

06

다항 로지스틱 회귀분석 실습

[다항 로지스틱 회귀분석 실습 - 데이터 분할]

01:54

다항 로지스틱 회귀분석은 종속변수의 level이 3개 이상일 때

사용합니다.

6.4 다항 로지스틱 회귀모형 실습

[예제]
Iris 데이터의 Species를 분류하는 다항 로지스틱 회귀분석을 실시하고 오분류표를 만들어라

```
In [1]: 1 import pandas as pd
        2 iris = pd.read_csv('../data/iris.csv')
        3
        4 X = iris.drop(['target'],axis=1)
        5 y = iris.target
```

In [2]: 1 X

Out [2]:

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows x 4 columns

In [3]: 1 # 훈련셋과 테스트셋 분리하기

[모델 생성]

모델을 생성하고 학습시키는 부분은 쉽습니다.

- 모델안에 어떠한 매개변수를 넣을 수 있는지 궁금하다면 `help(model)` 값으로 확인해보세요!^^

03:43

```

Name: target, dtype: int64

In [18]:
1 # 훈련셋 평가셋 분리하기
2 from sklearn.model_selection import train_test_split
3 X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, train_size=0.7,
4 test_size=0.3, random_state=23)
5 print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)

(105, 4) (45, 4) (105,) (45,)

In [20]:
1 y_test.value_counts()

Out[20]:
iris-setosa    15
iris-virginica 15
iris-versicolor 15
Name: target, dtype: int64

6.4.1 sklearn 모델 생성
• sklearn 모델은 2 레이블을 이용하여 전통적 통계모델에서 다중공선성의 문제를 내부적으로 해결해준다.
• 독립변수 간의 상관성이 높은 변수라면, 2레벨이 0에 가깝게하여 변수를 삭제하는 것과 같은 효과를 보인다.

In [21]:
1 # 모델 학습하기
2 from sklearn.linear_model import LogisticRegression
3
4 model = LogisticRegression()
5 model.fit(X_train, y_train)
6 print(model)

LogisticRegression()

C:\Users\mj\anaconda3\envs\ADP_Class\lib\site-packages\sklearn\linear_model\logistic.py:765: ConvergenceWarning: lbfgs failed to c
onverge (status=1):
  http://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
  http://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
extra_warning_msg=LOGISTIC_SOLVER_CONVERGENCE_MSG)

```

[모델 평가]

모델을 평가할 때, 멀티클래스 분류라면, roc_auc_score의 매개변수를 변경해주셔야 합니다.

05:46

```

In [23]:
1 # 모델 학습하기
2 from sklearn.linear_model import LogisticRegression
3
4 model = LogisticRegression()
5 model.fit(X_train, y_train)
6 print(model)

LogisticRegression()

C:\Users\mj\anaconda3\envs\ADP_Class\lib\site-packages\sklearn\linear_model\logistic.py:765: ConvergenceWarning: lbfgs failed to c
onverge (status=1):
  http://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
  http://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
extra_warning_msg=LOGISTIC_SOLVER_CONVERGENCE_MSG)

6.4.2 모델 평가

In [5]:
1 from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, roc_auc_score
2 # 테스트셋 예측
3 predicted = model.predict(X_test)
4
5 # 오분류율 생성
6 cm = confusion_matrix(y_test, predicted)
7 cmtb = pd.DataFrame(cm, columns=['predicted_setosa', 'predicted_versicolor', 'predicted_virginica'],
8 index = ['setosa', 'versicolor', 'virginica'])
9
10 cmtb

Out[5]:
      predicted_setosa  predicted_versicolor  predicted_virginica
setosa              15                  0                  0
versicolor           0                  14                  1
virginica            0                  0                  15

```

[회귀계수 해석]

11:05

File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (pykernel)

```

>>> roc_auc_score(y, clf.predict_proba(X), multi_class='ovr')
0.99...

Multilabel case:

>>> from sklearn.datasets import make_multilabel_classification
>>> from sklearn.multiclass import MultiOutputClassifier
>>> X, y = make_multilabel_classification(random_state=0)
>>> clf = MultiOutputClassifier(clf, fit(X, y))
>>> # get a list of n_output containing probability arrays of shape

```

6.4.3 다항 로지스틱 회귀 계수 해석

```

In [8]: 1 # 회귀계수 확인하기
        2 print('Intercept: %n', model.intercept_)
        3 print('Coefficient: %n', model.coef_)

Intercept:
[ 9.42940015  2.10066933 -11.53006848]
Coefficient:
[[ -0.45747795  0.87062087 -0.30469796 -0.86053751]
 [ 0.31575053 -0.19466078 -0.16297032 -0.75289644]
 [ 0.08169622 -0.6779661  2.47137828  1.71343395]]

In [9]: 1 # 오즈비 계산하기
        2 import numpy as np
        3 np.exp(model.coef_)

Out[9]: array([[ 0.63297835,  2.39316921,  0.09941941,  0.38268713],
               [ 1.45612796,  0.82011089,  0.8496164 ,  0.47100095],
               [ 1.08512612,  0.50764845, 11.83875275,  5.54796029]])

In [10]: 1 pd.DataFrame(np.exp(model.coef_), columns=X_train.columns, index = model.classes_)

Out[10]:

```

	sepal length	sepal width	petal length	petal width
lvs-setosa	0.632978	2.393169	0.099419	0.382687
lvs-versicolor	1.456128	0.820114	0.849616	0.471000