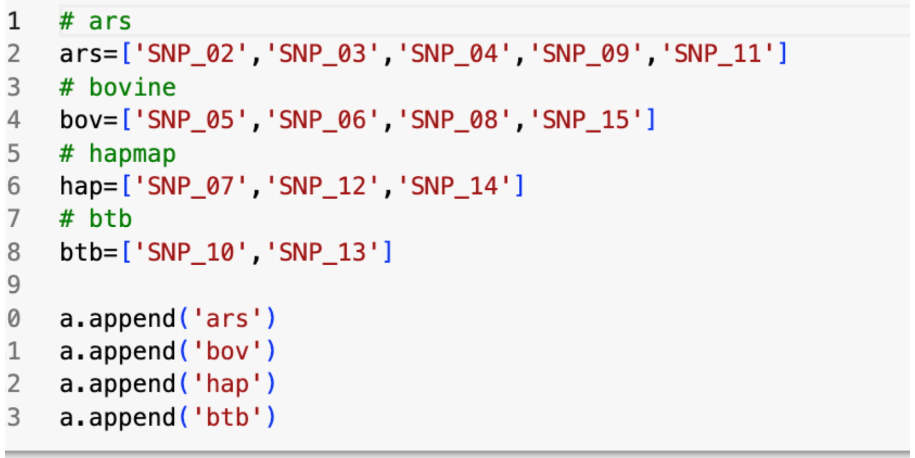
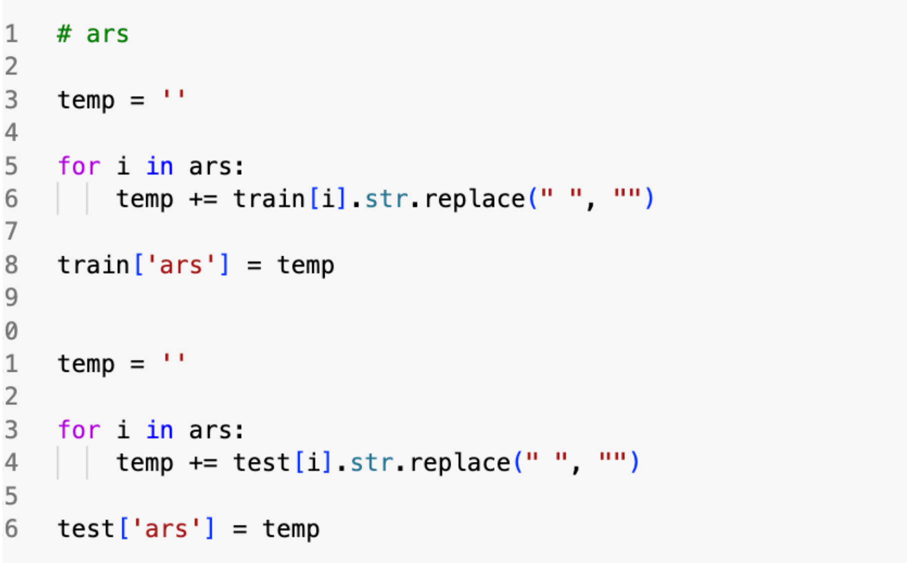
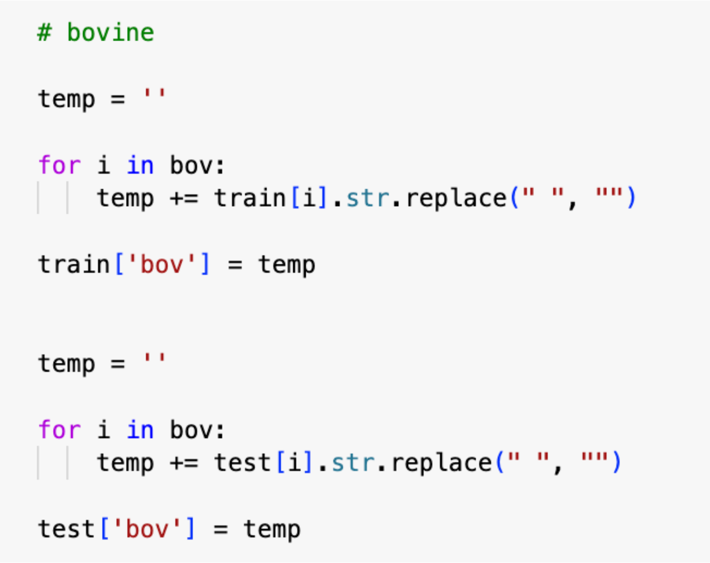
데이콘 유전체 정보 품종 예측



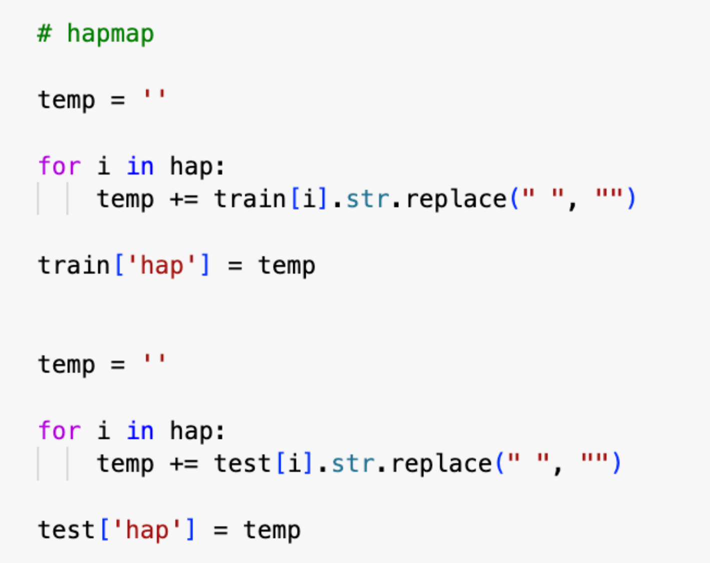
```python
# ars
ars=['SNP_02','SNP_03','SNP_04','SNP_09','SNP_11']
# bovine
bov=['SNP_05','SNP_06','SNP_08','SNP_15']
# hapmap
hap=['SNP_07','SNP_12','SNP_14']
# btb
btb=['SNP_10','SNP_13']

a.append('ars')
a.append('bov')
a.append('hap')
a.append('btb')
```



```python
# ars

temp = ''

for i in ars:
    temp += train[i].str.replace(" ", "")

train['ars'] = temp


temp = ''

for i in ars:
    temp += test[i].str.replace(" ", "")

test['ars'] = temp
```

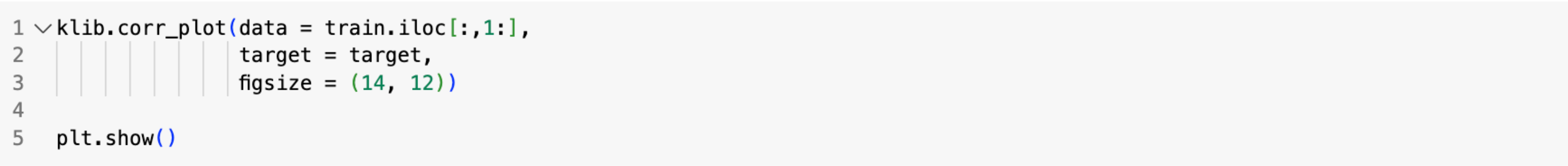


```python
# bovine

temp = ''

for i in bov:
    temp += train[i].str.replace(" ", "")

train['bov'] = temp


temp = ''

for i in bov:
    temp += test[i].str.replace(" ", "")

test['bov'] = temp
```

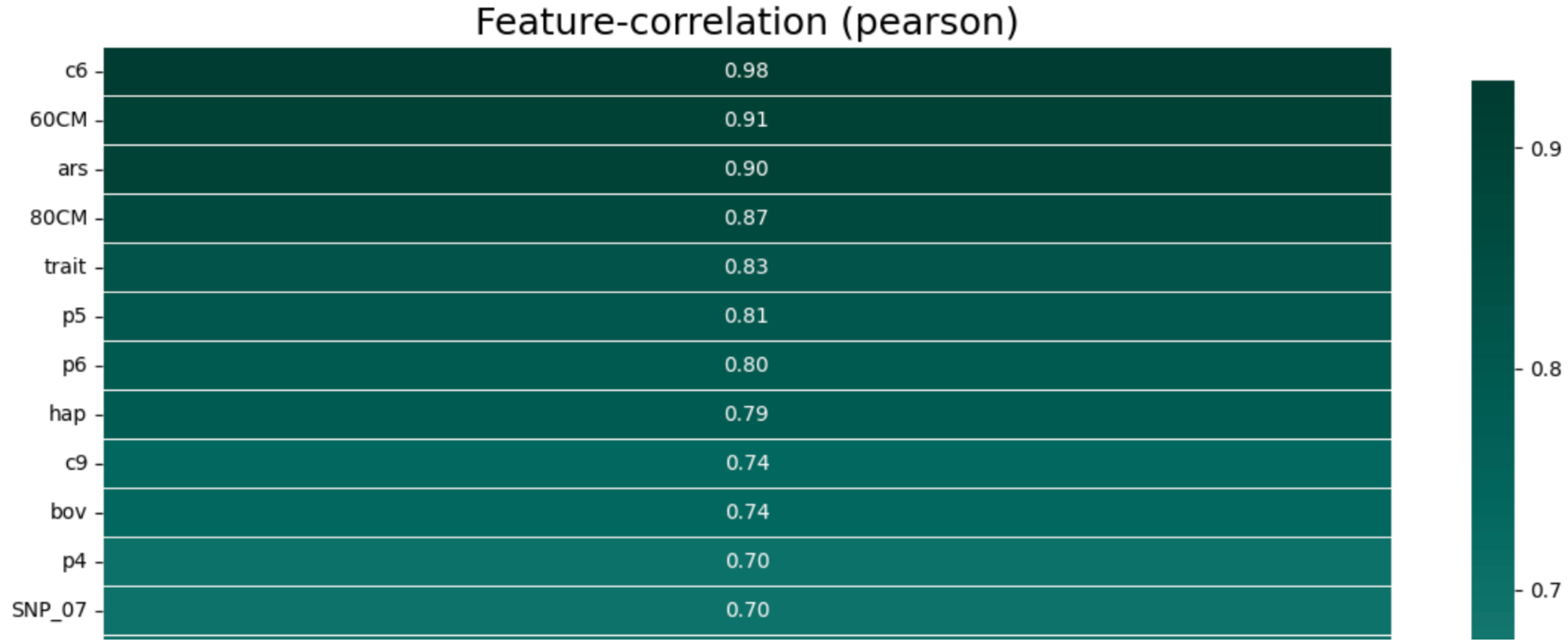


```python
# hapmap

temp = ''

for i in hap:
    temp += train[i].str.replace(" ", "")

train['hap'] = temp


temp = ''

for i in hap:
    temp += test[i].str.replace(" ", "")

test['hap'] = temp
```

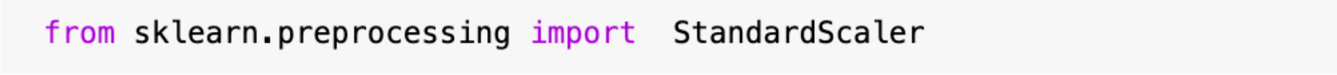
유전체 정보 피처 특징에 따라
피처 생성



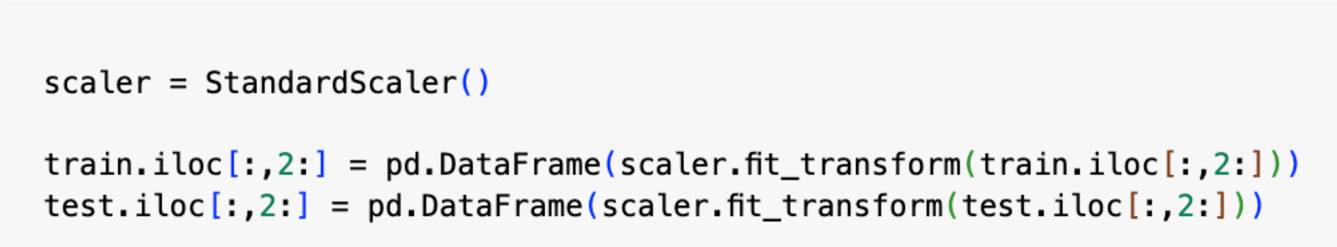
```
klib.corr_plot(data = train.iloc[:,1:],
               target = target,
               figsize = (14, 12))

plt.show()
```



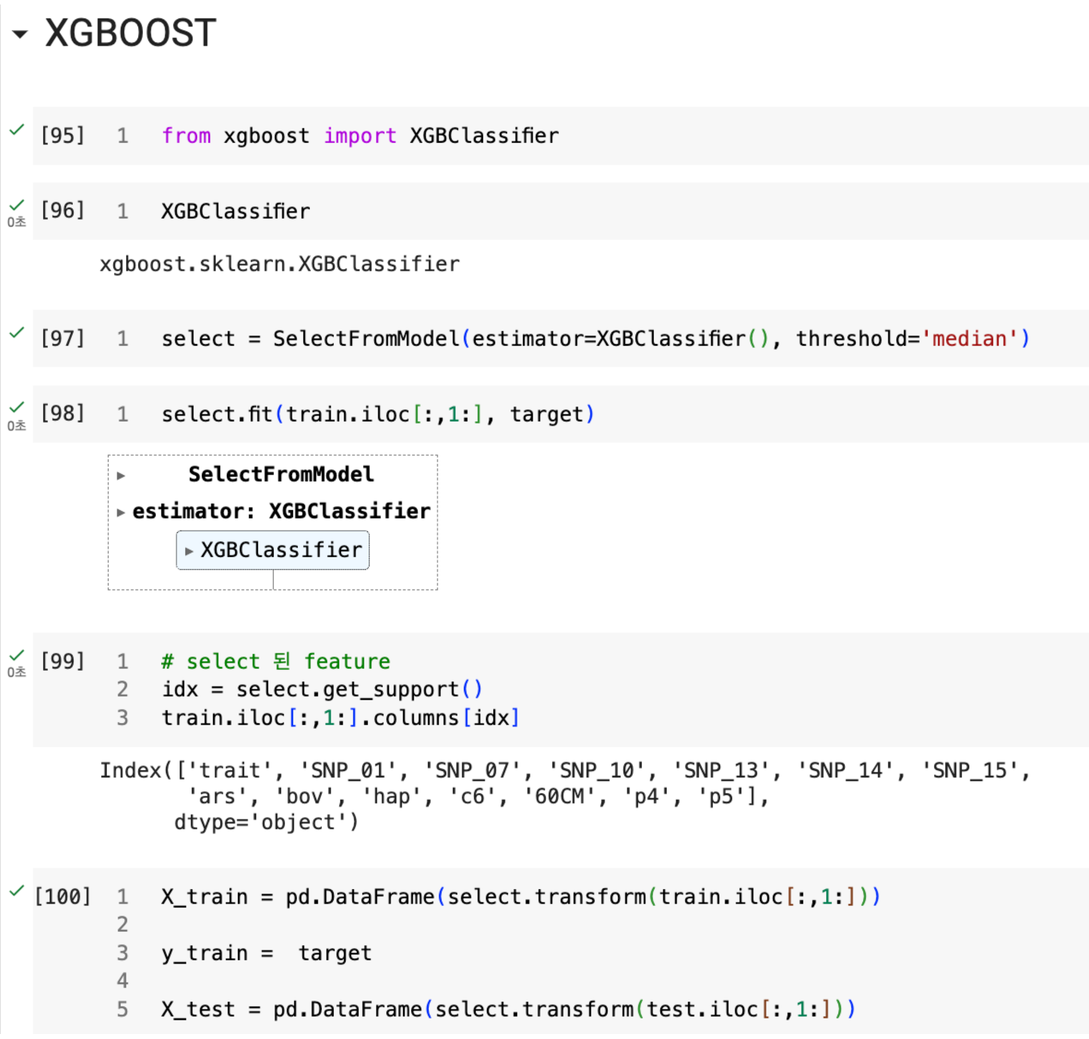
상관관계 확인 및 정규화 작업



```
from sklearn.preprocessing import  StandardScaler
```



```
scaler = StandardScaler()

train.iloc[:,2:] = pd.DataFrame(scaler.fit_transform(train.iloc[:,2:]))
test.iloc[:,2:] = pd.DataFrame(scaler.fit_transform(test.iloc[:,2:]))
```



XGBOOST

```
[95]  from xgboost import XGBClassifier
```

```
[96]  XGBClassifier
```

```
xgboost.sklearn.XGBClassifier
```

```
[97]  select = SelectFromModel(estimator=XGBClassifier(), threshold='median')
```

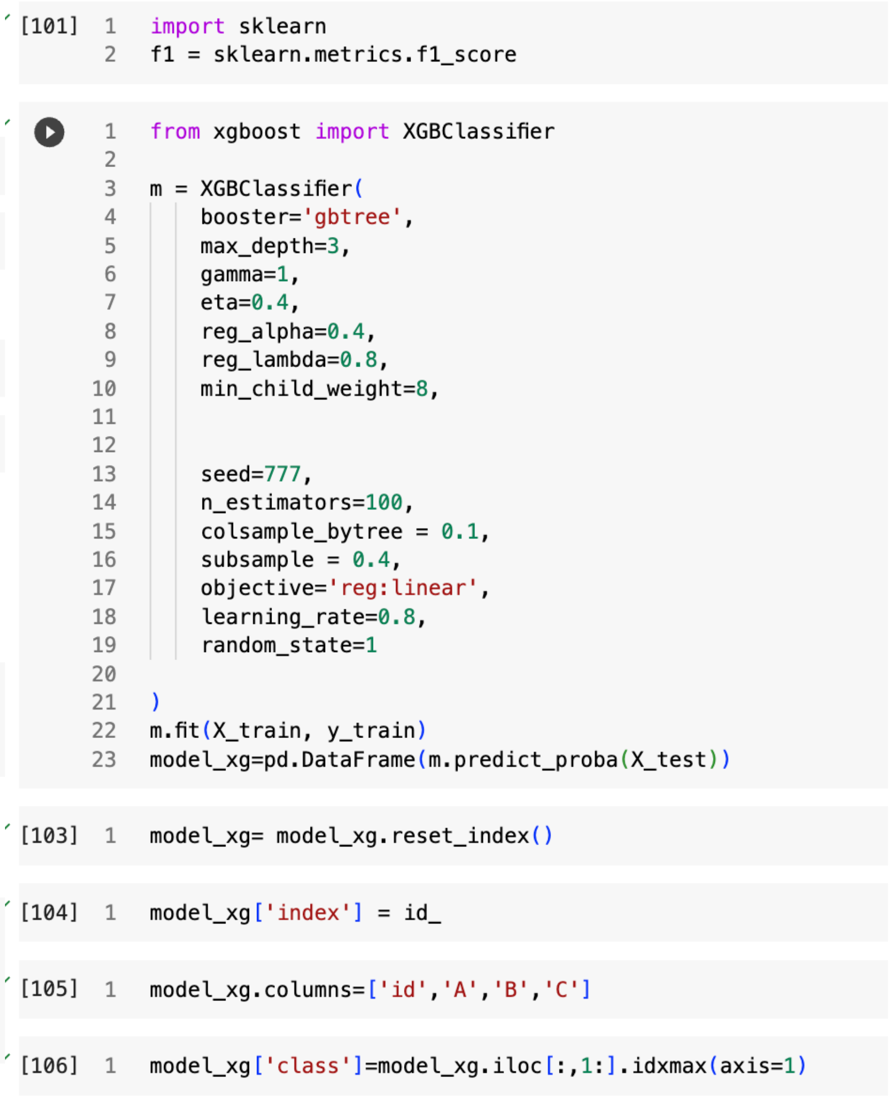
```
[98]  select.fit(train.iloc[:,1:], target)
```

```
▸ SelectFromModel
▸ estimator: XGBClassifier
   ▸ XGBClassifier
```

```
[99]  # select 된 feature
      idx = select.get_support()
      train.iloc[:,1:].columns[idx]
```

```
Index(['trait', 'SNP_01', 'SNP_07', 'SNP_10', 'SNP_13', 'SNP_14', 'SNP_15',
       'ars', 'bov', 'hap', 'c6', '60CM', 'p4', 'p5'],
      dtype='object')
```

```
[100] X_train = pd.DataFrame(select.transform(train.iloc[:,1:]))

      y_train =  target

      X_test = pd.DataFrame(select.transform(test.iloc[:,1:]))
```



```
[101] import sklearn
      f1 = sklearn.metrics.f1_score
```

```
from xgboost import XGBClassifier

m = XGBClassifier(
    booster='gbtree',
    max_depth=3,
    gamma=1,
    eta=0.4,
    reg_alpha=0.4,
    reg_lambda=0.8,
    min_child_weight=8,


    seed=777,
    n_estimators=100,
    colsample_bytree = 0.1,
    subsample = 0.4,
    objective='reg:linear',
    learning_rate=0.8,
    random_state=1

)
m.fit(X_train, y_train)
model_xg=pd.DataFrame(m.predict_proba(X_test))
```

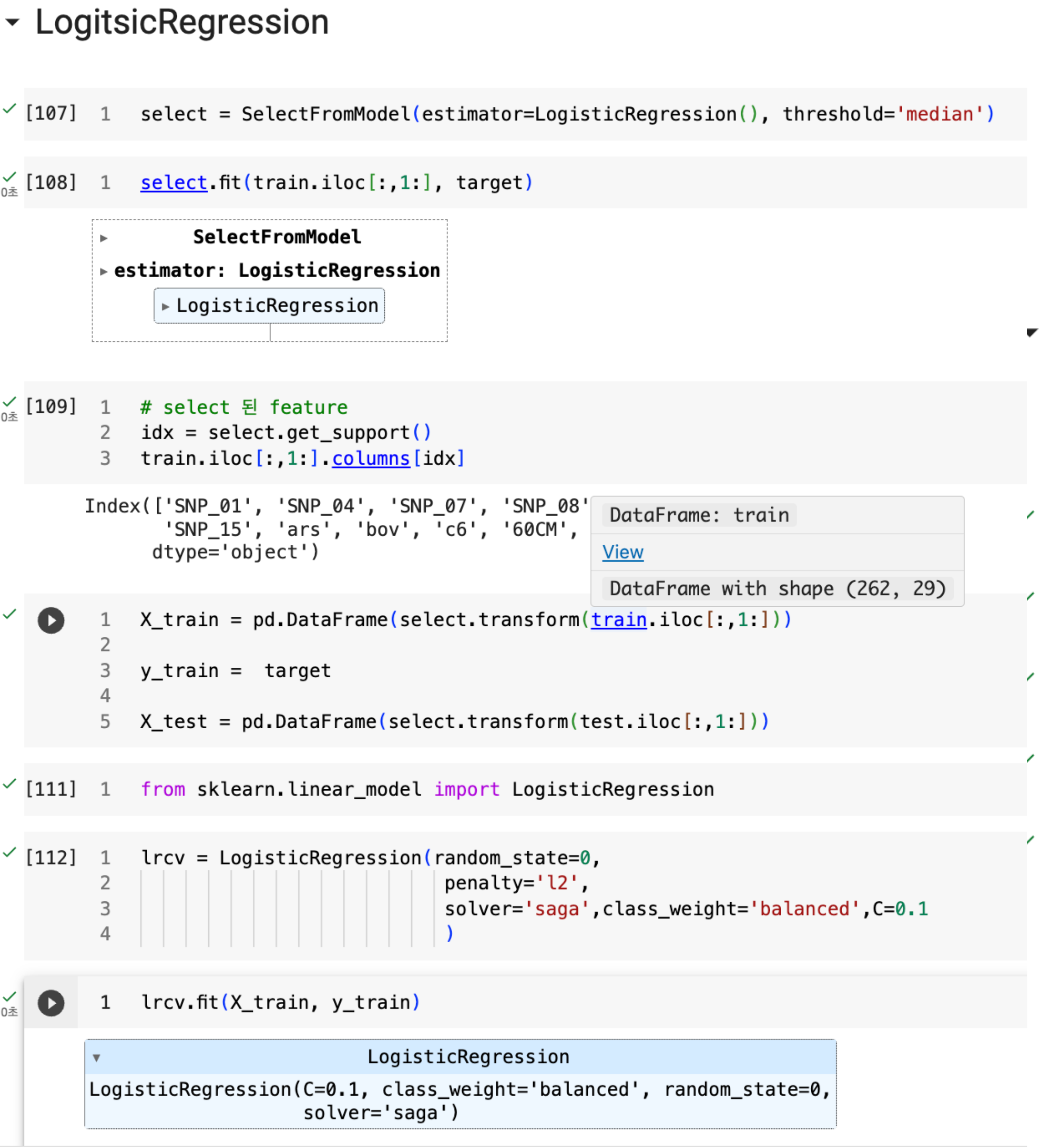
```
[103] model_xg= model_xg.reset_index()
```

```
[104] model_xg['index'] = id_
```

```
[105] model_xg.columns=['id','A','B','C']
```

```
[106] model_xg['class']=model_xg.iloc[:,1:].idxmax(axis=1)
```

사용 모델 1. Xgboost



LogitsicRegression

```
[107] select = SelectFromModel(estimator=LogisticRegression(), threshold='median')
```

```
[108] select.fit(train.iloc[:,1:], target)
```

```
       SelectFromModel
estimator: LogisticRegression
     LogisticRegression
```

```
[109] # select 된 feature
      idx = select.get_support()
      train.iloc[:,1:].columns[idx]
```

```
Index(['SNP_01', 'SNP_04', 'SNP_07', 'SNP_08'
       'SNP_15', 'ars', 'bov', 'c6', '60CM',
      dtype='object')
```

DataFrame: train

View

DataFrame with shape (262, 29)

```
X_train = pd.DataFrame(select.transform(train.iloc[:,1:]))

y_train =  target

X_test = pd.DataFrame(select.transform(test.iloc[:,1:]))
```

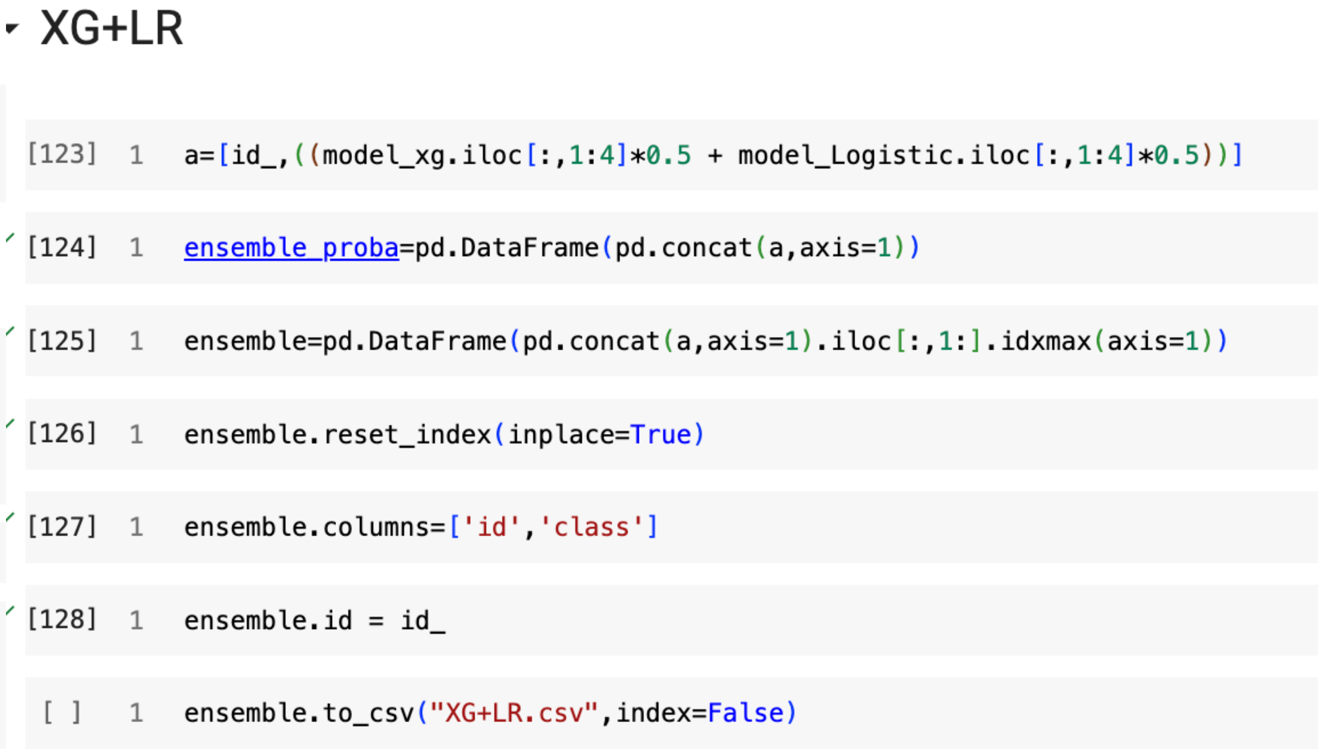
```
[111] from sklearn.linear_model import LogisticRegression
```

```
[112] lrcv = LogisticRegression(random_state=0,
                               penalty='l2',
                               solver='saga',class_weight='balanced',C=0.1
                               )
```

```
lrcv.fit(X_train, y_train)
```

```
                    LogisticRegression
LogisticRegression(C=0.1, class_weight='balanced', random_state=0,
                   solver='saga')
```

사용 모델 2. Logistic Regression



XG+LR

```
[123] a=[id_,((model_xg.iloc[:,1:4]*0.5 + model_Logistic.iloc[:,1:4]*0.5))]
```

```
[124] ensemble_proba=pd.DataFrame(pd.concat(a,axis=1))
```

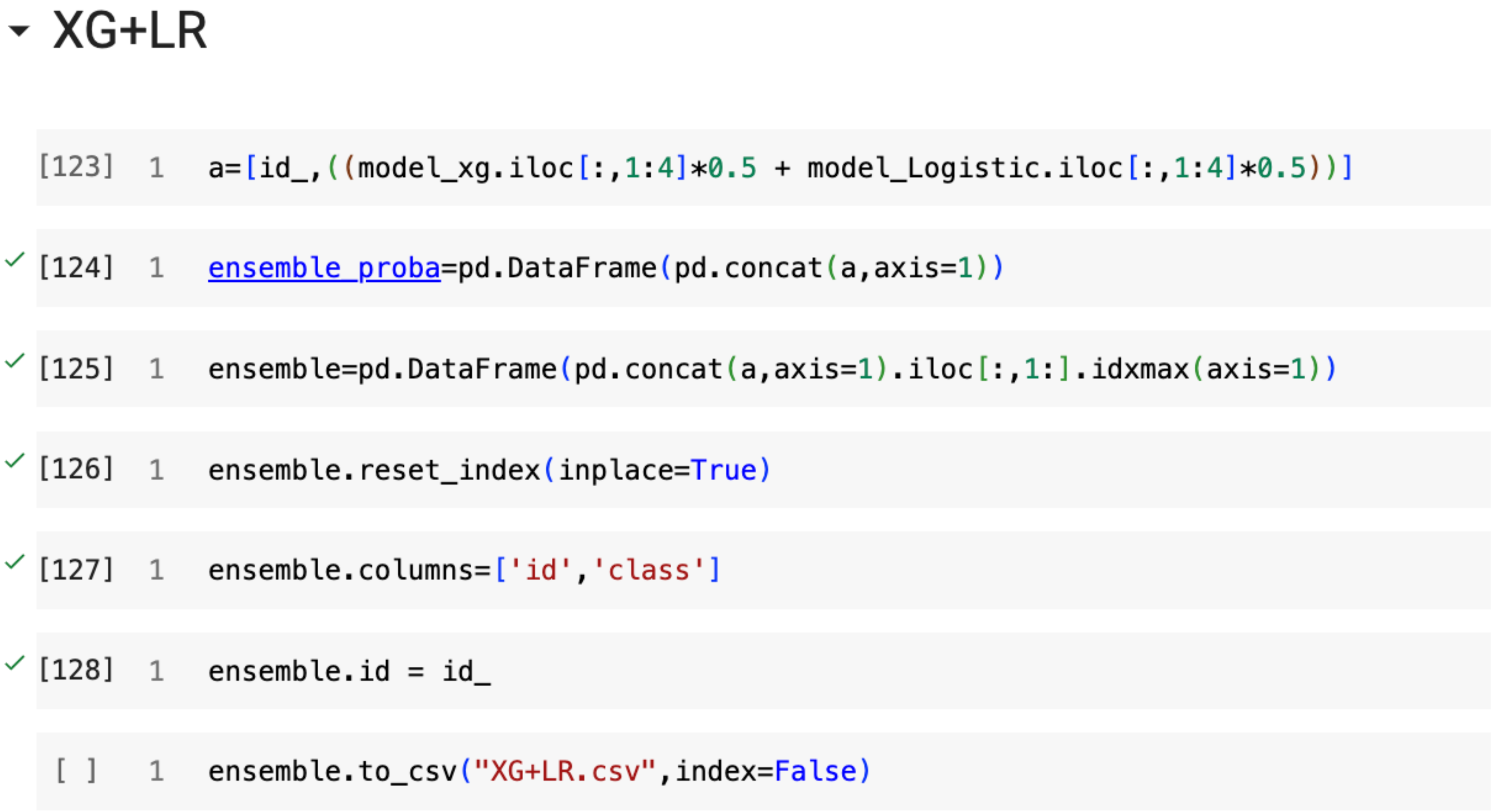
```
[125] ensemble=pd.DataFrame(pd.concat(a,axis=1).iloc[:,1:].idxmax(axis=1))
```

```
[126] ensemble.reset_index(inplace=True)
```

```
[127] ensemble.columns=['id','class']
```

```
[128] ensemble.id = id_
```

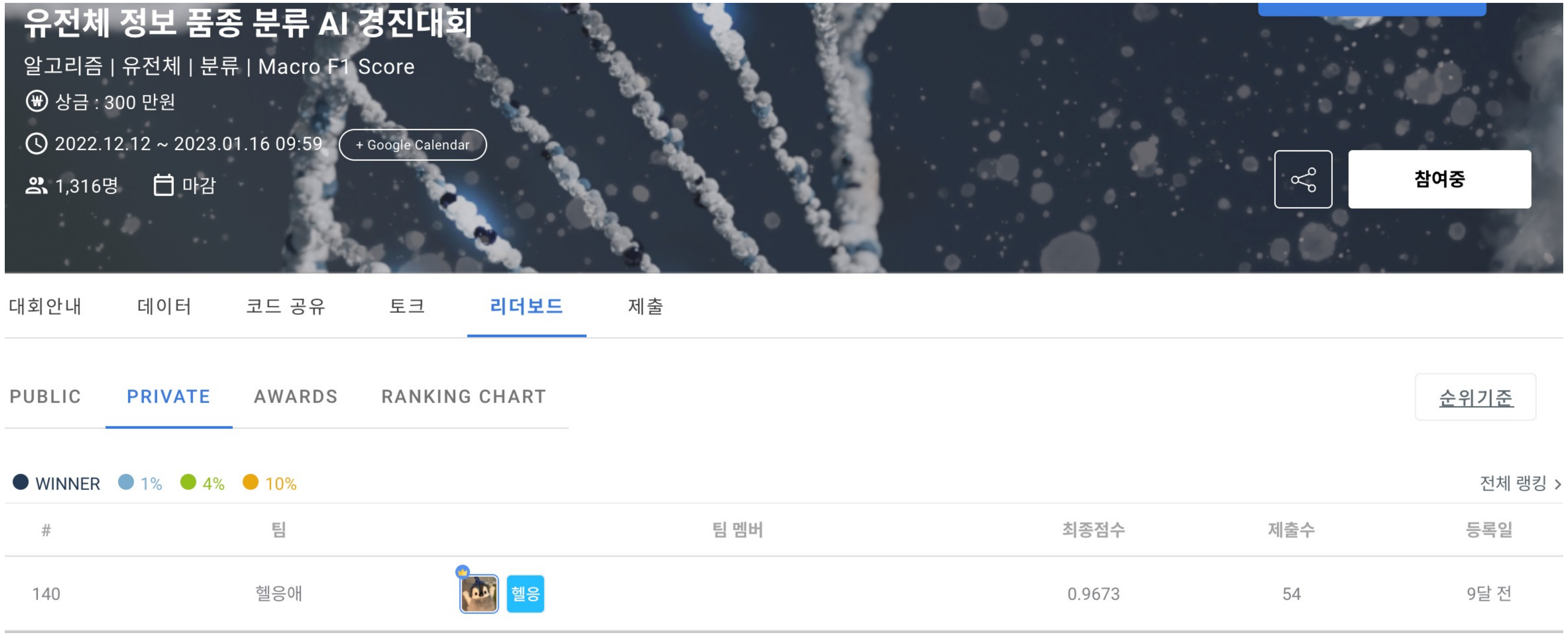
```
[ ] ensemble.to_csv("XG+LR.csv",index=False)
```



XG+LR

```
  [123]  a=[id_,((model_xg.iloc[:,1:4]*0.5 + model_Logistic.iloc[:,1:4]*0.5))]
✓ [124]  ensemble_proba=pd.DataFrame(pd.concat(a,axis=1))
✓ [125]  ensemble=pd.DataFrame(pd.concat(a,axis=1).iloc[:,1:].idxmax(axis=1))
✓ [126]  ensemble.reset_index(inplace=True)
✓ [127]  ensemble.columns=['id','class']
✓ [128]  ensemble.id = id_
  [ ]    ensemble.to_csv("XG+LR.csv",index=False)
```

두 모델의 결과 앙상블



유전체 정보 품종 분류 AI 경진대회

알고리즘 | 유전체 | 분류 | Macro F1 Score

상금 : 300 만원

2022.12.12 ~ 2023.01.16 09:59

1,316명 마감

● WINNER ● 1% ● 4% ● 10%

#	팀	팀 멤버	최종점수	제출수	등록일
140	헬응애	헬응	0.9673	54	9달 전

Private date 데이터
최종 140위
(상위 20%) 달성