fieldtrip.

분류 분석

[로지스틱 회귀 분석] 특징 데이터로 유방암 진단하기

Unit 1 목표설정

❖ **목표:** 유방암 특징을 측정한 데이터에 로지스틱 회귀 분석을 수행하여 유방암 발생을 예측

특징 데이터로 유방암 진단하기	
목표	로지스틱 회귀 분석을 이용해 유방암에 영향을 미치는 특징 데이터를 분석하고 유방암 여부를 진단하는 예측 모델을 생성한다.
핵심 개념	로지스틱 회귀, 시그모이드 함수, 성능 평가 지표, 오차 행렬, 정밀도, 재현율, F1 스코어, ROC 기반 AUC 스코어
데이터 준비	유방암 진단 데이터: 사이킷런 내장 데이터셋
데이터 탐색	1. 사이킷런 데이터셋에서 제공하는 설명 확인: b_cancer.DESCR 2. 사이킷런 데이터셋에 지정된 X 피처와 타깃 피처 결합 3. 로지스틱 회귀 분석을 위해 X 피처 값을 정규 분포 형태로 스케일링: b_cancer_scaled = scaler.fit_transform(b_cancer.data)
분석 모델 구축	사이킷런의 로지스틱 회귀 모델 구축
결과 분석	성능 평가 지표 계산: confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, roc_auc_score

Unit 1

데이터 준비 및 탐색

❖ 사이킷런에서 제공하는 데이터셋

데이터셋	샘플 갯수	독립 변수	종속 변수	데이터 로드 함수
보스톤 주택 가격 데이터	506	13개	주택 가격	load_boston()
붓꽃(아이리스) 데이터	150	4개	붓꽃 종류: setosa, versicolor, virginica	load_iris()
당뇨병 환자 데이터	442	10개	당뇨병 수치	load_diabetes()
숫자 0~9를 손으로 쓴 흑백 데이터	1797	64개	숫자: 0~9	load_digits()
와인의 화학 성분 데이터	178	13개	와인 종류: 0, 1, 2	load_wine()
체력 검사 데이터	20	3개	체력 검사 점수	load_linnerud()
유방암 진단 데이터	569	30개	악성(malignant), 양성(benign): 1, 0	load_ breast_ cancer()

Unit 1

데이터 준비 및 탐색

❖ 사이킷런의 유방암 진단 데이터셋 사용하기

1. 데이터 준비하기

In [1]:	**import** numpy **as** np **import** pandas **as** pd **from** sklearn.datasets **import load_breast_cancer**
In [2]:	b_cancer = **load_breast_cancer()**

In [1]: 사이킷런에서 제공하는 데이터셋sklearn.datasets중에서 유방암 진단 데이터셋을 사용하기 위해 load_breast_cancer를 임포트

In [2]: 데이터셋을 로드하여 객체b_cancer를 생성

[로지스틱 회귀 분석] 특징 데이터로 유방암 진단하기

데이터 준비 및 탐색

❖ 사이킷런의 유방암 진단 데이터셋 사용하기
2. 데이터 탐색하기

In [3]:
```
print(b_cancer.DESCR)
```

In [4]:
```
b_cancer_df = pd.DataFrame(b_cancer.data, columns = b_cancer.feature_names)
```

In [5]:
```
b_cancer_df['diagnosis']= b_cancer.target
```

In [6]:
```
b_cancer_df.head()
```

Out:[6]

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374

5 rows × 31 columns

In [3]: 데이터셋에 대한 설명을 확인
In [4]: 데이터셋 객체의 data 배열b_cancer.data, 즉 독립 변수 X가 되는 피처를 DataFrame 자료형으로 변환하여 b_cancer_df를 생성
In [5]: 유방암 유무 class로 사용할 diagnosis 컬럼을 b_cancer_df에 추가하고 데이터셋 객체의 target 컬럼 b_cancer.target을 저장
In [6]: b_cancer_df의 데이터 샘플 5개를 출력b_cancer_df.head( )하여 확인

[로지스틱 회귀 분석] 특징 데이터로 유방암 진단하기

데이터 준비 및 탐색

❖ 사이킷런의 유방암 진단 데이터셋 사용하기

3. 데이터셋의 크기와 독립 변수 X가 되는 피처에 대한 정보를 확인

In [7]:	print('유방암 진단 데이터셋 크기: ', b_cancer_df.shape)
Out:[7]	유방암 진단 데이터셋 크기: (569, 31)
In [8]:	b_cancer_df.head()
Out:[8]	<class 'pandas.core.frame.DataFrame'> RangeIndex: 569 entries, 0 to 568 Data columns (total 31 columns): mean radius 569 non-null float64 mean texture 569 non-null float64 mean perimeter 569 non-null float64 mean area 569 non-null float64 mean smoothness 569 non-null float64 mean compactness 569 non-null float64 mean concavity 569 non-null float64 mean concave points 569 non-null float64 mean symmetry 569 non-null float64 mean fractal dimension 569 non-null float64 radius error 569 non-null float64 texture error 569 non-null float64 perimeter error 569 non-null float64 area error 569 non-null float64 ....

Out:[8]	.... smoothness error 569 non-null float64 compactness error 569 non-null float64 concavity error 569 non-null float64 concave points error 569 non-null float64 symmetry error 569 non-null float64 fractal dimension error 569 non-null float64 worst radius 569 non-null float64 worst texture 569 non-null float64 worst perimeter 569 non-null float64 worst area 569 non-null float64 worst smoothness 569 non-null float64 worst compactness 569 non-null float64 worst concavity 569 non-null float64 worst concave points 569 non-null float64 worst symmetry 569 non-null float64 worst fractal dimension 569 non-null float64 diagnosis 569 non-null int32 dtypes: float64(30), int32(1) memory usage: 135.7 KB

In [7]: b_cancer_df.shape를 사용하여 데이터셋의 행의 개수(데이터 샘플 개수)와 열의 개수(변수 개수)를 확인
행의 개수가 569이므로 데이터 샘플이 569개, 열의 개수가 31이므로 변수가 31개 있음
In [8]: b_cancer_df에 대한 정보를 확인b_cancer_df.info( ) / 30개의 피처(독립 변수 X) 이름과 1개의 종속 변수 이름을 확인 가능 diagnosis는 악성이면 1, 양성이면 0의 값이므로 유방암 여부에 대한 이진 분류의 class로 사용할 종속 변수가 됨

데이터 준비 및 탐색

❖ 사이킷런의 유방암 진단 데이터셋 사용하기

4. 로지스틱 회귀 분석에 피처로 사용할 데이터를 평균이 0, 분산이 1이 되는 정규 분포 형태로 맞춤

In [9]:	from sklearn.preprocessing import StandardScaler scaler = StandardScaler()
In [10]:	b_cancer_scaled = scaler.fit_transform(b_cancer.data)
In [11]:	print(b_cancer.data[0])
Out:[11]	[1.799e+01 1.038e+01 1.228e+02 1.001e+03 1.184e-01 2.776e-01 3.001e-01 1.471e-01 2.419e-01 7.871e-02 1.095e+00 9.053e-01 8.589e+00 1.534e+02 6.399e-03 4.904e-02 5.373e-02 1.587e-02 3.003e-02 6.193e-03 2.538e+01 1.733e+01 1.846e+02 2.019e+03 1.622e-01 6.656e-01 7.119e-01 2.654e-01 4.601e-01 1.189e-01]
In [12]:	print(b_cancer_scaled[0])
Out:[12]	[1.09706398 -2.07333501 1.26993369 0.9843749 1.56846633 3.28351467 2.65287398 2.53247522 2.21751501 2.25574689 2.48973393 -0.56526506 2.83303087 2.48757756 -0.21400165 1.31686157 0.72402616 0.66081994 1.14875667 0.90708308 1.88668963 -1.35929347 2.30360062 2.00123749 1.30768627 2.61666502 2.10952635 2.29607613 2.75062224 1.93701461]

In [9]: 사이킷런의 전처리 패키지에 있는 정규 분포 스케일러를 임포트하고 사용할 객체scaler를 생성
In [10]: 피처로 사용할 데이터b_cancer.data에 대해 정규 분포 스케일링을 수행scaler.fit_ transform( )하여 b_cancer_scaled에 저장
In [11]~[12]: 정규 분포 스케일링 후에 값이 조정된 것을 확인

[로지스틱 회귀 분석] 특징 데이터로 유방암 진단하기

분석 모델 구축 및 결과 분석

1. 로지스틱 회귀를 이용하여 분석 모델 구축하기

In [13]:	**from** sklearn.linear_model import **LogisticRegression** **from** sklearn.model_selection **import** train_test_split
In [14]:	*#X, Y 설정하기* Y = b_cancer_df['diagnosis'] X = b_cancer_scaled
In [15]:	*#훈련용 데이터와 평가용 데이터 분할하기* X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3, random_state = 0)
In [16]:	*#로지스틱 회귀 분석: (1) 모델 생성* lr_b_cancer = **LogisticRegression()**
In [17]:	*#로지스틱 회귀 분석: (2) 모델 훈련* lr_b_cancer.fit(X_train, Y_train)
Out:[17]	LogisticRegression()
In [18]:	*#로지스틱 회귀 분석: (3) 평가 데이터에 대한 예측 수행 -> 예측 결과 Y_predict 구하기* Y_predict = lr_b_cancer.**predict**(X_test)

In [13]: 필요한 모듈을 임포트
In [14]: diagnosis를 Y, 정규 분포로 스케일링한 b_cancer_scaled를 X로 설정
In [15]: 전체 데이터 샘플 569개를 학습 데이터:평가 데이터=7:3으로 분할test_size=0.3함
In [16]: 로지스틱 회귀 분석 모델 객체lr_b_cancer를 생성
In [17]: 학습 데이터X_train, Y_train로 모델 학습을 수행fit( )함
In [18]: 학습이 끝난 모델에 대해 평가 데이터 XX_test를 가지고 예측을 수행predict( )하여 예측값 YY_predict를 구함

[로지스틱 회귀 분석] 특징 데이터로 유방암 진단하기

분석 모델 구축 및 결과 분석

2. 생성한 모델의 성능 확인하기

In [19]:	from sklearn.metrics import confusion_matrix, accuracy_score from sklearn.metrics import precision_score, recall_score, f1_score, roc_auc_score
In [20]:	*#오차 행렬* confusion_matrix(Y_test, Y_predict)
Out:[20]	array([[60, 3], [1, 107]], dtype = int64)
In [21]:	acccuracy = accuracy_score(Y_test, Y_predict) precision = precision_score(Y_test, Y_predict) recall = recall_score(Y_test, Y_predict) f1 = f1_score(Y_test, Y_predict) roc_auc = roc_auc_score(Y_test, Y_predict)
In [22]:	print('정확도: {0:.3f}, 정밀도: {1:.3f}, 재현율: {2:.3f}, F1: {3:.3f}'.format(acccuracy,precision,recall,f1))
Out:[22]	정확도: 0.977, 정밀도: 0.973, 재현율: 0.991, F1: 0.982
In [23]:	print('ROC_AUC: {0:.3f}'.format(roc_auc))
Out:[23]	ROC_AUC: 0.972

In [19]: 필요한 모듈을 임포트
In [20]: 평가를 위해 7:3으로 분할한 171개의 test 데이터에 대해 이진 분류의 성능 평가 기본이 되는 오차 행렬을 구함
실행 결과를 보면 TN이 60개, FP가 3개, FN이 1개, TP가 107개인 오차 행렬이 구해짐
In [21]: 성능 평가 지표인 정확도, 정밀도, 재현율, F1 스코어, ROC-AUC 스코어를 구함
In [22]~[23]: 성능 평가 지표를 출력하여 확인

Demos

지도학습 분류문제의 예 -붓꽃 분류

- 다음은 분류의 한 예로 scikit-learn 패키지에서 제공하는 붓꽃(iris) 분류 문제를 보였다. 이 문제는 붓꽃의 꽃받침 길이(sepal length), 꽃받침 폭(sepal width), 꽃잎 길이(petal length), 꽃잎 폭(petal width)을 이용하여 붓꽃의 세가지 종류(setosa, versicolor, virginica) 중 어느 것에 속하는지를 결정하는 문제이다.
- 아래에는 파이썬을 이용하여 붓꽃 데이터의 일부와 이를 시각화한 모습을 보였다.

```
import numpy as np
import seaborn as sns
from sklearn.datasets import load_iris

iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
sy = pd.Series(iris.target, dtype="category")
sy = sy.cat.rename_categories(iris.target_names)
df['species'] = sy

np.random.seed(0)
df.sample(frac=1).reset_index(drop=True).head(10)
```

- 데이터셋 요약

Classes	3
Samples per class	50
Samples total	150
Dimensionality	4
Features	real, positive

지도학습 분류문제의 예 -붓꽃 분류

❖ ...

In [9]:
```
import numpy as np
from sklearn.datasets import load_iris

iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
sy = pd.Series(iris.target, dtype="category")
sy = sy.cat.rename_categories(iris.target_names)
df['species'] = sy

np.random.seed(0)
df.sample(frac=1).reset_index(drop=True).head(10)
```

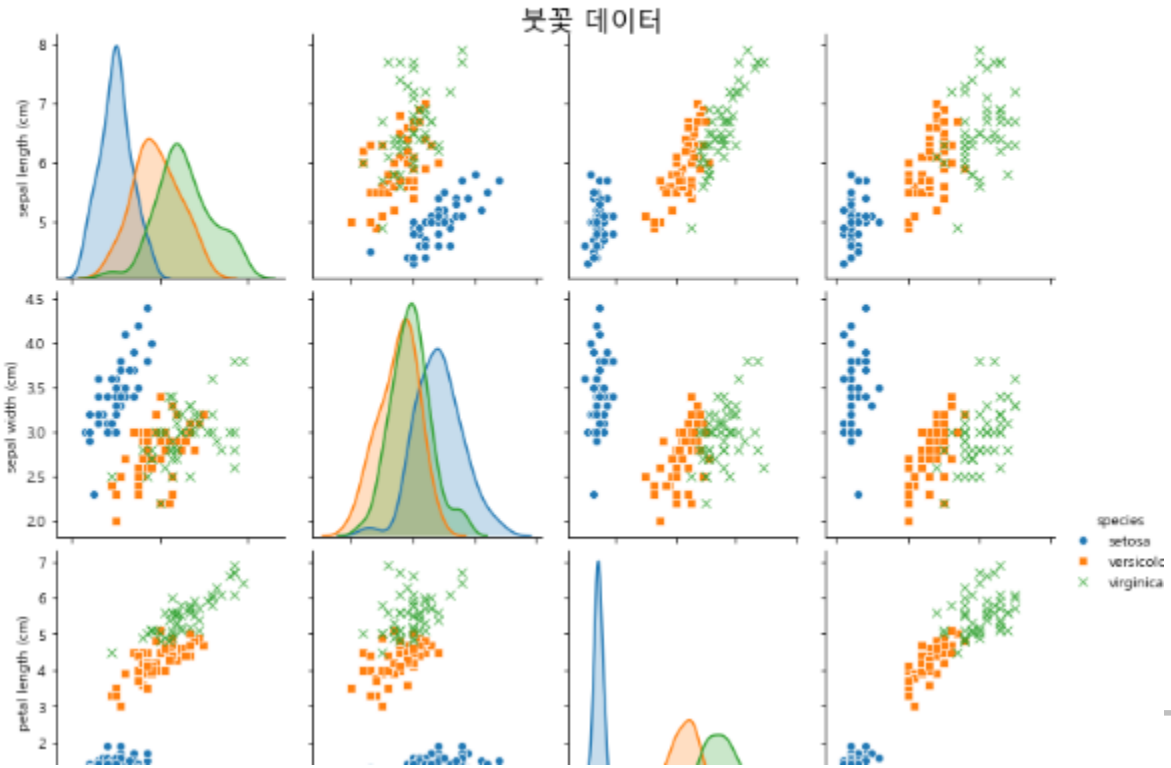
Out [9]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.8	2.8	5.1	2.4	virginica
1	6.0	2.2	4.0	1.0	versicolor
2	5.5	4.2	1.4	0.2	setosa
3	7.3	2.9	6.3	1.8	virginica
4	5.0	3.4	1.5	0.2	setosa
5	6.3	3.3	6.0	2.5	virginica
6	5.0	3.5	1.3	0.3	setosa
7	6.7	3.1	4.7	1.5	versicolor
8	6.8	2.8	4.8	1.4	versicolor
9	6.1	2.8	4.0	1.3	versicolor

지도학습 분류문제의 예 -붓꽃 분류

```
sns.pairplot(df, hue="species", markers=["o", "s", "X"]) # markers=["o", "s", "D","X","v"]
plt.suptitle("붓꽃 데이터", y=1.02, fontsize=18)
plt.savefig('iris.png')
plt.show()
```

```
In [75]: sns.pairplot(df, hue="species", markers=["o", "s", "x"])
         plt.suptitle("붓꽃 데이터", y=1.02, fontsize=18)
         plt.show()
```



지도학습 분류문제의 예 -붓꽃 분류

❖ 이 문제를 서포트 벡터 머신(SVC: Support Vector Machine)이라는 분류 모형으로 풀면 다음과 같다. 이 그림에서 하나의 점은 하나의 표본 데이터를 가리키고, 다른 색으로 칠해진 영역은 서포트 벡터 머신이 다른 종으로 분류하고 있는 영역을 뜻한다.

```
import matplotlib as mp

from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC

features = [2, 3]
X = iris.data[:, features]
y = iris.target

model = SVC(kernel="linear", random_state=0)
model.fit(X, y)

XX_min = X[:, 0].min() - 1
XX_max = X[:, 0].max() + 1
YY_min = X[:, 1].min() - 1
YY_max = X[:, 1].max() + 1
XX, YY = np.meshgrid(np.linspace(XX_min, XX_max, 1000),
                     np.linspace(YY_min, YY_max, 1000))
ZZ = model.predict(np.c_[XX.ravel(), YY.ravel()]).reshape(XX.shape)
```

지도학습 분류문제의 예 -붓꽃 분류

```
cmap = mp.colors.ListedColormap(['seashell', 'lightgreen', 'lightskyblue'])
plt.contourf(XX, YY, ZZ, cmap=cmap)
plt.contour(XX, YY, ZZ, colors='k')
plt.scatter(X[y == 0, 0], X[y == 0, 1], s=20, label=iris.target_names[0],
            marker="o", edgecolors="darkred", facecolors="red")
plt.scatter(X[y == 1, 0], X[y == 1, 1], s=20, label=iris.target_names[1],
            marker="s", edgecolors="darkgreen", facecolors="green")
plt.scatter(X[y == 2, 0], X[y == 2, 1], s=30, label=iris.target_names[2],
            marker="x", edgecolors="darkblue", facecolors="blue")
plt.xlim(XX_min, XX_max)
plt.ylim(YY_min, YY_max)
plt.xlabel("꽃잎의 길이(cm)")
plt.ylabel("꽃잎의 폭(cm)")
plt.title("서포트벡터머신을 이용한 붓꽃 분류 결과")
plt.legend(loc="lower right", framealpha=1)
plt.show()
```