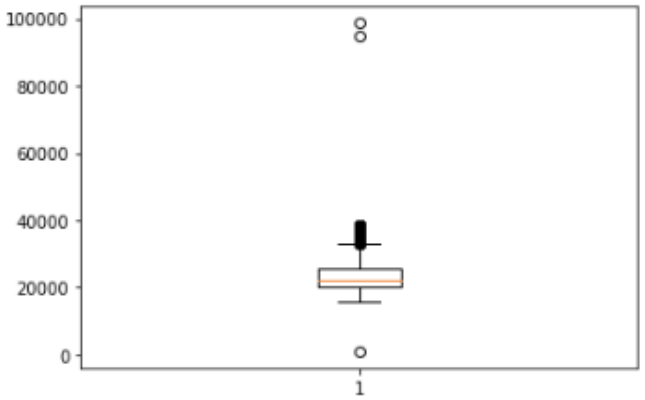
2) NumberOfFollowups



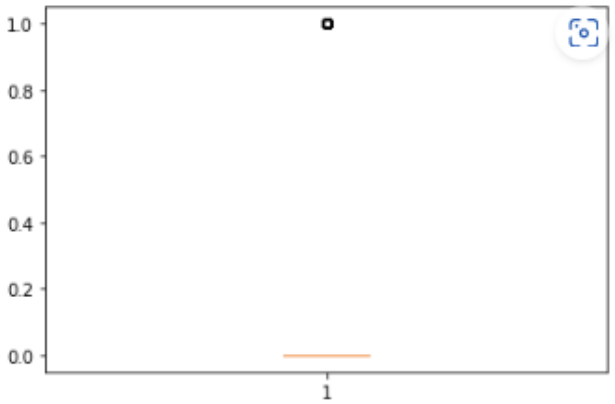
3) NumberOfTrips



4) MonthlyIncome



5) ProdTaken



* 변수 별 이상치 확인 결과
논리적으로 가능한 값이기에
이상치 처리는 하지 않음

데이터 전처리

: Normalization, LabelEncoder

1. Min-Max Normalization

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

min-max normaliztion 전

	Age	DurationOfPitch	MonthlyIncome
0	28.0	10.0	20384.000000
1	34.0	9.0	19599.000000
2	45.0	9.0	23624.108895
3	29.0	7.0	21274.000000
4	42.0	6.0	19907.000000
...	...	...	...
1950	28.0	10.0	20723.000000
1951	41.0	8.0	31595.000000
1952	38.0	28.0	21651.000000
1953	28.0	30.0	22218.000000
1954	22.0	9.0	17853.000000

min-max normaliztion 후

	Age	DurationOfPitch	MonthlyIncome
0	0.232558	0.161290	0.570134
1	0.372093	0.129032	0.547046
2	0.627907	0.129032	0.665435
3	0.255814	0.064516	0.596312
4	0.558140	0.032258	0.556105
...	...	...	...
1950	0.232558	0.161290	0.580105
1951	0.534884	0.096774	0.899879
1952	0.465116	0.741935	0.607400
1953	0.232558	0.806452	0.624077
1954	0.093023	0.129032	0.495691

2. LabelEncoder

	TypeofContact	Occupation	Gender	ProductPitched	MaritalStatus	Designation
0	Company Invited	Small Business	Male	Basic	Married	Executive
1	Self Enquiry	Small Business	Female	Deluxe	Single	Manager
2	Company Invited	Salaried	Male	Deluxe	Married	Manager
3	Company Invited	Small Business	Male	Basic	Married	Executive
4	Self Enquiry	Salaried	Male	Deluxe	Divorced	Manager
...	...	...	...	...	...	...
1950	Self Enquiry	Small Business	Male	Basic	Single	Executive
1951	Self Enquiry	Salaried	Female	Super Deluxe	Divorced	AVP
1952	Company Invited	Small Business	Female	Basic	Divorced	Executive
1953	Self Enquiry	Small Business	Female	Deluxe	Married	Manager
1954	Company Invited	Salaried	Male	Basic	Divorced	Executive

	TypeofContact	Occupation	Gender	ProductPitched	MaritalStatus	Designation
0	0	3	2	0	1	1
1	1	3	1	1	2	2
2	0	2	2	1	1	2
3	0	3	2	0	1	1
4	1	2	2	1	0	2
...	...	...	...	...	...	...
1950	1	3	2	0	2	1
1951	1	2	1	4	0	0
1952	0	3	1	0	0	1
1953	1	3	1	1	1	2
1954	0	2	2	0	0	1

04 Modeling

Boosting Model

- 트리 앙상블 모델
- 만들어지는 트리가 이전에 만들어진 트리에 영향을 받음
- 약한 모델이 잘못 예측한 sample(error)을 점점 강화함
- 잘못 분류된 샘플에 가중치를 줘서 더 잘 보이게 해줌

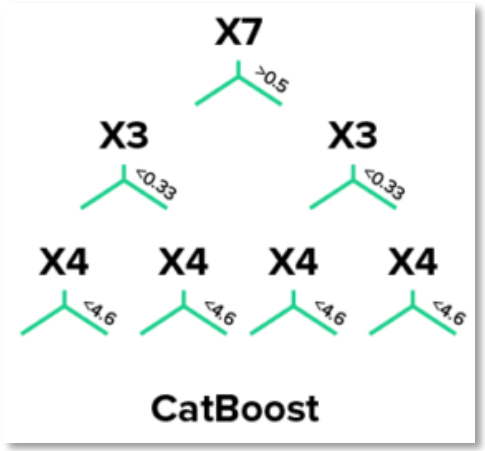
01. CatBoost Classifier

02. XGBoost Classifier

03. Ada Boost Classifier

Modeling

1. CatBoost Classifier



Cat Boost

- Level-wise Tree(대칭적) 방식으로 분할
- 일부 데이터를 대상으로 잔차 계산 후 모델 생성하고, 생성한 모델로 다음 잔차 예측
- 위의 과정을 반복할 때 데이터를 섞어주지 않으면 매번 같은 순서로 잔차 예측
- Cat Boost는 이를 감안해 셔플링을 통해 데이터를 뽑음

이미지 출처 https://heeya-stupidbutstudying.tistory.com/entry/ML-CatBoost-%EC%86%8C%EA%B0%9C-Ordered-Boosting-%EB%B2%94%EC%A3%BC%ED%98%95-%ED%94%BC%EC%B2%98-%EC%B2%98%EB%A6%AC

Cat Boost 파라미터

Cat Boost는 기본적으로 파라미터가 최적화가 잘 되어있어서,
파라미터 튜닝에 크게 신경쓰지 않아도 된다.

- **has_time**

시간이 지나면 변화하는 데이터 방지 True로 설정

- **fold_len_multiplier**

작은 데이터셋의 한계 극복; 1로 설정

- **approx_on_full_history**

작은 데이터셋의 한계 극복; True로 설정

- **custom_metric**

Error 모니터링; 'AUG' or 'Logloss'

- **ord_type**

Iter로 설정하면 교차검증 시 Early Stopping 가능

- **class_weights**

클래스 불균형 문제 해소

Modeling

1. CatBoost Classifier

```
import catboost as ctb

model_CBC = ctb.CatBoostClassifier()
model_CBC.fit(train_X_scaled, train_y)
```

```
from sklearn import datasets
from sklearn import metrics

expected_y  = valid_y
predicted_y = model_CBC.predict(valid_X_scaled)
print(metrics.classification_report(expected_y, predicted_y))
```

```
              precision    recall  f1-score   support

           0       0.86      0.98      0.92       465
           1       0.85      0.41      0.55       122

    accuracy                           0.86       587
   macro avg       0.86      0.70      0.74       587
weighted avg       0.86      0.86      0.84       587
```

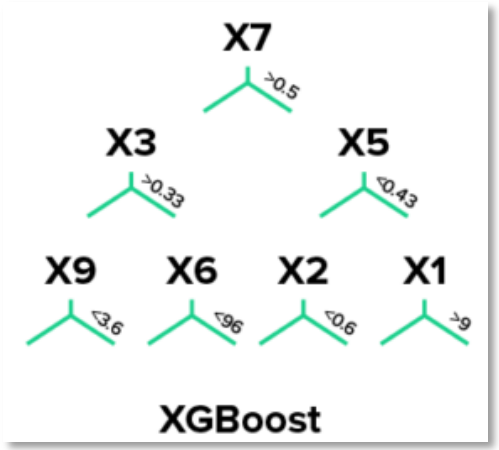
```
classificationSummary(valid_y, predicted_y)
```

```
Confusion Matrix (Accuracy 0.8620)

       Prediction
Actual   0   1
     0 456   9
     1  72  50
```

Modeling

2. XGBoost Classifier

→ 최종모델



XG Boost

- 이전 모델의 오류를 순차적으로 보완
- 반복 수행시마다 내부적인 교차검증을 수행해 최적화된 반복 수행횟수를 가질 수 있음
- (장점) GBM 대비 빠른 수행 시간, 과저합 규제, 트리 가지치기로 분할 수 줄임, 자체 내장된 교차 검증

이미지 출처 https://heeya-stupidbutstudying.tistory.com/entry/ML-CatBoost-%EC%86%8C%EA%B0%9C-Ordered-Boosting-%EB%B2%94%EC%A3%BC%ED%98%95-%ED%94%BC%EC%B2%98-%EC%B2%98%EB%A6%AC

XGBoost 파라미터

- **n_estimatiors** : 트리의 개수, 높을수록 정확도는 높아지나 시간이 오래 걸림
- **n_jobs** : 병렬처리 여부
- **random_state**: 결과를 고정시킴
- **max_depth** : 생성할 Decision Tree의 깊이
- **learning_rate** : 학습할 때 모델을 얼마나 업데이트 할 것인지
- **colsample_bytree** : 열 샘플링에 사용하는 비율
- **subsample**: 행 샘플링에 사용하는 비율
- **colsample_bytree** : 열 샘플링에 사용하는 비율
- **Reg_alpha** : L1 정규화 계수
- **Reg_lambda** : L2 정규화 계수
- **Booster** : 부스팅 방법, 주로 gbtree 이용

Modeling

2. XGBoost Classifier

→ 최종모델

```
# id 제외
train = train_na.drop(columns=['id'])
test = test_na.drop(columns=['id'])

# 분석 목적: prodTake 여행상품 예측
x_train = train.drop(columns=['ProdTaken'])
y_train = train[['ProdTaken']]
```

```
# 전체 데이터셋을 학습용 80%, 테스트용 20%로 분할
X_train, X_test, Y_train, Y_test = train_test_split(x_train, y_train, test_size=0.2, rando
```

```
#모델 정의
clf = XGBClassifier(learning_rate =0.1,
                    n_estimators=1000,
                    use_label_encoder=False,
                    random_state=42)

#learning_rate: 이전의 결과를 얼마나 반영할 것인가? 일반적으로 0.01~0.2
#n_estimator: 나무 개수
#use_label_encoder: 라벨 인코더 사용
#random_state : seed값 고정

clf.fit(X_train, Y_train, eval_metric='logloss')

Y_pred= clf.predict(X_test)
```

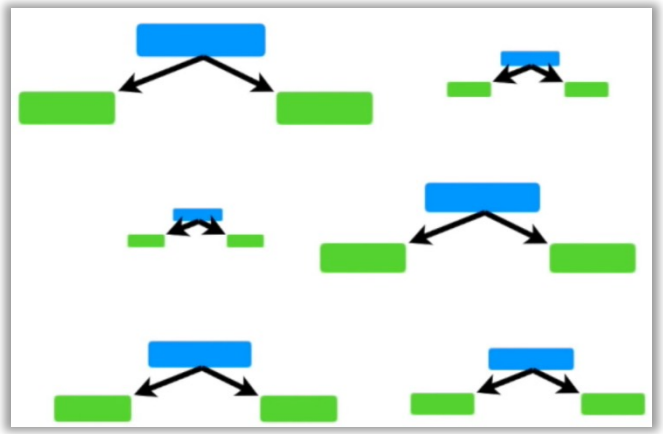
```
def get_clf_eval(y_test, pred):
    confusion = confusion_matrix(y_test, pred)
    accuracy = accuracy_score(y_test, pred)
    precision = precision_score(y_test, pred)
    recall = recall_score(y_test, pred)
    print('Confusion Matrix')
    print(confusion)
    print('정확도:{}, 정밀도:{}, 재현율:{}'.format(accuracy, precision, recall))
```

```
# 예측 결과 확인
get_clf_eval(Y_test, Y_pred)
```

```
Confusion Matrix
[[304  14]
 [ 30  43]]
정확도:0.887468030690537, 정밀도:0.7543859649122807, 재현율:0.589041095890411
```

Modeling

3. Ada Boost Classifier



Ada Boost

- 간단한 약분류기들이 상호보완하도록 단계적으로 학습, 이들을 조합해 최종 강분류기의 성능을 증폭시킨다.
- 약한 학습기를 순차적으로 학습시키고, 개별 학습기에 가중치를 부여하여 모두 결합함으로써 개별 약한 학습기보다 높은 정확도의 예측 결과를 만든다.

Ada Boost 파라미터

- **base_estimators** : 학습에 사용하는 알고리즘을 설정(Default = None); DecisionTreeClassifier(max_depth=1)가 적용됨
- **n_estimators** : 생성할 약한 학습기의 개수를 지정(Default = 50)
- **learning_rate** : 학습을 진행할 때마다 적용하는 학습률(Default = 1.0)

- n_estimators를 늘린다면
  - 생성하는 weak learner의 수는 늘어남
  - 이 여러 학습기들의 decision boundary가 많아지면서 모델이 복잡해짐
- learning_rate을 줄인다면
  - 가중치 갱신의 변동폭이 감소해서, 여러 학습기들의 decision boundary 차이가 줄어듦
- 위의 두 가지는 trade-off 관계
  → 이 두 파라미터를 잘 조정하는 것이 알고리즘의 핵심

Modeling

3. Ada Boost Classifier

```
from sklearn.ensemble import AdaBoostClassifier
abc = AdaBoostClassifier(n_estimators=50,  learning_rate=1)
```

```
from sklearn import metrics
model = abc.fit(X_train, y_train)
# Predict the response for test dataset
y_pred = model.predict(X_valid)
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_valid, y_pred))
```

```
C:₩Users₩rhskr₩anaconda3₩lib₩site-packages₩sklearn₩utils₩validation.py:1111: DataConversion
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
Accuracy: 0.84
```

Modeling

[Accuracy]

Cat Boost

0.86

2nd

XG Boost

0.89

1st

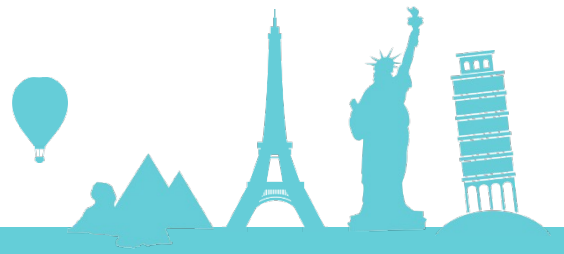
Ada Boost

0.84

3rd

[아쉬운 점]

- 3개의 모델 외에도 Logistic Regression, KNN, Naives Bayes, Decision Tree, SVM 등 모델을 적합시켰으나 정확도가 높지 않았다.
- 분류 모델에 대해 GridSearchCV, RandomSearchCV 등 하이퍼 파라미터를 튜닝하였으나 디폴트 값을 가졌을 때 정확도가 제일 높아 튜닝에서는 큰 도움을 받지 못해 아쉬움이 남는다.



Thank you

https://github.com/jyoung19/DACON_BASIC