+2과목 데이터 모델링

```
# 모의고사 2유형
# 2유형 1번(악성종양 분류)
# 제공된 데이터는 종양의 크기, 모양 등 다양한 변수를 기반으로 해당 종양이 
# 인지 양성종양(benign)인지 분류한 데이터셋이다. 학습 데이터(train)를 이
# 류하는 모델을 개발하고 해당 모델을 기반으로 평가 데이터(test)를 적용하여
# 래 제출 양식에 맞추어 CSV 파일로 제출하시오.
# 단, 예측결과는 악성종양(0)일 확률값을 제출하시오.
# 예측결과는 Accuracy, ROC-AUC 지표로 평가함.
# (실제 시험에서는 test셋의 y값이 주어지지 않으나,
# 해당 모의고사에서는 본인이 직접 y_test 값을 이용하여 성능평가 해보시오)
# 제출형식 (아래 예시)
# csv파일명 : result.csv
# 예측 종양 칼럼명 : pred
# 제출 칼럼 개수 : pred 칼럼 1개 (아래)
```

```
 예시자료(result 결과값과 같은 형태로 제출)
# import pandas as pd
# result = pd.DataFrame({'pred':[0.7, 0.2, 0.3]})
# result
# # 아래 코드는 예시이며 변수명 등 개인별로 변경하여 활용
# pd.DataFrame변수.to_csv("result.csv", index=False)

# 데이터 설명
# 1. 변수설명
# 종속변수(Y) : target 칼럼(0 : 악성종양, 1 : 양성종양)
# 독립변수(X) : no, target을 제외한 칼럼
# 종양번호 : no 칼럼
```

```
###############  실기환경    복사    영역  ###############
import pandas as pd
import numpy as np
# 실기    시험    데이터셋으로    셋팅하기 (수정금지)
from sklearn.datasets import    load_breast_cancer
```

```
# 유방암    데이터셋 load
cancer = load_breast_cancer()
X, y = load_breast_cancer(return_X_y=    True, as_frame=True)
# 실기    시험    데이터셋으로    셋팅하기 (수정금지)
from sklearn.model_selection import train_test_split
x_tr, x_te, y_tr, y_te = train_test_split(X, y, test_size=0.2
stratify=y, random_state=2024)
test = pd.DataFrame(x_te.reset_index())
y_test = pd.DataFrame(y_te.reset_index())
x_tr = pd.DataFrame(x_tr.reset_index())
y_tr = pd.DataFrame(y_tr.reset_index())
test.rename(columns={'index':'no'}, inplace=True)
x_tr.rename(columns={'index':'no'}, inplace=True)
y_test.columns = ['no', 'target']
y_tr.columns = ['no1', 'target']
train = pd.concat([y_tr, x_tr], axis=1)
train = train.drop('no1', axis=1)
y_test = y_test.drop('no', axis=1)
train.loc[0:2, 'mean radius'] = np.nan
###############  실기환경    복사    영역  ###############
```

```
 print(x_tr.head())
 print(y_tr.head())
```

```
no  mean radius      mean texture    mean perimeter  mean a
rea    mean smoothness  \\
0  516       18.310         20.58          120.80     1052.
0          0.10680

1   71        8.888         14.64           58.79      244.
0          0.09783

2  277       18.810         19.98          120.90     1102.
0          0.08923

3   10       16.020         23.24          102.70      797.
```

```
8          0.08206

4   20       13.080         15.71           85.63      520.
0          0.10750

       mean compactness  mean concavity  mean concave point
s  mean symmetry  ...  \
0           0.12480         0.15690             0.09451
0.1860  ...

1           0.15310         0.08606             0.02872
0.1902  ...

2           0.05884         0.08020             0.05843
0.1550  ...

3           0.06669         0.03299             0.03323
0.1528  ...

4           0.12700         0.04568             0.03110
0.1967  ...

     worst radius   worst texture  worst perimeter  worst a
rea  worst smoothness  \
0        21.860          26.20           142.20      1493.0
0.1492

1         9.733          15.67            62.56       284.4
0.1207

2        19.960          24.30           129.00      1236.0
0.1243

3        19.190          33.88           123.80      1150.0
0.1181

4        14.500          20.49            96.09       630.5
0.1312
```

```
       worst compactness  worst concavity  worst concave po
ints  worst symmetry  \
0              0.2536           0.3759               0.15100
0.3074

1              0.2436           0.1434               0.04786
0.2254

2              0.1160           0.2210               0.12940
0.2567

...
1   71       1
2  277       0
3   10       0
4   20       1
```

```
#print(x_tr.head())
#print(y_tr.head())
```

no1 target
0 516 0
1 71 1
2 277 0
3 10 0
4 20 1

```
x_tr = x_tr.drop(columns='no')
y_tr = y_tr.drop(columns='no1')

y_te = y_te.drop(columns='no')
```

```
print(x_tr.isnull().sum())
print(y_tr.isnull().sum())
print(x_te.isnull().sum())
```

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(random_state=2024, penalty=None)
```

```
model.fit(x_tr, y_tr)
pred = model.predict(x_te)
```

`pred`

array([1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1,
1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1,
1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1,
0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1,
0, 1, 0, 1])

`pred = pd.DataFrame({'pred':pred})`

0 1
1 0
2 1
3 1
4 0
..
109 1
110 0
111 1
112 0
113 1
Name: target, Length: 114, dtype: int32

```
from sklearn.metrics import classification_report, confusion_

rpt = classification_report(y_te, pred)
matrix = confusion_matrix(y_te, pred)
y_result_proba = model.predict_proba(x_te)

print(rpt)
print(matrix)
print(np.round(y_result_proba,2))
```

```
              precision    recall  f1-score   support

           0       0.92      0.83      0.88        42
```

```
           1       0.91      0.96      0.93        72

accuracy                           0.91       114
macro avg       0.91      0.90      0.90       114
weighted avg       0.91      0.91      0.91       114

[[35  7]
 [ 3 69]]
[[0.14 0.86]
 [0.92 0.08]
 [0.   1.  ]
 [0.   1.  ]
 [0.49 0.51]
 [0.11 0.89]
 [0.57 0.43]
 [0.01 0.99]
 [0.01 0.99]
 [0.76 0.24]
 [1.   0.  ]
 [0.01 0.99]
 [0.   1.  ]
 [1.   0.  ]
 ...
 [1.   0.  ]
 [0.34 0.66]
 [1.   0.  ]
 [0.   1.  ]]
```

```
# 모의고사 2유형
# 2유형 2번(당뇨 진척정도 예측)
# 다음은 당뇨 환자에게서 얻은 데이터셋이다.
# 학습 데이터(train)를 이용하여 환자의 당뇨 진척 정도를 예측하는 모델을
# 개발하고 해당 모델을 기반으로 평가 데이터(test)를 적용하여 얻은 예측결과
# 아래 제출 양식에 맞추어 CSV 파일로 제출하시오.
# 예측결과는 Rsq, MSE 지표로 평가하시오.
```

```
# 제출형식 (아래 예시)
# csv파일명 : result.csv
# 예측 진척정도 칼럼명 : pred
# 제출 칼럼 개수 : pred 칼럼 1개 (아래)

# 데이터 설명
# 1. 변수설명
# 환자번호 : cust_id 칼럼
# 종속변수(Y) : target 칼럼
# 독립변수(X) : 그 외 칼럼
# 1. 학습/평가 데이터
# train : 학습 데이터
# test : 평가 데이터(taget 변수 제외)
# y_test : 평가 데이터(target 변수 only) / 단, 실제 시험에서는 주어지
```

```
###############  실기환경 복사 영역  ###############
import pandas as pd
import numpy as np
# 실기 시험 데이터셋으로 셋팅하기 (수정금지)
from sklearn.datasets import load_diabetes
# diabetes 데이터셋 로드
diabetes = load_diabetes()
X, y = load_diabetes(return_X_y= True, as_frame=True)

# 실기 시험 데이터셋으로 셋팅하기 (수정금지)
from sklearn.model_selection import train_test_split
x_tr, x_te, y_tr, y_te = train_test_split(X, y, test_size=0.2
                                          random_state=2024)
test = pd.DataFrame(x_te)
x_tr = pd.DataFrame(x_tr)
y_tr = pd.DataFrame(y_tr)
y_tr.columns = ['target']
y_test = pd.DataFrame(y_te)
train = pd.concat([y_tr, x_tr], axis=1)
train = train.reset_index().rename(columns={'index':'cust_id'
test = test.reset_index().rename(columns={'index':'cust_id'})
train['sex'] = np.where(train['sex']>0, 'F', 'M')
test['sex'] = np.where(test['sex']>0, 'F', 'M')
```

```
train.loc[0:2, 'age'] = np.nan
test.loc[0:2, 'age'] = np.nan
###############  실기환경 복사 영역  ###############
```

`train.head()`

`test.head()`

`train.shape`

(353, 12)

`test.shape`

(89, 11)

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

`print(test.isnull().sum())`

```
median1 = train['age'].median()
median2 = test['age'].median()

train['age'] = train['age'].fillna(median1)
test['age'] = test['age'].fillna(median2)
```

```
print(train.isnull().sum())
print(test.isnull().sum())
```

```
train['sex'] = train['sex'].replace('M',0)   # 회귀문제,svm에서는
train['sex'] = train['sex'].replace('F',1)     # 1,2,3이 회귀알
```

```
from sklearn.model_selection import train_test_split

x = train.drop(columns=['cust_id','target'])
y = train['target']
x_train, x_val, y_train, y_val = train_test_split(x, y, test_

print(x_train.shape)
print(y_train.shape)
print(x_val.shape)
```

```
print(y_val.shape)
```

```
from sklearn.ensemble import RandomForestRegressor

rfr = RandomForestRegressor(random_state=2024)
model = rfr.fit(x_train, y_train)
pred = model.predict(x_val)
```

```
from sklearn.metrics import r2_score, mean_squared_error

r2 = r2_score(y_val, pred)
mse = mean_squared_error(y_val, pred)

print(r2, mse)
```

0.48709310473636735 3199.9673887640447

```
# 모의고사 2유형
# 2유형 3번(와인 종류 분류)
# 제공된 데이터는 와인의 조성 데이터를 기반으로 해당 와인이 어떤 종류의 와인
# 터셋이다. 학습 데이터(train)를 이용하여 와인의 종류을 분류하는 모델을 개
# 기반으로 평가 데이터(test)를 적용하여 얻은 예측결과를 아래 제출 양식에 맞
# 출하시오.
# 예측결과는 Accuracy, macro F1 score 지표로 평가함
# 제출형식 (아래 예시)
#  csv파일명 : result.csv
# 예측 진척정도 칼럼명 : pred
# 제출 칼럼 개수 : pred 칼럼 1개 (아래)

# # 예시자료(result 결과값과 같은 형태로 제출)
#  import pandas as pd
#  result = pd.DataFrame({'pred':[1,0,2]})
#  result
```

```
#  # 아래 코드는 예시이며 변수명 등 개인별로 변경하여 활용
# # pd.DataFrame변수.to_csv("result.csv", index=False)

# 데이터 설명
# 1. 변수설명
# class : 와인의 종류, 종속변수(0, 1, 2)
# 그 외 변수 : 와인의 특징을 나타내는 독립변수(Xs)
#  1. 학습/평가 데이터
# train : 학습 데이터
# test : 평가 데이터(taget 변수 제외)
#  y_test : 평가 데이터(target 변수 only) / 단, 실제 시험에서는 주어지
```

```
###############  실기환경 복사 영역  ###############
import pandas as pd
import numpy as np

# 실기 시험 데이터셋으로 셋팅하기 (수정금지)
from sklearn.datasets import load_wine
# 와인 데이터셋 load
X, y = load_wine(return_X_y= True, as_frame=True)

# 실기 시험 데이터셋으로 셋팅하기 (수정금지)
from sklearn.model_selection import train_test_split
x_tr, x_te, y_tr, y_te = train_test_split(X, y, test_size=0.2
                                          stratify=y,
                                          random_state=2024)
test = pd.DataFrame(x_te)
x_tr = pd.DataFrame(x_tr)
y_tr = pd.DataFrame(y_tr)
y_tr.columns = ['class']
y_test = pd.DataFrame(y_te)
train = pd.concat([y_tr, x_tr], axis=1)

train['hue'] = np.where(train['hue']>=1, 'H', 'L')
test['hue'] = np.where(test['hue']>=1, 'H', 'L')
train.iloc[0:3, train.columns.get_loc('alcohol')] = np.nan
```

```
test.iloc[0:3, test.columns.get_loc('alcohol')] = np.nan
###############  실기환경 복사 영역  ###############
```

`train.head()`

`test.head()`

`train.shape`

`test.shape`

`train.isnull().sum()`

`test.isnull().sum()`

```
median1 = train['alcohol'].median()
median2 = test['alcohol'].median()

train['alcohol'] = train['alcohol'].fillna(median1)
test['alcohol'] = test['alcohol'].fillna(median2)
```

```
print(train.isnull().sum())
print(test.isnull().sum())
```

train['hue'].unique()

test['hue'].unique()

```
train = pd.get_dummies(train)
test = pd.get_dummies(test)
print(train.head())
print(test.head())
```

```
from sklearn.model_selection import train_test_split

x = train.drop(columns='class')
y = train['class']
```

```
x_train, x_val, y_train, y_val = train_test_split(x, y, test_

print(x_train.shape)
print(y_train.shape)
```

```
print(x_val.shape)
print(y_val.shape)
```

```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(random_state=2024)
model = rfc.fit(x_train, y_train)
pred = model.predict(x_val)
```

```
from sklearn.metrics import accuracy_score, classification_re

acc = accuracy_score(y_val, pred)
rpt = classification_report(y_val, pred)
matrix = confusion_matrix(y_val, pred)
```

`print(acc, rpt, matrix)`

```
pred2 = model.predict(test)

acc2 = accuracy_score(y_test, pred2)
rpt2 = classification_report(y_test, pred2)
matrix2 = confusion_matrix(y_test, pred2)

print(acc2, rpt2, matrix2)
```