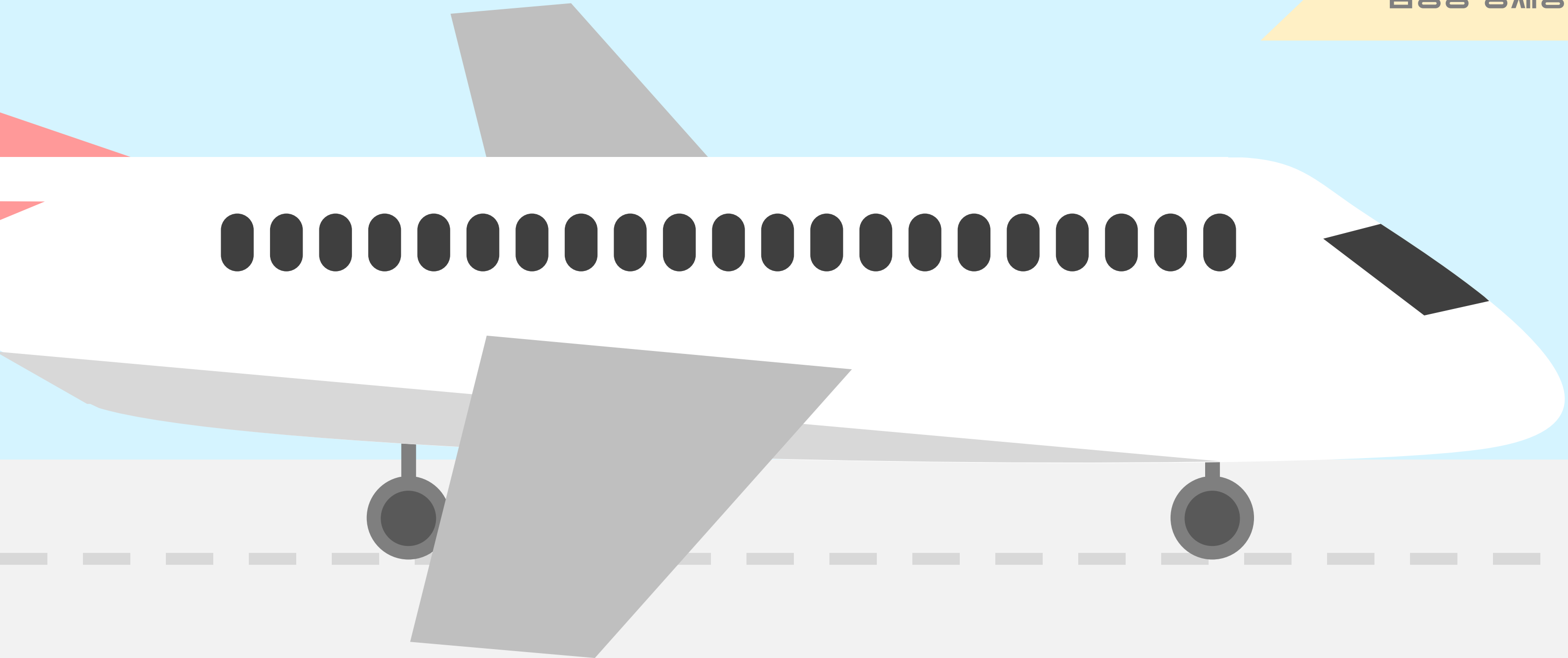
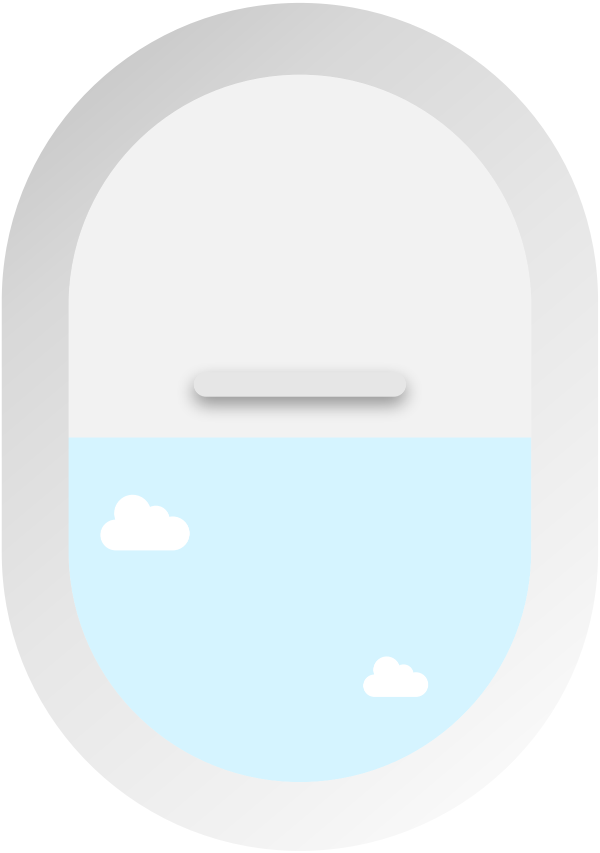
항공사 고객 만족도 예측

동물농장
김경동 정재형 태형배 한지민



목차



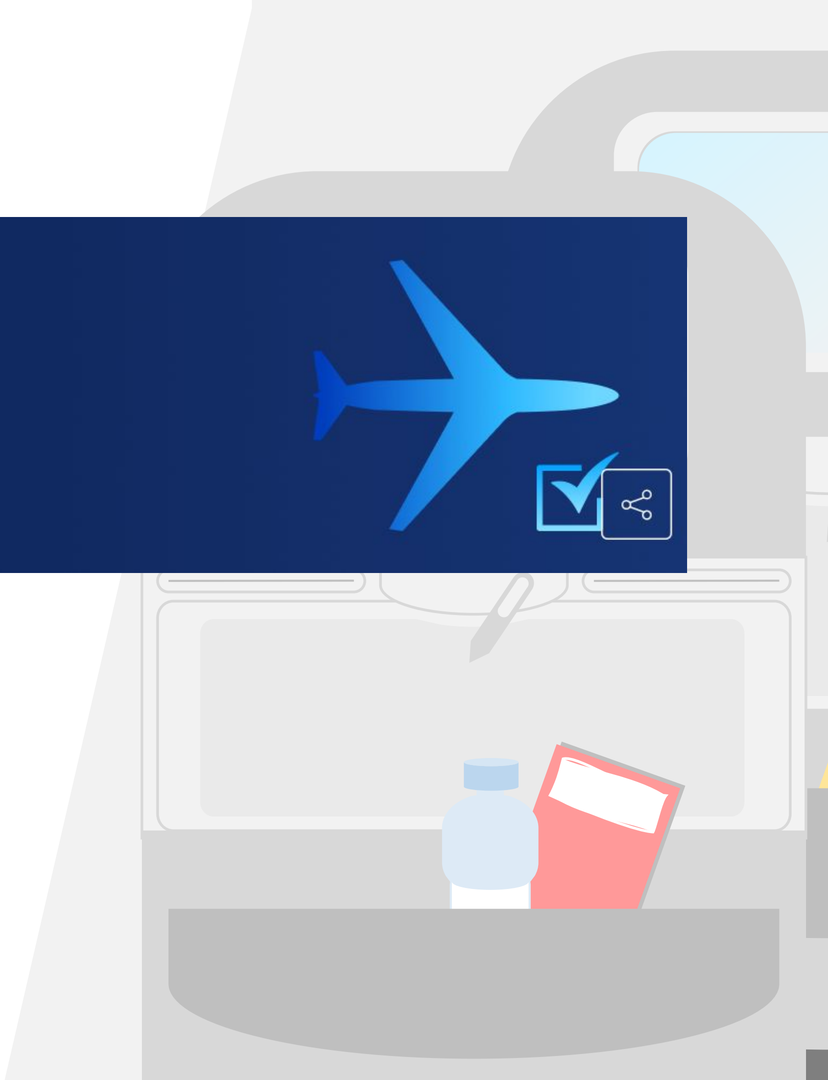
01 주제

주제

- 항공사 고객 만족도 예측

주최

- DACON



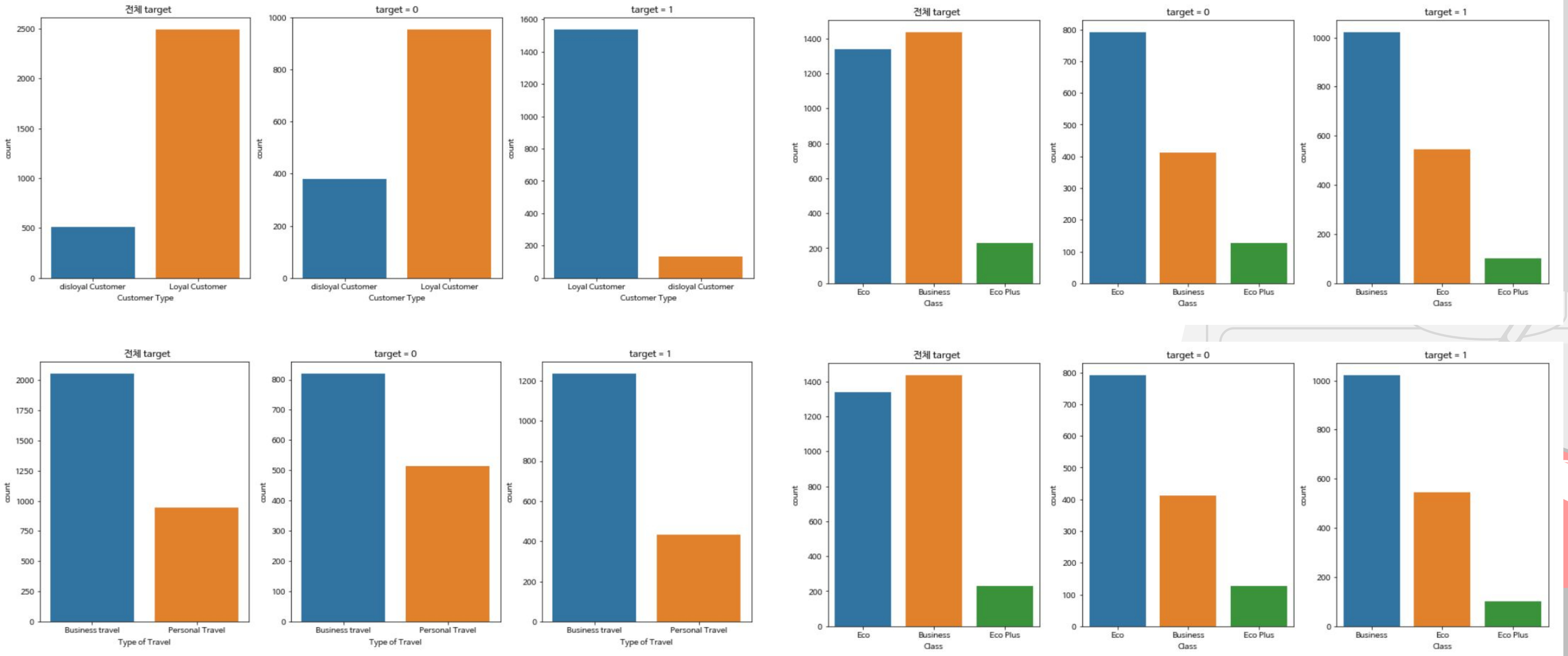
02 EDA

Histogram

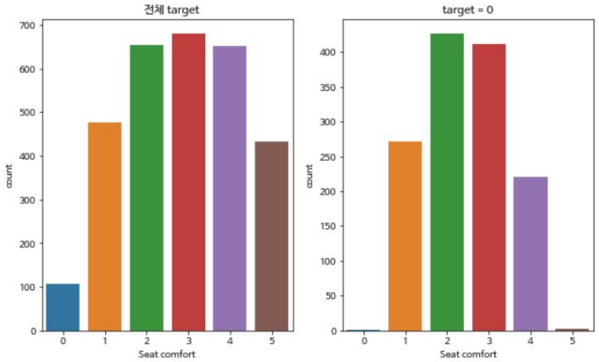
데이터의 분포 확인

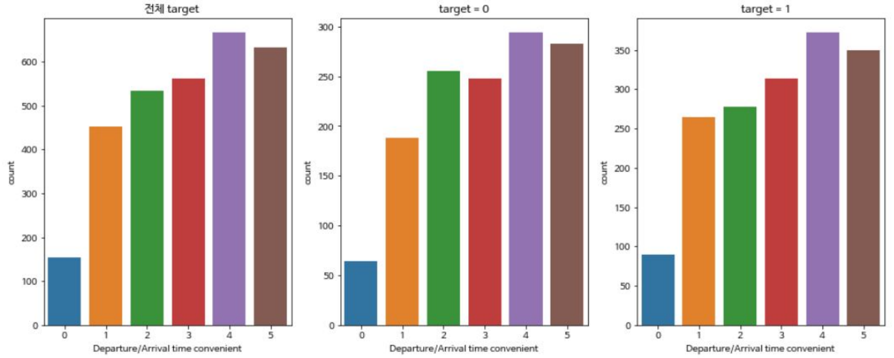
02 EDA

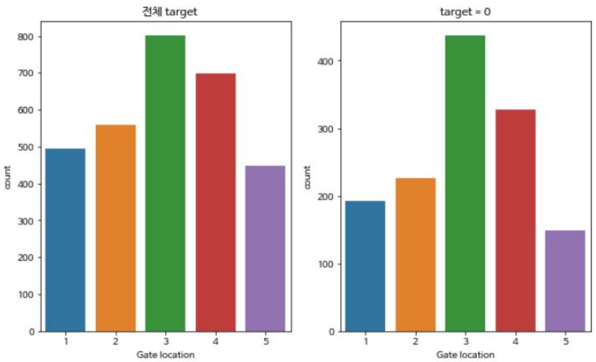
범주형 변수와 Countplot

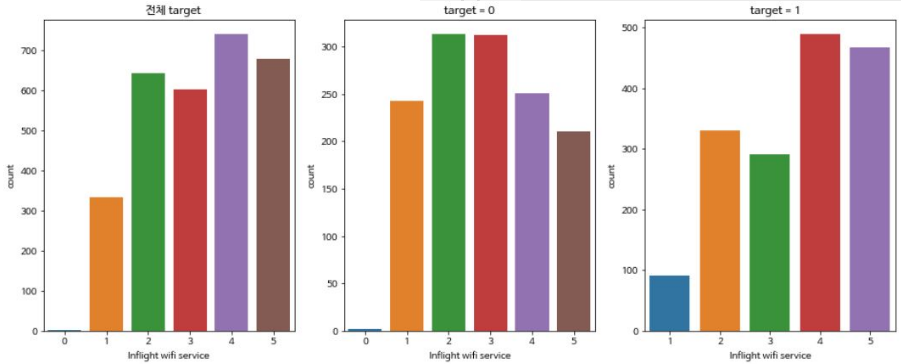


수치형 변수와 Countplot



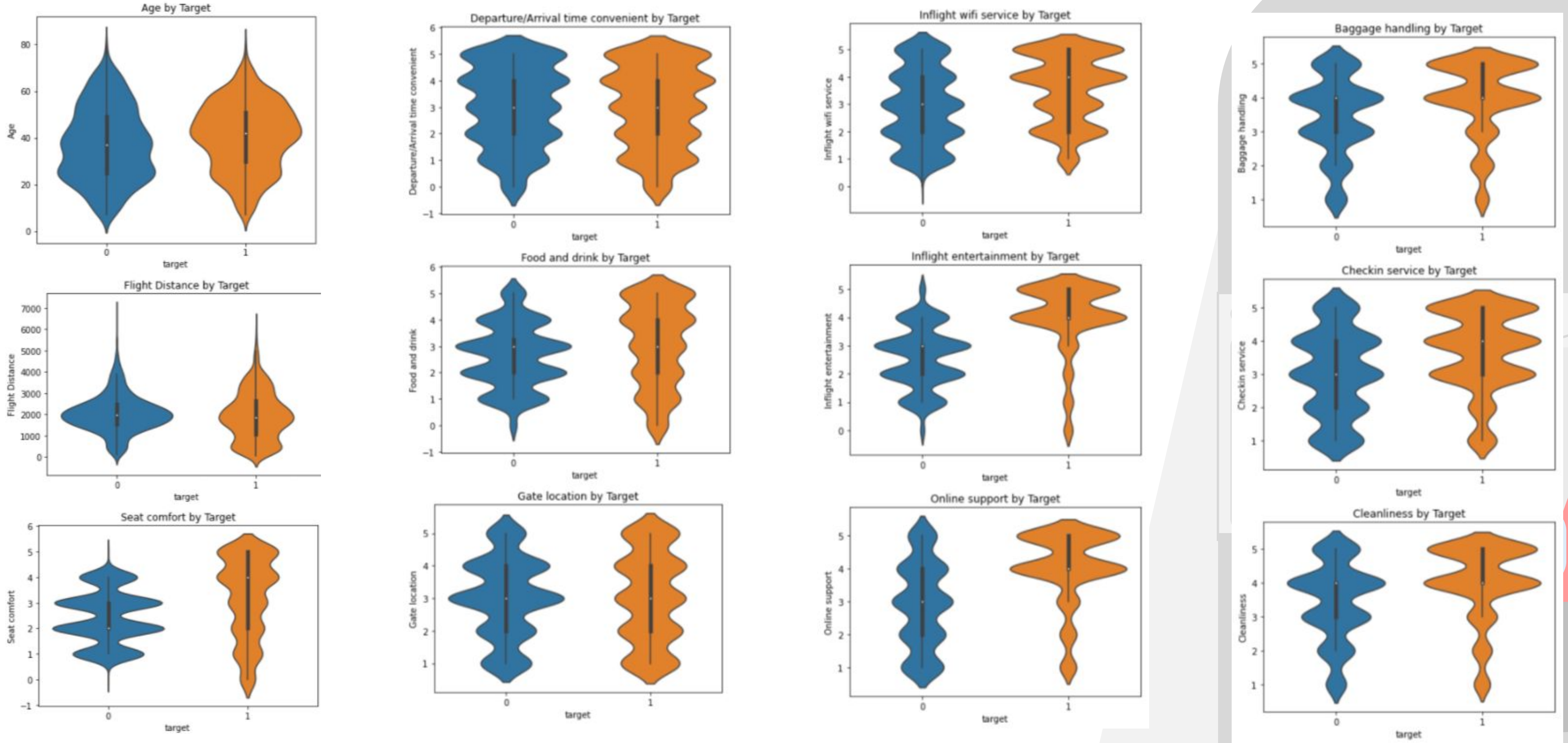




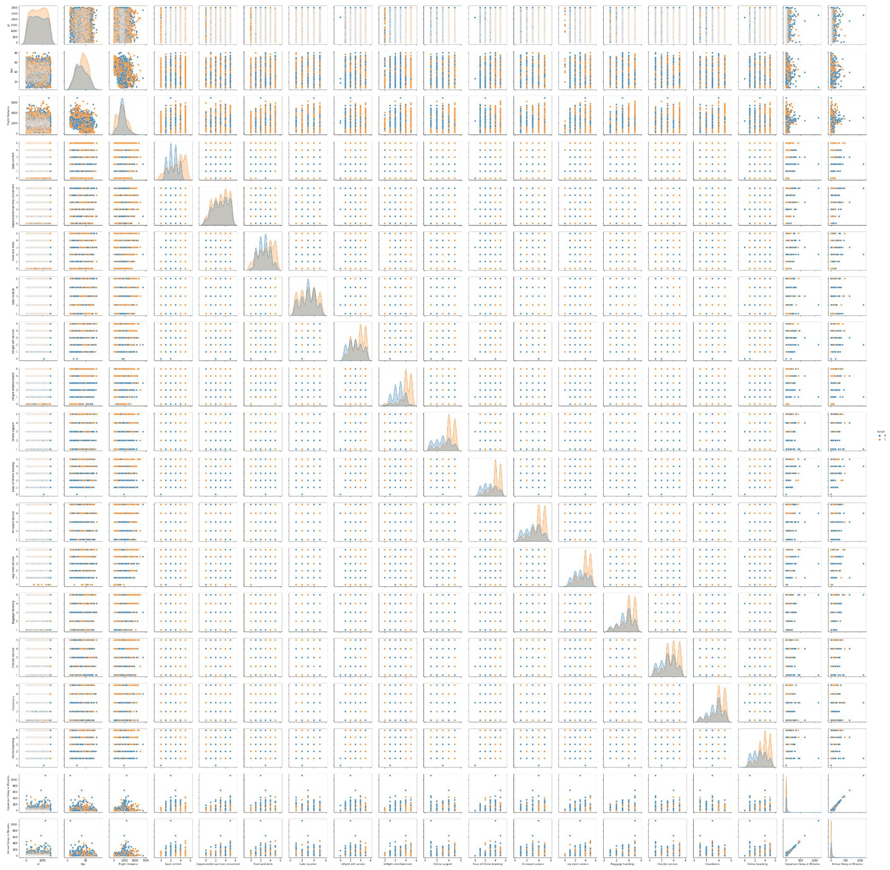


02 EDA

Violinplot



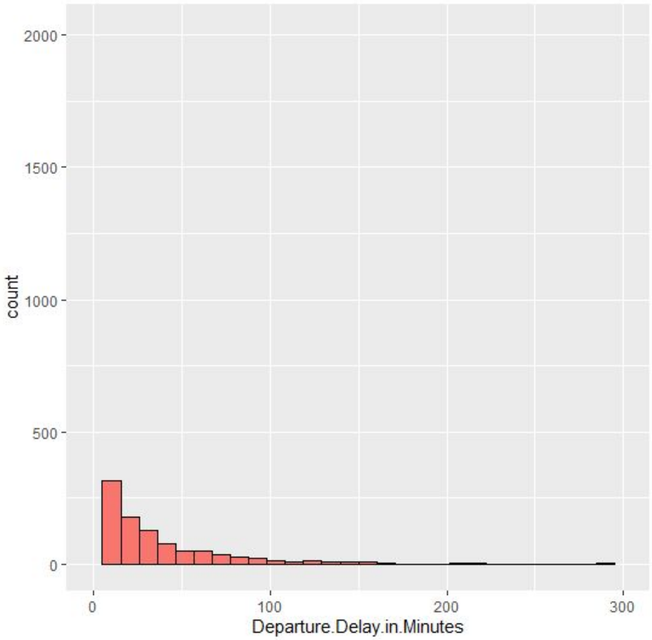
02 EDA

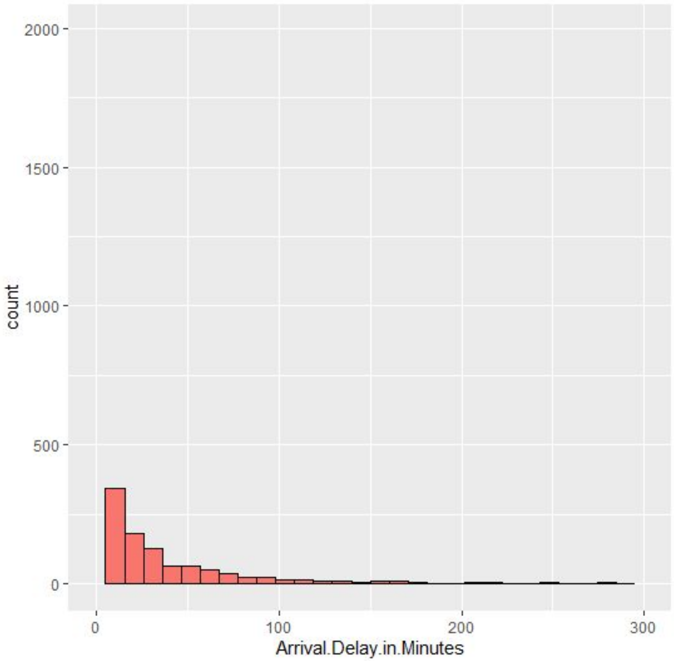


수치형 변수간
Pairplot

연속형, 범주형 변수
판별 가능

연속형 변수와 Histplot





02 EDA

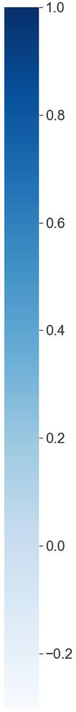
	Gender	Customer Type	Age	Type of Travel	Class	Flight Distance	Seat comfort	Departure/Arrival time convenient	Food and drink	Gate location	Inflight wifi service	Inflight entertainment	Online support	Ease of Online booking	On-board service	Leg room service	Baggage handling	Checkin service	Cleanliness	Online boarding	Departure Delay in Minutes	Arrival Delay in Minutes	target
Gender	1.00	-0.02	0.00	-0.02	0.01	-0.12	0.08	-0.03	0.07	0.02	0.04	0.14	0.12	0.12	0.11	0.10	0.04	0.03	0.03	0.08	-0.02	-0.02	0.23
Customer Type	-0.02	1.00	0.27	0.30	0.09	-0.04	0.04	0.23	0.05	0.01	0.06	0.22	0.17	0.13	0.06	0.08	-0.02	0.03	-0.02	0.09	-0.02	-0.02	0.27
Age	0.00	0.27	1.00	-0.05	0.13	-0.24	0.01	0.03	0.01	-0.00	0.02	0.11	0.12	0.08	0.04	0.09	-0.02	0.04	-0.02	0.04	-0.02	-0.02	0.11
Type of Travel	-0.02	0.30	-0.05	1.00	-0.56	-0.12	-0.01	0.19	-0.05	-0.03	-0.02	-0.10	-0.05	-0.04	0.02	-0.02	0.02	0.05	0.05	-0.02	0.00	-0.01	-0.13
Class	0.01	0.09	0.13	-0.56	1.00	0.15	-0.03	-0.04	0.05	0.02	0.07	0.24	0.17	0.11	0.14	0.14	0.12	0.14	0.10	0.10	-0.02	-0.03	0.29
Flight Distance	-0.12	-0.04	-0.24	-0.12	0.15	1.00	-0.02	0.01	0.03	0.02	-0.01	-0.04	-0.05	-0.03	-0.03	-0.01	0.04	-0.01	0.02	-0.01	0.11	0.11	-0.05
Seat comfort	0.08	0.04	0.01	-0.01	-0.03	-0.02	1.00	0.44	0.73	0.42	0.15	0.43	0.13	0.23	0.13	0.16	0.11	0.03	0.10	0.15	-0.03	-0.02	0.27
Departure/Arrival time convenient	-0.03	0.23	0.03	0.19	-0.04	0.01	0.44	1.00	0.52	0.52	0.01	0.11	0.02	0.03	0.07	0.01	0.06	0.04	0.07	0.03	0.04	0.04	-0.01
Food and drink	0.07	0.05	0.01	-0.05	0.05	0.03	0.73	0.52	1.00	0.52	0.03	0.37	0.03	0.08	0.08	0.09	0.04	0.01	0.05	0.05	-0.01	-0.01	0.15
Gate location	0.02	0.01	-0.00	-0.03	0.02	0.02	0.42	0.52	0.52	1.00	0.01	0.03	0.03	0.03	0.01	-0.02	-0.00	-0.04	-0.01	0.04	0.00	0.01	0.00
Inflight wifi service	0.04	0.06	0.02	-0.02	0.07	-0.01	0.15	0.01	0.03	0.01	1.00	0.27	0.56	0.60	0.07	0.05	0.04	0.09	0.03	0.63	-0.04	-0.04	0.24
Inflight entertainment	0.14	0.22	0.11	-0.10	0.24	-0.04	0.43	0.11	0.37	0.03	0.27	1.00	0.42	0.33	0.18	0.16	0.10	0.21	0.08	0.35	-0.05	-0.05	0.52
Online support	0.12	0.17	0.12	-0.05	0.17	-0.05	0.13	0.02	0.03	0.03	0.56	0.42	1.00	0.62	0.17	0.16	0.08	0.21	0.08	0.68	-0.05	-0.06	0.41
Ease of Online booking	0.12	0.13	0.08	-0.04	0.11	-0.03	0.23	0.03	0.08	0.03	0.60	0.33	0.62	1.00	0.42	0.36	0.38	0.12	0.39	0.68	-0.01	-0.02	0.45
On-board service	0.11	0.06	0.04	0.02	0.14	-0.03	0.13	0.07	0.08	0.01	0.07	0.18	0.17	0.42	1.00	0.42	0.55	0.28	0.53	0.15	-0.03	-0.04	0.36
Leg room service	0.10	0.08	0.09	-0.02	0.14	-0.01	0.16	0.01	0.09	-0.02	0.05	0.16	0.16	0.36	0.42	1.00	0.43	0.19	0.41	0.13	-0.01	-0.01	0.31
Baggage handling	0.04	-0.02	-0.02	0.02	0.12	0.04	0.11	0.06	0.04	-0.00	0.04	0.10	0.08	0.38	0.55	0.43	1.00	0.26	0.61	0.09	-0.02	-0.02	0.23
Checkin service	0.03	0.03	0.04	0.05	0.14	-0.01	0.03	0.04	0.01	-0.04	0.09	0.21	0.21	0.12	0.28	0.19	0.26	1.00	0.25	0.17	-0.05	-0.05	0.25
Cleanliness	0.03	-0.02	-0.02	0.05	0.10	0.02	0.10	0.07	0.05	-0.01	0.03	0.08	0.08	0.39	0.53	0.41	0.61	0.25	1.00	0.09	-0.11	-0.11	0.23
Online boarding	0.08	0.09	0.04	-0.02	0.10	-0.01	0.15	0.03	0.05	0.04	0.63	0.35	0.68	0.68	0.15	0.13	0.09	0.17	0.09	1.00	-0.04	[illegible]	0.35
Departure Delay in Minutes	-0.02	-0.02	-0.02	0.00	-0.02	0.11	-0.03	0.04	-0.01	0.00	-0.04	-0.05	-0.05	-0.01	-0.03	-0.01	-0.02	-0.05	-0.11	-0.04	1.00	0.98	-0.10
Arrival Delay in Minutes	-0.02	-0.02	-0.02	-0.01	-0.03	0.11	-0.02	0.04	-0.01	0.01	-0.04	-0.05	-0.06	-0.02	-0.04	-0.01	-0.02	-0.05	-0.11	-0.03	0.98	1.00	-0.11
target	0.23	0.27	0.11	-0.13	0.29	-0.05	0.27	-0.01	0.15	0.00	0.24	0.52	0.41	0.45	0.36	0.31	0.23	0.25	0.23	0.35	-0.10	-0.11	1.00

피어슨 상관관계 분석

수치형(등간, 등비)

02 EDA

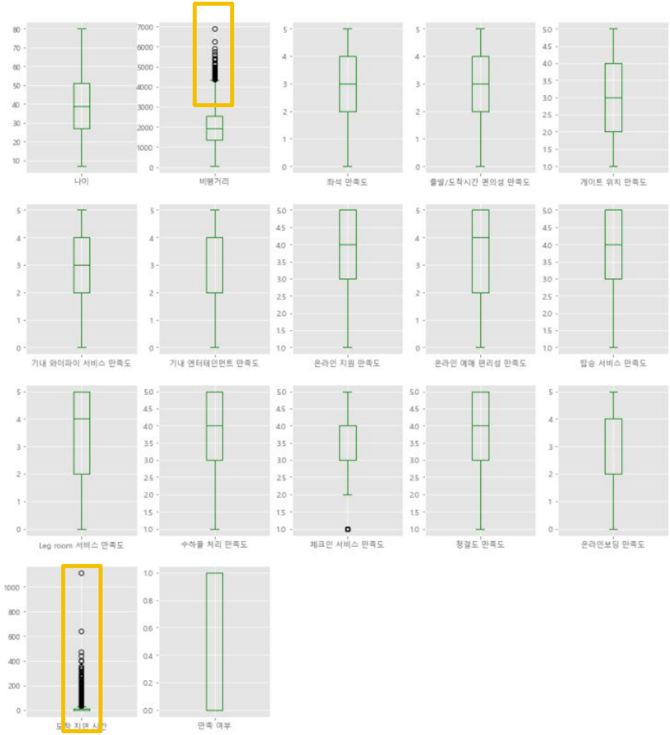
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	1.00	-0.02	-0.01	0.02	-0.01	0.13	-0.08	0.04	-0.08	-0.02	-0.04	-0.15	-0.12	-0.12	-0.11	-0.10	-0.05	-0.03	-0.04	-0.08	0.02	0.00	-0.23
1	-0.02	1.00	-0.29	-0.30	0.04	0.06	-0.04	-0.21	-0.05	-0.00	-0.06	-0.22	-0.16	-0.12	-0.07	-0.08	0.01	-0.03	0.01	-0.08	0.02	0.02	-0.27
2	-0.01	-0.29	1.00	-0.05	-0.11	-0.25	0.01	0.03	0.01	-0.01	0.02	0.12	0.12	0.09	0.06	0.10	-0.00	0.04	-0.01	0.04	-0.02	-0.00	0.12
3	0.02	-0.30	-0.05	1.00	0.53	-0.11	-0.00	0.20	-0.04	-0.03	-0.02	-0.09	-0.04	-0.04	0.02	-0.02	0.02	0.05	0.05	-0.02	0.00	-0.00	-0.13
4	-0.01	0.04	-0.11	0.53	1.00	-0.15	0.03	0.05	-0.04	-0.02	-0.07	-0.23	-0.16	-0.11	-0.17	-0.15	-0.14	-0.16	-0.12	-0.09	-0.01	0.02	-0.29
5	0.13	0.06	-0.25	-0.11	-0.15	1.00	-0.03	0.00	0.01	0.00	-0.01	-0.05	-0.05	-0.04	-0.02	-0.00	0.03	-0.01	0.01	-0.01	0.08	0.04	-0.06
6	-0.08	-0.04	0.01	-0.00	0.03	-0.03	1.00	0.44	0.72	0.42	0.15	0.40	0.13	0.22	0.13	0.14	0.11	0.03	0.11	0.15	-0.03	-0.04	0.29
7	0.04	-0.21	0.03	0.20	0.05	0.00	0.44	1.00	0.53	0.54	0.00	0.09	0.02	0.02	0.08	-0.00	0.08	0.05	0.08	0.03	0.01	0.02	-0.01
8	-0.08	-0.05	0.01	-0.04	-0.04	0.01	0.72	0.53	1.00	0.53	0.03	0.33	0.04	0.08	0.08	0.08	0.04	0.01	0.06	0.05	0.00	0.00	0.16
9	-0.02	-0.00	-0.01	-0.03	-0.02	0.00	0.42	0.54	0.53	1.00	0.01	0.03	0.03	0.03	0.01	-0.02	-0.01	-0.04	-0.01	0.04	0.00	0.02	0.00
10	-0.04	-0.06	0.02	-0.02	-0.07	-0.01	0.15	0.00	0.03	0.01	1.00	0.28	0.55	0.58	0.07	0.05	0.06	0.08	0.04	0.62	-0.05	-0.06	0.24
11	-0.15	-0.22	0.12	-0.09	-0.23	-0.05	0.40	0.09	0.33	0.03	0.28	1.00	0.45	0.35	0.21	0.18	0.14	0.22	0.13	0.37	-0.03	-0.06	0.58
12	-0.12	-0.16	0.12	-0.04	-0.16	-0.05	0.13	0.02	0.04	0.03	0.55	0.45	1.00	0.61	0.19	0.18	0.12	0.21	0.11	0.66	-0.05	-0.05	0.42
13	-0.12	-0.12	0.09	-0.04	-0.11	-0.04	0.22	0.02	0.08	0.03	0.58	0.35	0.61	1.00	0.45	0.38	0.42	0.12	0.43	0.66	-0.04	-0.05	0.44
14	-0.11	-0.07	0.06	0.02	-0.17	-0.02	0.13	0.08	0.08	0.01	0.07	0.21	0.19	0.45	1.00	0.42	0.56	0.26	0.56	0.15	-0.04	-0.06	0.37
15	-0.10	-0.08	0.10	-0.02	-0.15	-0.00	0.14	-0.00	0.08	-0.02	0.05	0.18	0.18	0.38	0.42	1.00	0.43	0.18	0.42	0.13	-0.04	-0.04	0.31
16	-0.05	0.01	-0.00	0.02	-0.14	0.03	0.11	0.08	0.04	-0.01	0.06	0.14	0.12	0.42	0.56	0.43	1.00	0.27	0.62	0.11	-0.07	-0.07	0.26
17	-0.03	-0.03	0.04	0.05	-0.16	-0.01	0.03	0.05	0.01	-0.04	0.08	0.22	0.21	0.12	0.26	0.18	0.27	1.00	0.26	0.17	-0.07	-0.08	0.25
18	-0.04	0.01	-0.01	0.05	-0.12	0.01	0.11	0.08	0.06	-0.01	0.04	0.13	0.11	0.43	0.56	0.42	0.62	0.26	1.00	0.10	-0.07	-0.09	0.26
19	-0.08	-0.08	0.04	-0.02	-0.09	-0.01	0.15	0.03	0.05	0.04	0.62	0.37	0.66	0.66	0.15	0.13	0.11	0.17	0.10	1.00	-0.04	-0.04	0.35
20	0.02	0.02	-0.02	0.00	-0.01	0.08	-0.03	0.01	0.00	0.00	-0.05	-0.03	-0.05	-0.04	-0.04	-0.04	-0.07	-0.07	-0.07	-0.04	1.00	0.75	-0.10
21	0.00	0.02	-0.00	-0.00	0.02	0.04	-0.04	0.02	0.00	0.02	-0.06	-0.06	-0.05	-0.05	-0.06	-0.04	-0.07	-0.08	-0.09	-0.04	0.75	1.00	-0.14
22	-0.23	-0.27	0.12	-0.13	-0.29	-0.06	0.29	-0.01	0.16	0.00	0.24	0.58	0.42	0.44	0.37	0.31	0.26	0.25	0.26	0.35	-0.10	-0.14	1.00



스피어만 상관관계 분석

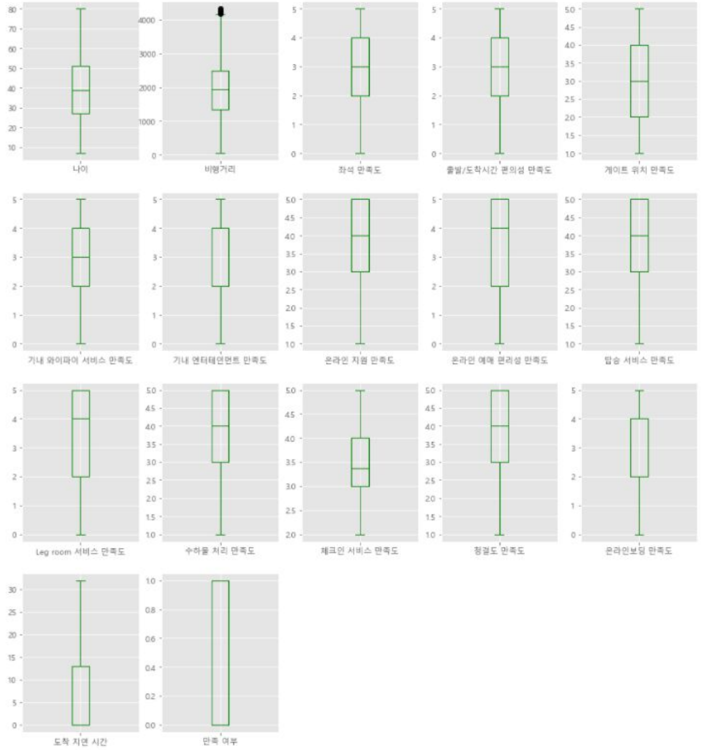
서열척도

02 EDA



이상치 확인 후

평균값 대체



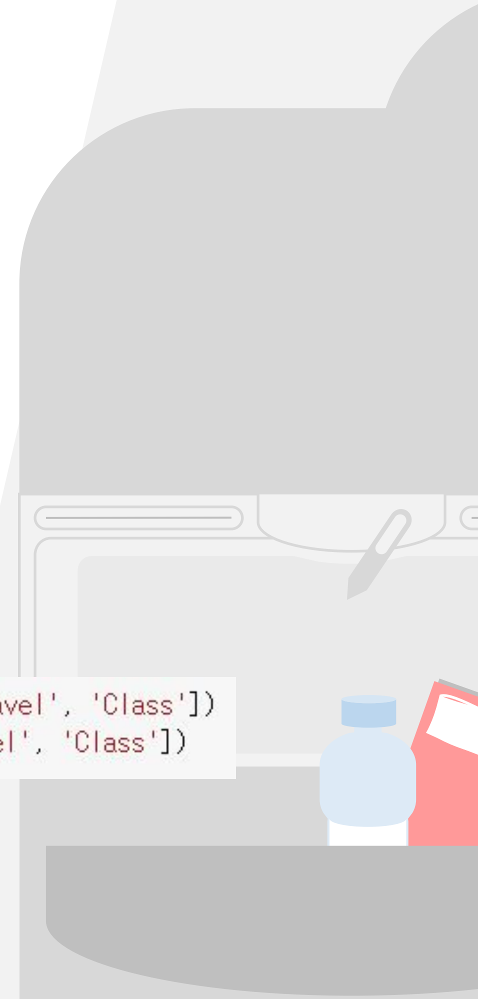
03 데이터 전처리

라벨 인코딩

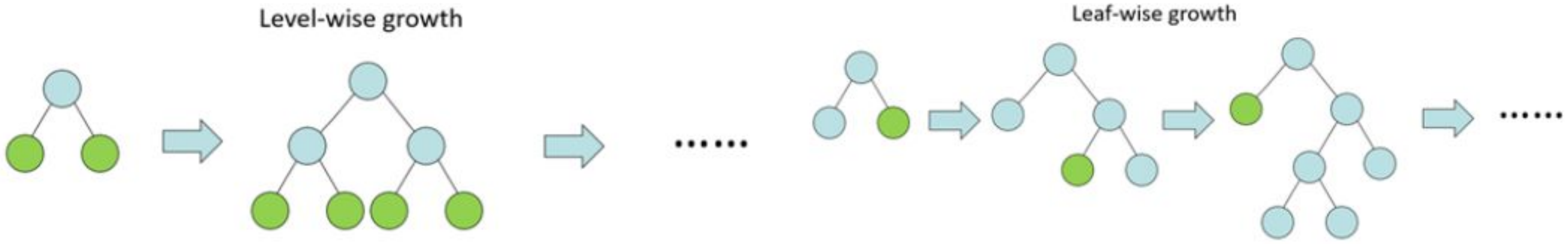
```
le = LabelEncoder()

train["Gender"] = le.fit_transform(train["Gender"])
train["Customer Type"] = le.fit_transform(train["Customer Type"])
train["Type of Travel"] = le.fit_transform(train["Type of Travel"])
train["Class"] = le.fit_transform(train["Class"])
```

원핫 인코딩

```
train1 = pd.get_dummies(data = train1, columns = ['Gender', 'Customer Type', 'Type of Travel', 'Class'])
test1 = pd.get_dummies(data = test1, columns = ['Gender', 'Customer Type', 'Type of Travel', 'Class'])
```



04 모델링 1) LGBM



Leaf Wise 트리 분할 사용

장점

- 대용량 데이터 처리
- 빠른 속도

단점

- 과적합 우려 높음
- 데이터양이 작으면 비효율적

04 모델링 1) LGBM

코드

```
model_LGBM = LGBMClassifier()

gridParams = {
    "learning_rate" : [0.005, 0.01],
    "n_estimators" : [100,500,1000],
    "num_leaves" : [12,16,20],
    "max_bin" : [300,600],
}

grid_cv_LGBM = GridSearchCV(model_LGBM, param_grid=gridParams, cv=5, n_jobs=-1)
grid_cv_LGBM.fit(train,target)
print("최적 하이퍼 파라미터 : ",grid_cv_LGBM.best_params_)
print("최고 예측 정확도 : {:4f}".format(grid_cv_LGBM.best_score_))
```

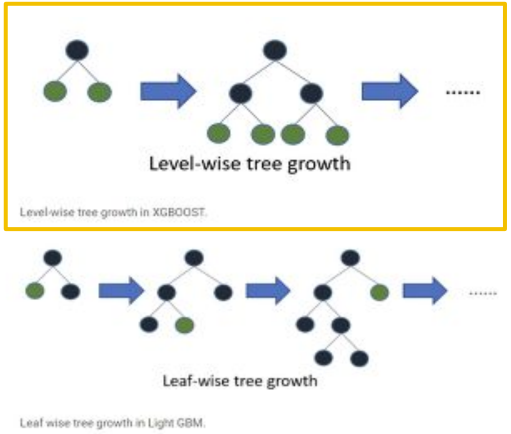
```
최적 하이퍼 파라미터 :  {'learning_rate': 0.01, 'max_bin': 300, 'n_estimators': 1000, 'num_leaves': 16}
최고 예측 정확도 : 0.933333
```

정확성

93.33 %

04 모델링 2) XGBoost

여러개의 의사결정나무를 앙상블한 알고리즘, error 감소 방식



XGBoost
Level-Wise 트리 분할 사용

LGBM
Leaf-Wise 트리 분할 사용

장점

- GBM보다 빠른 처리속도
- 과적합 규제

단점

- 학습속도가 느림
- 하이퍼 파라미터 튜닝시 오래 걸림

04 모델링 2) XGBoost

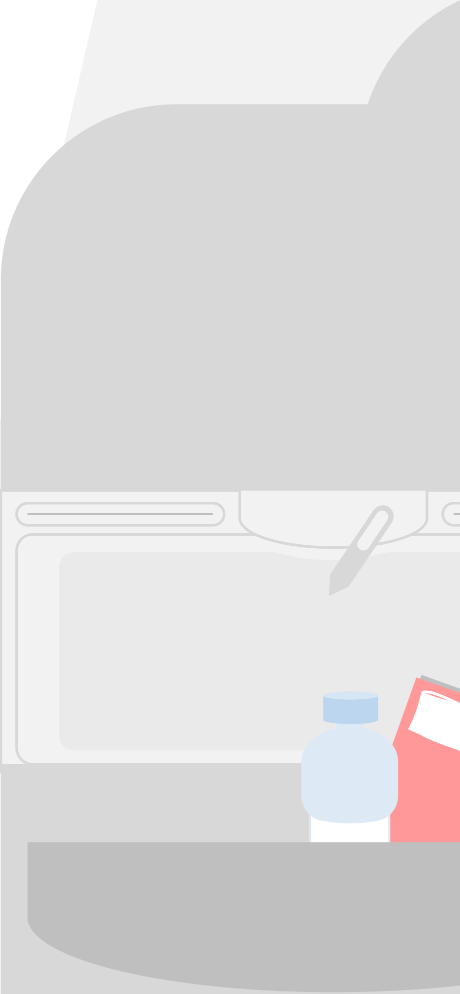
코드

```
#모델링
models = XGBClassifier()

#최적 하이퍼 파라미터 탐색
model_XGB = XGBClassifier(eval_metric='logloss', silent = True)

param_grid={'learning_rate' : [0.05, 0.1, 0.3],
            'max_depth':[4,6,8,10],
            'n_estimators':[50,100,150]}

grid_cv_XGB=GridSearchCV(model_XGB, param_grid=param_grid, cv=5 , n_jobs=-1)
grid_cv_XGB.fit(X_train, y_train)

print('최적 하이퍼 파라미터: ', grid_cv_XGB.best_params_)
print('최고 예측 정확도: {:.4f}'.format(grid_cv_XGB.best_score_))
```

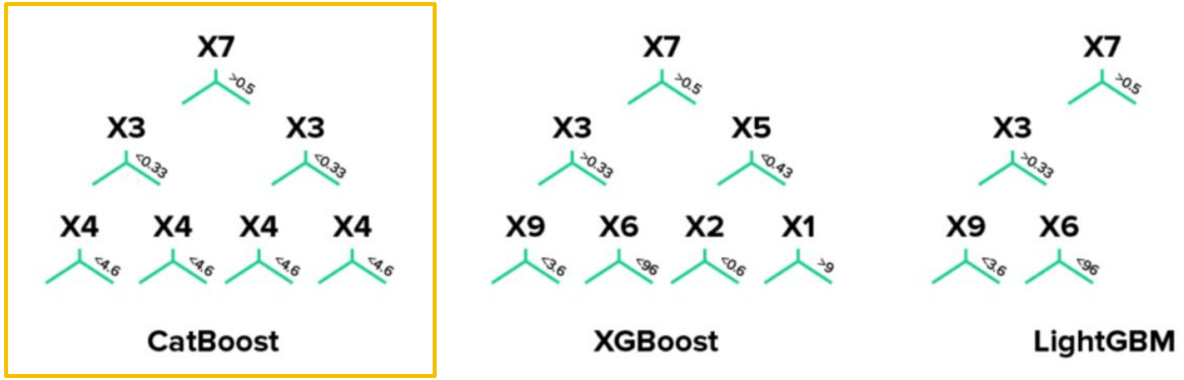
```
최적 하이퍼 파라미터:  {'learning_rate': 0.1, 'max_depth': 6, 'n_estimators': 150}
최고 예측 정확도: 0.9263
```

정확성

92.63 %



04 모델링 3) CatBoost



Feature 모두 동일한 대칭적인 트리 구조 형성

장점

- 빠른 예측 가능
- 과적합 문제 해결

단점

- 결측치 多
  데이터셋에서는 부적합

04 모델링 3) CatBoost

코드

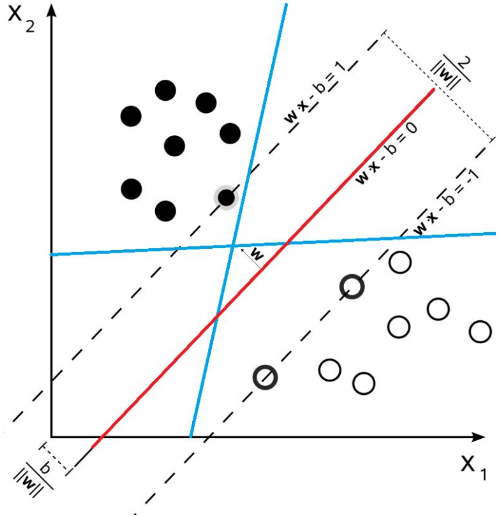
```
#모델 생성
model=CatBoostClassifier()

#최적화할 파라미터 범위
cb_params = {'depth' : [4, 6, 8, 10],
             'learning_rate' : [0.05, 0.1, 0.15],
             'iterations' : [100, 150, 200]
            }


grid_cb = GridSearchCV(model, param_grid=cb_params, scoring = 'accuracy', cv=3, n_jobs=-1)
grid_cb.fit(x_train, y_train)

print("최적 하이퍼 파라미터 : ",grid_cb.best_params_)
print("최고 예측 정확도 : {:4f}".format(grid_cb.best_score_))
```

```
최적 하이퍼 파라미터 :  {'depth': 6, 'iterations': 100, 'learning_rate': 0.15}
최고 예측 정확도 : 0.927500
```

정확성

92.75 %

04 모델링 4) SVM



- **SVM이란 ?**

각각의 분류군을 가지는 데이터에 대해
가장 잘 분류하는 직선을 찾는 방법

장점

- 적은 표본수에도 성능 우수
- 잡음에 강함

단점

- 스케일링에 민감
- 고차원으로 갈수록 계산이 부담

04 모델링 4) SVM

전처리		정확성
OneHot + MinMax	LinearSVC	trainset score : 90.6% valid score : 90.6% (ACC)
OneHot + Standard	LinearSVC	trainset score : 90.6% valid score : 90.6% (ACC)
OneHot + MinMax	SVC	trainset score : 96.6% valid score : 92.8% (ACC)
OneHot + Standard	SVC	trainset score : 96.3% valid score : 92.3% (ACC)

04 모델링 5) 앙상블 : XGB+캣부스트+LGBM

코드

```
best_model_XGB = XGBClassifier(booster='gbtree', colsample_bylevel= 0.9, colsample_bytree=0.8, gamma=1,
                               max_depth= 8, min_child_weight=3, n_estimators=50, nthread=4, objective='binary:logistic', random_state=2, silent=True)
best_model_LGBM = LGBMClassifier(learning_rate=0.01, max_bin=300, n_estimators=1000, num_leaves=16)
best_model_CAT = CatBoostClassifier(silent=True, depth=6, l2_leaf_reg=7, learning_rate=0.1, n_estimators=500)

softVoting_model = VotingClassifier(estimators=[('XGB', best_model_XGB), ('LGBM', best_model_LGBM), ('CAT', best_model_CAT)], voting='soft')
softVoting_model.fit(x_train, y_train)
```

```
# VotingClassifier 학습/예측/평가
pred = softVoting_model.predict(x_test)

# softVoting 분류기 정확도
print('softVoting 분류기 정확도: {0:.4f}'.format(accuracy_score(y_test, pred)))
```

```
softVoting 분류기 정확도: 0.9417
```

정확성

softVoting

94.17 %

04 모델링 5) 앙상블 : XGB+캣부스트+LGBM

코드

```
best_model_XGB = XGBClassifier(booster='gbtree', colsample_bylevel= 0.9, colsample_bytree=0.8, gamma=1,
                               max_depth= 8, min_child_weight=3, n_estimators=50, nthread=4, objective='binary:logistic', random_state=2, silent=True)
best_model_LGBM = LGBMClassifier(learning_rate=0.01, max_bin=300, n_estimators=1000, num_leaves=16)
best_model_CAT = CatBoostClassifier(silent=True, depth=6, l2_leaf_reg=7, learning_rate=0.1, n_estimators=500)

softVoting_model = VotingClassifier(estimators=[('XGB', best_model_XGB), ('LGBM', best_model_LGBM), ('CAT', best_model_CAT)], voting='hard')
softVoting_model.fit(x_train, y_train)
```

```
# VotingClassifier 학습/예측/평가
pred = softVoting_model.predict(x_test)

# hardVoting 분류기 정확도
print('hardVoting 분류기 정확도: {0:.4f}'.format(accuracy_score(y_test, pred)))
```

```
hardVoting 분류기 정확도: 0.9383
```

정확성

hardVoting
93.83 %

04 모델링 5) 앙상블 : XGB+캣부스트+SVM+LGBM

앙상블에 사용할 모델 선정

```
SVM_Linear SVC Model
OneHot + MinMax   trainset score : 0.9066666666666666    OneHot + MinMax valid score : 0.9066666666666666
OneHot + Standard trainset score : 0.9066666666666666    OneHot + Standard valid score : 0.9066666666666666
SVM_SVC Model
OneHot + MinMax   trainset score : 0.96625   OneHot + MinMax valid score : 0.9283333333333333
OneHot + Standard trainset score : 0.9633333333333334    OneHot + Standard valid score : 0.9233333333333333
Catboost
Non-scale trainset score : 0.98375   Non-scale valid score : 0.9416666666666667
XGBoost_Classifier Model
OneHot + MinMax   trainset score : 1.0   OneHot + MinMax valid score : 0.94
OneHot + Standard trainset score : 1.0   OneHot + Standard valid score : 0.9416666666666667
RandomForest Model
OneHot + MinMax   trainset score : 1.0   OneHot + MinMax valid score : 0.9183333333333333
Label + MinMax   trainset score : 1.0    OneHot + MinMax valid score : 0.915
OneHot + Standard trainset score : 1.0   OneHot + Standard valid score : 0.9083333333333333
Label + Standard trainset score : 1.0    OneHot + Standard valid score : 0.9133333333333333
logistic reg Model
OneHot + MinMax   trainset score : 0.9008333333333334    OneHot + MinMax valid score : 0.905
Label + MinMax   trainset score : 0.8441666666666666     OneHot + MinMax valid score : 0.8266666666666667
OneHot + Standard trainset score : 0.90125   OneHot + Standard valid score : 0.905
Label + Standard trainset score : 0.845      OneHot + Standard valid score : 0.825
```

모델 간 독립성을 가지게 선정

```
In [39]: print('LGBM model')
LGBM model

In [40]: print(f'trainset score : {model_LGBM.score(X_train, y_train)} \t valid score : {model_LGBM.score(X_valid, y_valid)}')
trainset score : 0.98125     valid score : 0.9783333333333334

In [41]: print(f'LGBM model confusion_matrix \n {confusion_matrix(y_valid, model_LGBM.predict(X_valid))}')
LGBM model confusion_matrix
 [[276   7]
 [  6 311]]
```

04 모델링 5) 앙상블

앙상블 코드 구성

```
from sklearn.ensemble import VotingClassifier

# 하드 보팅

voting_clf = VotingClassifier(
    estimators = [('svm',model_svm3), ('catboost',model_catboost), ('xgboost',model_xgbc2), ('lgbm',model_LGBM)],
    voting='hard'
    )

voting_clf.fit(X_train,y_train)
```

```
model_svm5 = Pipeline([
    ('scaleing', Scaler3),
    ('SVC', SVC(probability=True))
    ])


# ('catboost',model_catboost),
voting_clf2 = VotingClassifier(
    estimators = [('svm',model_svm5),('catboost',model_catboost), ('xgboost',model_xgbc2), ('lgbm',model_LGBM)],
    voting='soft'
    )

voting_clf2.fit(X_train,y_train)

print('앙상블 모델-Soft Voting')
print(f'trainset score : {voting_clf2.score(X_train, y_train)} \t valid score : {voting_clf2.score(X_valid, y_valid)}')
print(f'앙상블모델 confusion_matrix \n {confusion_matrix(y_valid, voting_clf2.predict(X_valid))}')
```

SVC는 predict_proba()메소드가
probability = True를 사용해야 등장

04 모델링 5) 앙상블

전처리		정확성	
OneHot + MinMax	SVC	train score : 96.6% valid score : 92.8% (ACC)	voting = Hard ACC 94.0%
Non-scaleing	Catboost	train score : 98.3% valid score : 94.1% (ACC)	
OneHot + Standard	XGboost	train score : 100% valid score : 94.1% (ACC)	voting = soft ACC 94.0%
Non-scaleing	LGBM	train score : 98.1% valid score : 97.8% (ACC)	

감사합니다