사이킷런(sklearn)

1. 사이킷런(sklearn) 개요
2. 회귀분석(boston 주택가격 예측하기)
3. 분류 분석(iris 붓꽃분류)

1. 사이킷런(sklearn) 개요

❖ 사이킷런

- 파이썬으로 머신러닝을 수행하기 위한 쉽고 효율적인 개발 라이브러리를 제공
- 다양한 분류, 회귀, 서포트 벡터 머신, 랜덤 포레스트, 그라디언트 부스팅, k-평균, DBSCAN을 포함한 클러스터링 알고리즘을 지원
- 파이썬의 수치 및 과학 라이브러리 NumPy 및 SciPy와 함께 운용되도록 설계
- 다양한 머신러닝 분석용 데이터셋 을 제공

  - load_boston: 보스톤 집값 데이터
  - load_iris: 아이리스 붓꽃 데이터
  - load_diabetes: 당뇨병 환자 데이터
  - load_digits: 손글씨 데이터
  - load_linnerud: multi-output regression 용 데이터
  - load_wine: 와인 데이터
  - load_breast_cancer: 위스콘신 유방암 환자 데이터

1. 사이킷런(sklearn) 개요

❖ 분석 평가 지표

- 회귀 분석 결과에 대한 평가 지표는 예측값과 실제값의 차이인 오류의 크기가 됨
- 정확한 평가를 위해 오류의 절대값 평균이나 제곱의 평균, 제곱 평균의 제곱근 또는 분산 비율 을 사용

평가 지표	수식	사이킷런 라이브러리
MAE: Mean Absolute Error	$\frac{1}{n}\sum_{i=1}^{n}\lvert Y_i-\hat{Y}_i\rvert$	metrics.mean_absolute_error()
MSE: Mean Squared Error	$\frac{1}{n}\sum_{i=1}^{n}(Y_i-\hat{Y}_i)^2$	metrics.mean_squared_error()
RMSE: Root Mean Squared Error	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i-\hat{Y}_i)^2}$	없음
R^2: Variance score, 결정 계수coefficient of determination	$\frac{\text{예측값의 분산}}{\text{실제값의 분산}}$	metrics.r2_score()

2. 회귀분석(boston 주택가격 예측하기)

❖ 데이터 수집, 준비 및 탐색

- 주택가격분석'으로 노트북 페이지를 추가하고 입력

In [1]:	!pip install sklearn # 사이킷런 설치
In [2]:	import **numpy** as **np** import **pandas** as **pd** from **sklearn.datasets** import **load_boston** boston = **load_boston()** **print(boston)**

In [2]: 사이킷런에서 제공하는 데이터셋sklearn.datasets 중에서 보스톤 주택 가격 데이터셋을 사용하기 위해 load_boston을 임포트하고, 데이터셋을 로드하여load_boston( ) 객체boston를 생성

2. 회귀분석(boston 주택가격 예측하기)

❖ 데이터 수집, 준비 및 탐색

- 데이터가 이미 정리된 상태이므로 데이터셋 구성을 확인

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

In [4]:
```
boston_df = pd.DataFrame(boston.data, columns = boston.feature_names)
boston_df.head()
```

Out[4]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33

In [5]:
```
boston_df['PRICE'] = boston.target
boston_df.head()
```

Out[5]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

In [3]: 데이터셋에 대한 설명boston.DESCR을 확인

In [4]: 데이터셋 객체의 data 배열 boston.data, 즉 독립 변수 X가 되는 피처들을 DataFrame 자료형으로 변환하여 boston_df를 생성, boston_df의 데이터 5개를 확인bostone_df.head( )

In [5]: 데이터셋 객체의 target 배열 boston.target, 즉 종속 변수인 주택 가격('PRICE') 컬럼을 boston_df에 추가 boston_df의 데이터 5개를 확인 bostone_df.head( )

2. 회귀분석(boston 주택가격 예측하기)

❖ 데이터 수집, 준비 및 탐색

- 데이터가 이미 정리된 상태이므로 데이터셋 구성을 확인

In [6]:

```
print('보스톤 주택 가격 데이터셋 크기: ', boston_df.shape)
```

Out[6]:

```
보스톤 주택 가격 데이터셋 크기: (506, 14)
```

In [7]:

```
boston_df.info()
```

Out[7]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
CRIM        506 non-null float64
ZN          506 non-null float64
INDUS       506 non-null float64
CHAS        506 non-null float64
NOX         506 non-null float64
RM          506 non-null float64
AGE         506 non-null float64
DIS         506 non-null float64
RAD         506 non-null float64
TAX         506 non-null float64
PTRATIO     506 non-null float64
B           506 non-null float64
LSTAT       506 non-null float64
PRICE       506 non-null float64
dtypes: float64(14)
memory usage: 55.4KB
```

In [6]: 데이터셋의 형태 boston_df.shape, 즉 행의 개수(데이터 개수)와 열의 개수(변수 개수)를 확인,
행의 개수가 506이므로 데이터가 506개 있으며, 열의 개수가 14이므로 변수가 14개 있음
변수 중에서 13개는 독립 변수 X가 되고, 마지막 변수 'PRICE'는 종속 변수 Y가 됨

In [7]: boston_df에 대한 정보를 확인

- 14개의 독립 변수(피처)의 의미
  - CRIM: 지역별 범죄 발생률
  - ZN: 25,000평방피트를 초과하는 거주 지역 비율
  - INDUS: 비상업 지역의 넓이 비율
  - CHAS: 찰스강의 더미변수(1은 강의 경계, 0은 경계 아님)
  - NOX: 일산화질소 농도
  - RM: 거주할 수 있는 방 개수
  - AGE: 1940년 이전에 건축된 주택 비율
  - DIS: 5개 주요 고용센터까지 가중 거리
  - RAD: 고속도로 접근 용이도
  - TAX: 10,000달러당 재산세 비율
  - PTRATIO: 지역의 교사와 학생 수 비율
  - B: 지역의 흑인 거주 비율
  - LSTAT: 하위 계층의 비율
  - PRICE(MEDV): 본인 소유 주택 가격의 중앙값

2. 회귀분석(boston 주택가격 예측하기)

❖ 분석 모델 구축, 결과 분석 및 시각화

- 선형 회귀를 이용해 분석 모델 구축하기
  - 사이킷런의 선형 분석 모델 패키지sklearn.linear_model에서 선형 회귀LinearRegression를 이용하여 분석 모델을 구축

In [8]:	from sklearn.linear_model import **LinearRegression** from sklearn.model_selection import **train_test_split** from sklearn.metrics import **mean_squared_error, r2_score)**

In [9]:	#X, Y 분할하기
	Y = boston_df['PRICE']
	X = boston_df.**drop**(['PRICE'], axis = 1, inplace = False)

In [10]:	*#훈련용 데이터와 평가용 데이터 분할하기* X_train, X_test, Y_train, Y_test = **train_test_split**(X, Y, test_size = 0.3, random_state = 156)

In [8]: 사이킷런을 사용하여 머신러닝 회귀 분석을 하기 위한 LinearRegression과 데이터셋 분리 작업을 위한 train_test_split, 성능 측정을 위한 평가 지표인 mean_squared_ error, r2_score를 임포트

In [9]: PRICE 피처를 회귀식의 종속 변수 Y로 설정하고 PRICE를 제외 drop( )한 나머지 피처를 독립 변수 X로 설정

In [10]: X와 Y 데이터 506개를 학습 데이터와 평가 데이터로 7:3 비율로 분할test_size=0.3

2. 회귀분석(boston 주택가격 예측하기)

❖ 분석 모델 구축, 결과 분석 및 시각화

- 선형 회귀를 이용해 분석 모델 구축하기
  - 사이킷런의 선형 분석 모델 패키지sklearn.linear_model에서 선형 회귀LinearRegression를 이용하여 분석 모델을 구축

In [11]:	#선형 회귀 분석 : 모델 생성
	lr = **LinearRegression()**

In [12]:	#선형 회귀 분석 : 모델 훈련 lr.**fit**(X_train, Y_train)
Out[12]:	LinearRegression()

In [13]:	#선형 회귀 분석 : 평가 데이터에 대한 예측 수행 -> 예측 결과 Y_predict 구하기
	Y_predict = lr.**predict**(X_test)

In [11]: 선형 회귀 분석 모델 객체 lr을 생성

In [12]: 학습 데이터 XX_train와 YY_train를 가지고 학습을 수행fit( ).

In [13]: 평가 데이터 XX_test를 가지고 예측을 수행하여predict( ) 예측값YY_predict를 구함

2. 회귀분석(boston 주택가격 예측하기)

❖ 분석 모델 구축, 결과 분석 및 시각화

- 선형 회귀를 이용해 분석 모델 구축하기
  - 선형 회귀 분석 모델을 평가 지표를 통해 평가하고 회귀 계수를 확인하여 피처의 영향을 분석

In [14]:	mse = **mean_squared_error**(Y_test, Y_predict) rmse = np.**sqrt**(mse) print('MSE : {0:.3f}, RMSE : {1:.3f}'.format(mse, rmse)) print('R^2(Variance score) : {0:.3f}'.format(**r2_score**(Y_test, Y_predict)))
Out[14]:	MSE : 17.297, RMSE : 4.159 R^2(Variance score) : 0.757
In [15]:	print('Y 절편 값: ', lr.**intercept_**) print('회귀 계수 값: ', np.round(lr.**coef_**, 1))
Out[15]:	Y 절편 값: 40.995595172164336 회귀 계수 값: [-0.1 0.1 0. 3. -19.8 3.4 0. -1.7 0.4 -0. -0.9 0. -0.6]

In [14]: 회귀 분석은 지도 학습이므로 평가 데이터 X에 대한 결과값 YY_test를 이미 알고 있는 상태에서 평가 데이터 YY_test와 In [13]에서 구한 예측 결과Y_predict의 오차를 계산하여 모델을 평가. 평가 지표 MSE를 구하고mean_squared_error( ) 구한 값의 제곱근을 계산하여np.sqrt(mse) 평가 지표 RMSE를 구함 그리고 평가 지표 R2 을 구함r2_score( )

In [15]: 선형 회귀의 Y절편 lr.intercept_과 각 피처의 회귀 계수 lr.coef_를 확인

2. 회귀분석(boston 주택가격 예측하기)

❖ 분석 모델 구축, 결과 분석 및 시각화

- 선형 회귀를 이용해 분석 모델 구축하기
  - 선형 회귀 분석 모델을 평가 지표를 통해 평가하고 회귀 계수를 확인하여 피처의 영향을 분석

In [16]:
```
coef = pd.Series(data = np.round(lr.coef_, 2), index = X.columns)
coef.sort_values(ascending = False)
```

Out[16]:
```
RM       3.35
CHAS     3.05
RAD      0.36
ZN       0.07
INDUS    0.03
B        0.01
AGE      0.01
TAX     -0.01
CRIM    -0.11
LSTAT   -0.57
PTRATIO -0.92
DIS     -1.74
NOX     -19.80
dtype: float64
```

In [16]: 회귀 모델에서 구한 회귀 계수 값lr.coef_과 피처 이름X.columns을 묶어서 Series 자료 형으로 만들고, 회귀 계수 값을 기준으로 내림차순으로 정렬하여 ascending=False 확인 sort_ values( )

- 회귀 모델 결과를 토대로 보스톤 주택 가격에 대한 회귀식

YPRICE = -0.11XCRIM + 0.07XZN + 0.03XINDUS + 3.05XCHAS – 19.80XNOX + 3.35XRM + 0.01XAGE - 1.74XDIS + 0.36XRAD - 0.01XTAX - 0.92XPTRATIO + 0.01XB - 0.57XLSTAT + 41.00

2. 회귀분석(boston 주택가격 예측하기)

❖ 회귀 분석 결과를 산점도 + 선형 회귀 그래프로 시각화하기

- 선형 회귀를 이용해 분석 모델 구축하기

In [17]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [18]:
```
fig, axs = plt.subplots(figsize = (16, 16), ncols = 3, nrows = 5)

x_features = ['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX', 'PTRATIO', 'B', 'LSTAT']

for i, feature in enumerate(x_features):
  row = int(i/3)
  col = i%3
  sns.regplot(x = feature, y = 'PRICE', data = boston_df, ax = axs[row][col])
```

In [17]: 시각화에 필요한 모듈을 임포트

In [18]: 독립 변수인 13개 피처와 종속 변수인 주택 가격, PRICE와의 회귀 관계를 보여주는 13개 그래프를 subplots()를 사용하여 5행 3열 구조로 모아서 나타냄

seaborn의 regplot()은 산점도 그래프와 선형 회귀 그래프를 함께 그려줌

2. 회귀분석(boston 주택가격 예측하기)

❖ 회귀 분석 결과를 산점도 + 선형 회귀 그래프로 시각화하기

- 선형 회귀를 이용해 분석 모델 구축하기

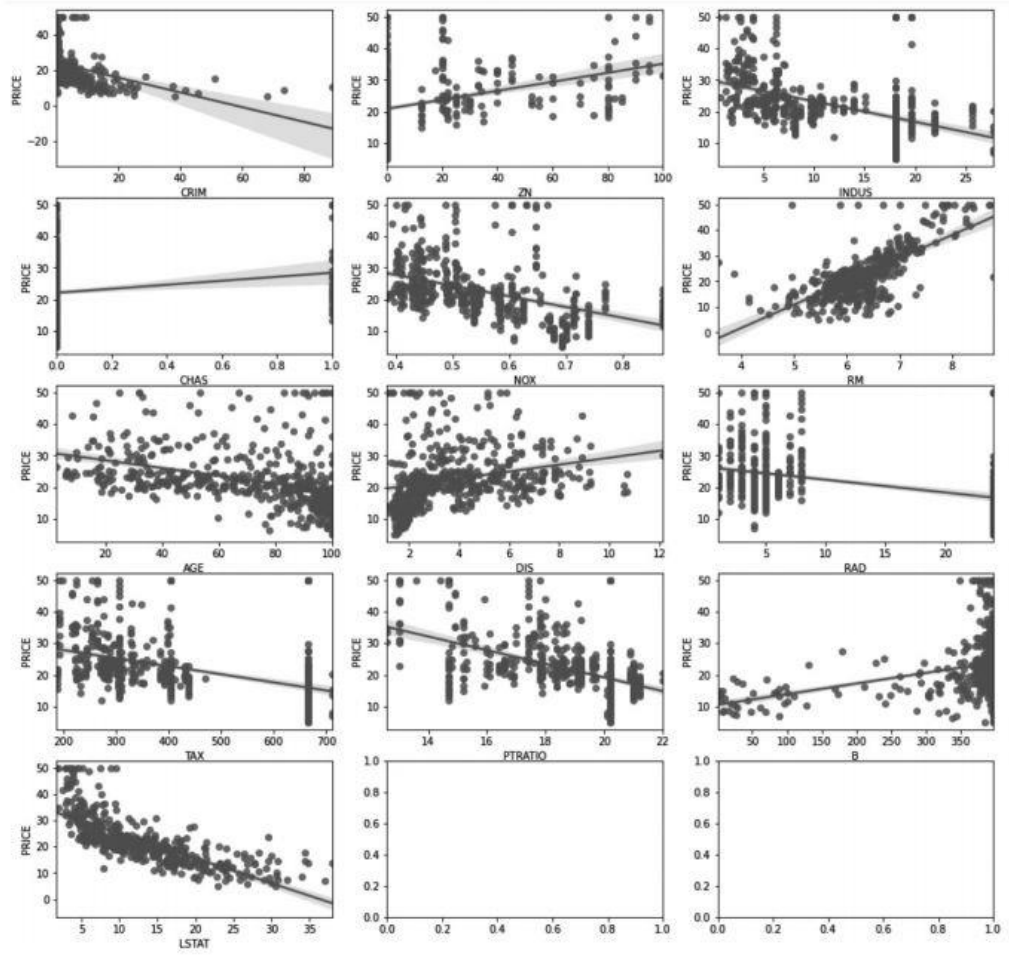

그림 10-3 13개 피처와 주택 가격의 회귀 관계를 나타낸 산점도/선형 회귀 그래프

1. 분류 분석(iris 붓꽃분류)

❖ 라이브러리 및 데이터 셋 로딩

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn.metrics import confusion_matrix  #오차행렬은 sklearn.metircs 패키지 내 confusion_matrix로 확인 가능
```

```
iris = load_iris() #lord_iris import
iris_data = iris.data #중요 데이터 iris_data 변수에 저장 후 데이터 크기 확인
iris_label = iris.target #iris_label변수에 iris 데이터의 target저장
```

```
print(iris)
print(iris_data)
print(iris_label)
```

❖ 훈련데이터와 테스트 데이터 분류

```
#(3)train, test 데이터 분리
 #X:feature데이터만 / y:정답label데이터만
 #X데이터셋을 머신러닝 모델에 입력 -> 모델이 내뱉는 품종 예측 결과를 정답 y와 비교하여 학습
X_train, X_test, y_train, y_test = train_test_split(iris_data,  #feature(입력받는 특징 데이터)
                                                    iris_label,  #label(모델이 맞춰야하는 정답값)
                                                    test_size=0.2,  #test dataset 크기 조절(0.2=전체 20%를 test데이터로 사용)
                                                    random_state=7)  #train데이터와 test데이터 분리시 적용되는 랜덤성)
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

❖ 모델 훈련 및 테스트

```
logistic_model = LogisticRegression()
logistic_model.fit(X_train, y_train)
y_pred = logistic_model.predict(X_test)
scores=metrics.accuracy_score(y_test, y_pred)
print(scores)

print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```