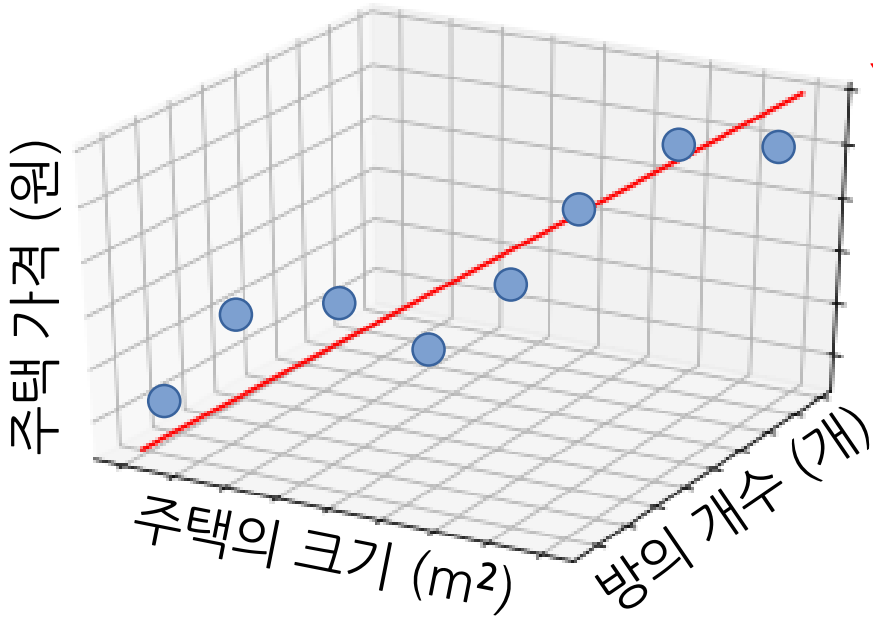
다중 선형 회귀

다중 선형 회귀

- 다중 선형 회귀 (Multiple Linear Regression)
  - 독립변수가 2개 이상이고 종속변수도 1개인 경우, 그들 간의 관계를 선형적으로 파악하는 회귀 방식
  - 독립변수들 $X_1, X_2, X_3, \ldots, X_n$과 종속변수 Y의 관계를 $Y = w_0 + w_1X_1 + w_2X_2 + w_3X_3 + \ldots + w_nX_n$ 형태의 1차 함수식으로 표현할 수 있다.



$$Y = w_0 + w_1 \cdot X_1 + w_2 \cdot X_2$$

다중 선형 회귀

- 다중 선형 회귀
  - 단순 선형 회귀에서 독립변수의 개수만 늘어난 것이다.
  - 따라서 단순 선형 회귀와 동일한 절차를 이용하여 분석을 수행할 수 있다.
  - 단, 변수의 수가 많아지므로 이로 인해 발생할 수 있는 경우들을 고려해서 적절한 조치를 취해야 한다.

다중 선형 회귀

- 독립변수들의 최초 선택
  - 회귀 분석의 목적은 종속변수를 가장 잘 설명하는 독립변수들의 성향/특징을 찾아내어 이를 기반으로 기존의 자료를 설명하거나 새로운 결과를 예측하는 것이다.
  - 즉, 독립변수 일부를 임의로 누락시키는 것은 해당 모형의 설명력이 낮아지는 문제를 불러올 수 있다.
  - 따라서 통계(분석)적으로 회귀 분석을 수행하는 경우, 관련 있는 독립변수들을 일단 가급적 모두 고려하는 것이 바람직하다.

다중 선형 회귀

- 다중 공선성 (Multi-collinearity) 문제
  - 독립변수들 간에 강한 상관 관계가 나타나는 것을 다중 공선성이라고 한다. (즉, 어떤 독립변수의 값이 독립적이지 않고 다른 독립변수(들)의 값에 의해서 결정된다는 것이다.)
  - 독립변수들끼리의 상관 계수 R이 높으면 (아무리 설명력이 좋다고 하더라도) 회귀 모형이 유의미하다고 보기 어렵다.
  - 따라서 상관 계수가 높은 변수들을 삭제하거나, 주성분 분석(PCA) 기법 등을 이용하여 의존적인 성분을 제거한 뒤 회귀 분석을 수행해야 한다.

다중 선형 회귀

- 상관 계수 R
  - 공분산을 각각의 표준편차로 나누어 정규화한 수치
    - 변수 X, Y에 대하여 각각의 크기(단위)에 영향을 받지 않도록 단위를 보정한 것이라고 볼 수 있다.

$$R = \frac{Cov(X, Y)}{\sigma_X \times \sigma_Y}$$

(이 때, σ_X는 X의 표준편차, σ_Y는 Y의 표준편차이다.)

    - 상관 계수의 값은 $-1 \leq R \leq 1$이며, 1에 가까울수록 강한 양(+) 의 상관 관계, -1에 가까울수록 강한 음(-)의 상관 관계이다. 0이면 서로 상관 관계가 없다.

다중 선형 회귀

- 독립변수들의 설명력과 수정된 결정 계수
  - 독립변수의 개수가 많아질수록 그 변수들이 종속변수에 끼치는 영향력은 늘어나게 된다. (즉, 독립변수가 많을수록 종속변수에 대한 설명력은 증가한다.)
  - 따라서 다중 회귀 분석에서는 결정 계수 R^2의 값이 단순 회귀보다 높게 나오는 경향이 있고, 이는 독립변수의 수가 많아질수록 더욱 증가한다.
  - 이를 보완하여 수정된 결정 계수를 도입했으며, 다중 회귀에서는 일반적으로 수정 결정 계수의 값을 이용하여 분석 결과를 판단한다.

다중 선형 회귀

- 결정 계수 R^2
  - 회귀식이 얼마나 설명력이 있는지 (즉, 얼마나 정확한지) 나타내는 지표이다.

$$R^2 = \frac{\text{예측값의 분산}}{\text{실제값의 분산}} = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{RSS}{\sum(y_i - \bar{y})^2}$$

(이 때, $\hat{y}_i$는 실제값 y_i에 대한 예측값, $\bar{y}$는 실제값들의 평균이다.)

  - 결정 계수의 값은 $0 \leq R^2 \leq 1$이며, 1에 가까울수록 설명력이 강하고 0에 가까울수록 설명력이 약하다.
  - 일반적으로 결정 계수 R^2의 값이 0.65 (65%) 이상이면 설명력이 있다고 판단한다.

다중 선형 회귀

- 수정 결정 계수 (Adjusted R^2)
  - 결정 계수의 값이 커지는 것을 보정하기 위해 데이터(표본)의 크기와 독립변수의 개수를 고려하여 계산한 지표이다.

$$Adj\_R^2 = 1 - \frac{(1 - R^2) \times (N - 1)}{(N - k - 1)}$$

(이 때, N은 데이터의 개수, k는 독립변수의 개수이다.)

  - 수정 결정 계수의 값은 항상 결정 계수 R^2보다 작거나 같은 값이다.
  - (모든 독립변수가 아니라) 종속변수에 영향을 주는 독립변수들만으로 설명되는 분산의 비율이라고 볼 수 있다.

다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 보스턴 주택 가격 데이터를 불러와서 정보를 확인한다.

```
1  import sklearn.datasets as d
2
3  boston = d.load_boston()
4  print(boston.DESCR)
```

여기에서는 결과의 일부만 보였다.

```
- CRIM     per capita crime rate by town
- ZN       proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS    proportion of non-retail business acres per town
- CHAS     Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX      nitric oxides concentration (parts per 10 million)
- RM       average number of rooms per dwelling
- AGE      proportion of owner-occupied units built prior to 1940
- DIS      weighted distances to five Boston employment centres
- RAD      index of accessibility to radial highways
- TAX      full-value property-tax rate per $10,000
- PTRATIO  pupil-teacher ratio by town
- B        1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT    % lower status of the population
- MEDV     Median value of owner-occupied homes in $1000's
```

다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 데이터를 DataFrame으로 변환한다.

```
1  import pandas as pd
2
3  # 특성(독립변수)들을 DataFrame으로 변환한다.
4  boston_df = pd.DataFrame(boston.data, ₩
5                           columns=boston.feature_names)
6
7  # 종속변수를 덧붙인다.
8  boston_df["PRICE"] = boston.target
9
10 # 데이터의 형태와 크기를 확인한다.
11 print(boston_df.shape)
12
13 # 데이터의 일부를 확인해 본다.
14 boston_df.head()
```

```
(506, 14)
```

마지막 라인의 실행 결과는 생략하였다.

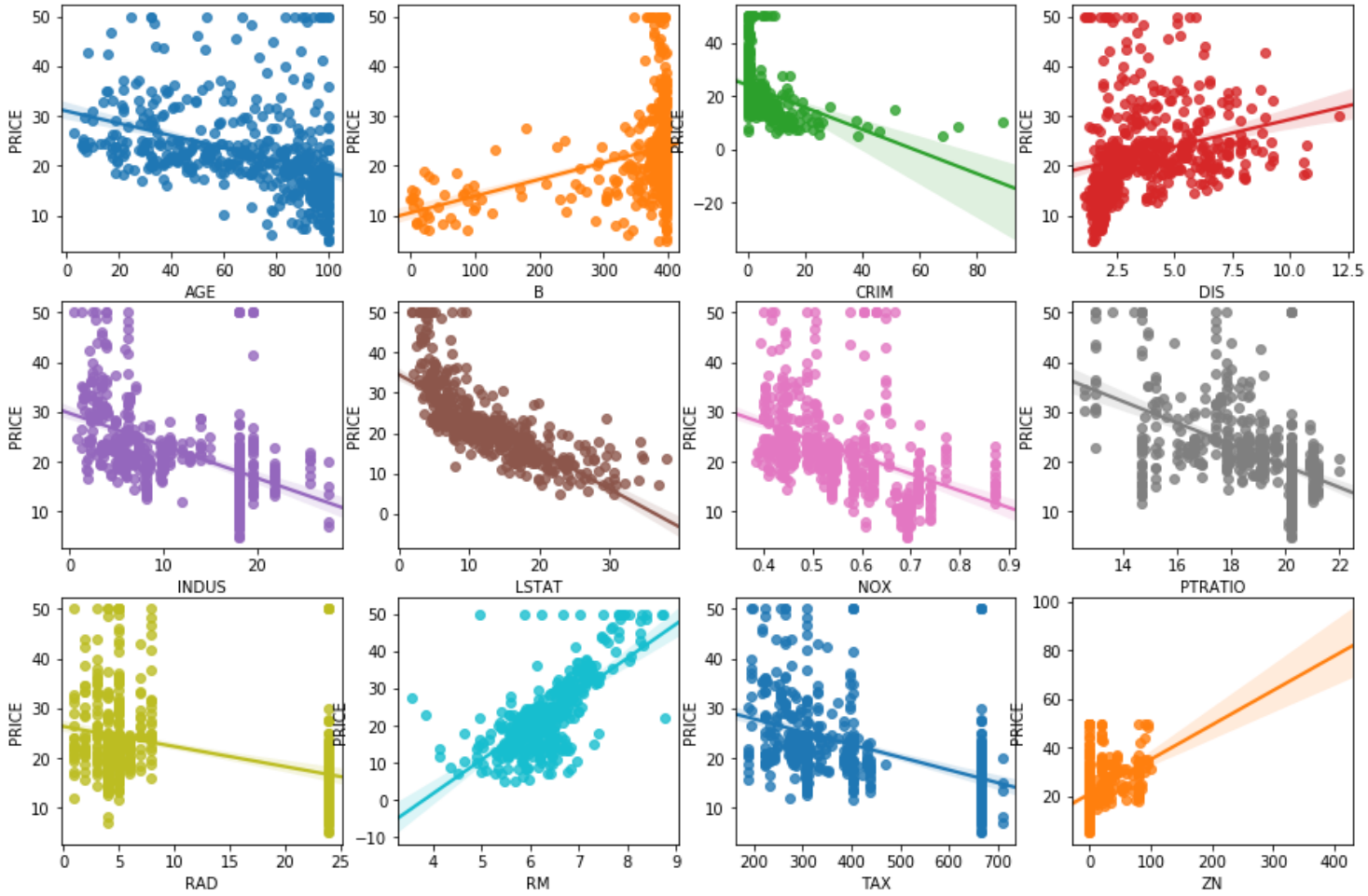
다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 컬럼 별로 종속변수와의 관계에 대한 산점도를 그린다.

```
import matplotlib.pyplot as plt
import seaborn as sns

# 3행 4열의 subplot를 준비한다.
fig, axs = plt.subplots(figsize=(15,10), nrows=3, ncols=4)

# 전체 컬럼들에서 종속변수 및 불필요한 컬럼을 뺀다.
features = [f for f in list(boston_df.columns)[:-1] if f != "CHAS"]
features.sort()

for i, feature in enumerate(features):
    # 현재 subplot의 행/열 번호를 갱신한다.
    r = int(i / 4)
    c = i % 4
    # 산점도와 회귀선을 그린다.
    sns.regplot(x=feature, y="PRICE", data=boston_df, ax=axs[r][c])
```

다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 산점도와 회귀선을 바탕으로, 각 변수들의 영향을 확인한다.



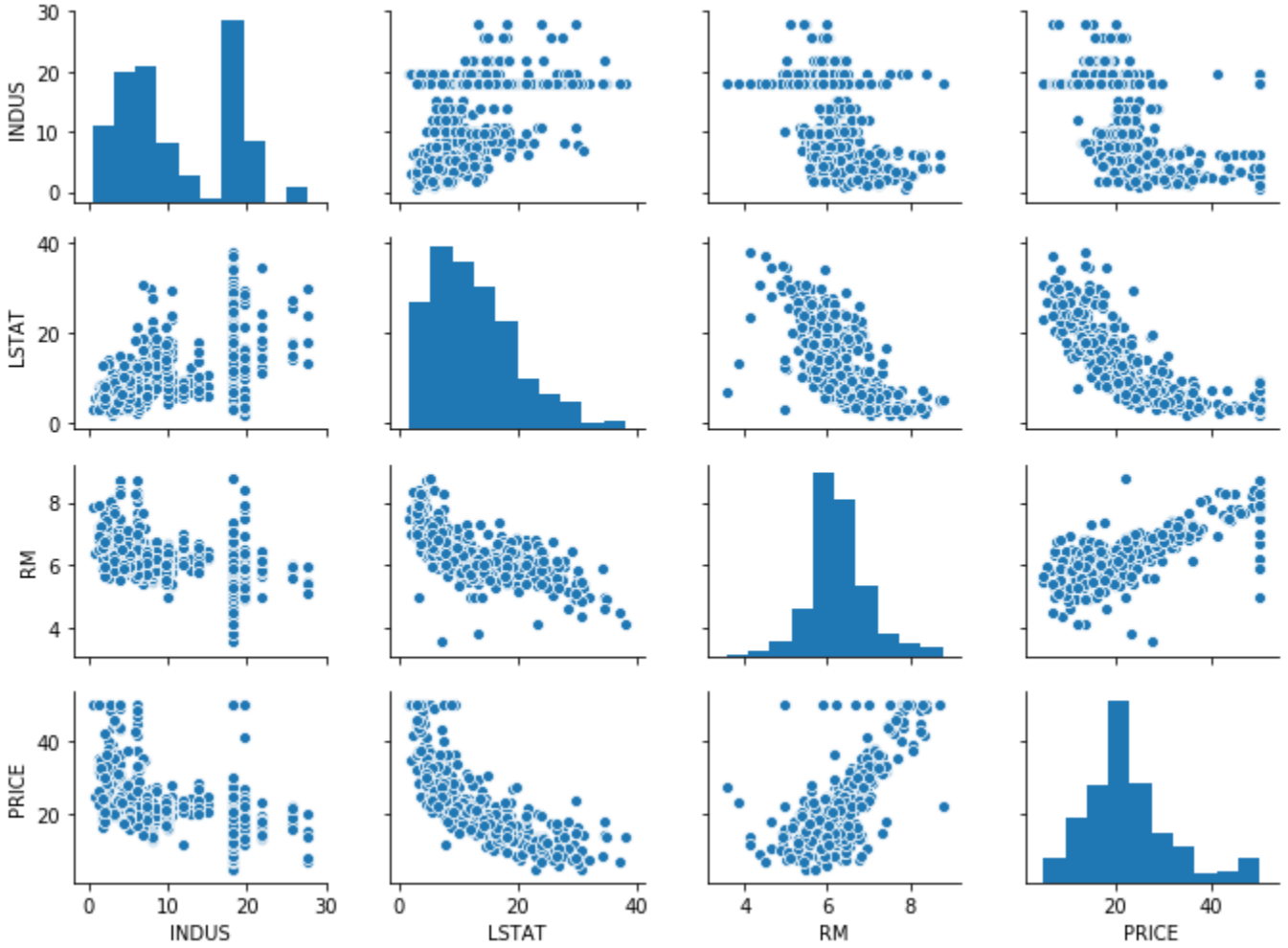
다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 컬럼 별로 독립변수들 간의 관계에 대한 플롯을 그린다.

```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3
4  # 전체 컬럼들에서 상호간의 관계를 파악할 변수들을 선정한다.
5  ccol = ["INDUS", "LSTAT", "RM", "PRICE"]
6
7  # 페어플롯을 그린다.
8  sns.pairplot(boston_df[ccol])
9
10 #또는 매개변수로 컬럼들을 직접 선택해서 그릴 수 있다.
11 sns.pairplot(boston_df, vars=ccol)
```

다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 페어플롯을 바탕으로, 각 변수들끼리의 영향을 확인한다.



다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 컬럼 별로 독립변수들 간의 관계에 대한 히트맵을 그린다.

```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3
4  # 전체 컬럼들에서 상호간의 관계를 파악할 변수들을 선정한다.
5  ccol = ["INDUS", "LSTAT", "RM", "PRICE"]
6
7  # 변수들 간의 상관 계수를 구한다.
8  corrs = boston_df[ccol].corr()
9
10 # 상관 계수 값에 대한 히트맵을 그린다.
11 # 매개변수 annot은 맵 상에 값을 표시할 것인지의 여부를 지정한다.
12 # 매개변수 annot_kws는 표시되는 값에 대한 추가 옵션이다.
13 sns.heatmap(corrs, annot=True, annot_kws={"size": 13})
```

다중 선형 회귀

- 데이터 선정 및 분포 정보 확인
  - 히트맵을 바탕으로, 각 변수들끼리의 영향을 확인한다.

다중 선형 회귀

- 사이킷런으로 다중 선형 회귀 수행
  - 데이터를 학습용과 검증용으로 분리한다.

```
1  import sklearn.model_selection as ms
2
3  # DataFrame에서 독립변수와 종속변수를 다시 구분한다.
4  X = boston_df.drop(["PRICE"], 1)
5  y = boston_df["PRICE"]
6
7  # 학습용 및 검증용 데이터로 분리한다.
8  X_train, X_test, y_train, y_test = ₩
9  ms.train_test_split(X, y, test_size=0.3, random_state=42)
```

```
1  # 데이터의 일부를 확인해 본다.
2  X_train.head()
```

마지막 라인의 실행 결과는 생략하였다.

다중 선형 회귀

- 사이킷런으로 다중 선형 회귀 수행
  - 최소제곱법으로 학습, 예측, 평가를 진행한다.

```
1  import sklearn.linear_model as lm
2  import sklearn.metrics as mt
3  import numpy as np
4
5  # 훈련 데이터로 학습을 수행한다.
6  reg = lm.LinearRegression().fit(X_train, y_train)
7
8  # 검증 데이터로 예측을 수행한다.
9  y_pred = reg.predict(X_test)
10
11 # 평가 지표 값들을 계산한다.
12 mse = mt.mean_squared_error(y_test, y_pred)
13 rmse = np.sqrt(mse)
14 r2 = mt.r2_score(y_test, y_pred)
15
16 # 평가 지표 값들을 출력한다.
17 print("MSE: {:.3f}₩nRMSE: {:.3f}₩nR2: {:.3f}".format(mse, rmse, r2))
```

마지막 라인의 출력 결과는 생략하였다.

다중 선형 회귀

- 사이킷런으로 다중 선형 회귀 수행
  - 추가로 수정 결정 계수를 계산한다.

```
# 데이터의 크기, 즉 레코드 개수
n = len(X_train)

# 독립변수의 개수
k = len(X_train.columns)

# 수정된 결정 계수 값을 계산한다.
adj_r2 = 1 - ((1-r2) * (n-1) / (n-k-1))

# 수정 결정 계수 값을 출력한다.
print("Adjusted R2: {:.4f}".format(adj_r2))
```

```
Adjusted R2: 0.7002
```

다중 선형 회귀

- 사이킷런으로 다중 선형 회귀 수행
  - 회귀 계수와 절편 값을 확인한다.

```
1  # ndarray의 실수값들을 부동소수점으로 출력되도록 옵션을 변경한다.
2  np.set_printoptions(suppress=True)
3
4  # 회귀 계수들 적절히 반올림하여 출력한다.
5  print("회귀 계수:", np.round(reg.coef_, 3))
6
7  # 절편 값을 적절히 반올림하여 출력한다.
8  print("절편:", round(reg.intercept_, 3))
9
10 # ndarray의 실수값 출력 옵션을 원래대로 되돌린다.
11 np.set_printoptions(suppress=False)
```

```
회귀 계수: [ -0.133   0.036   0.05    3.12  -15.417   4.057
             -0.011  -1.386   0.243  -0.009  -0.911   0.012  -0.547]
절편: 31.631
```

다중 선형 회귀

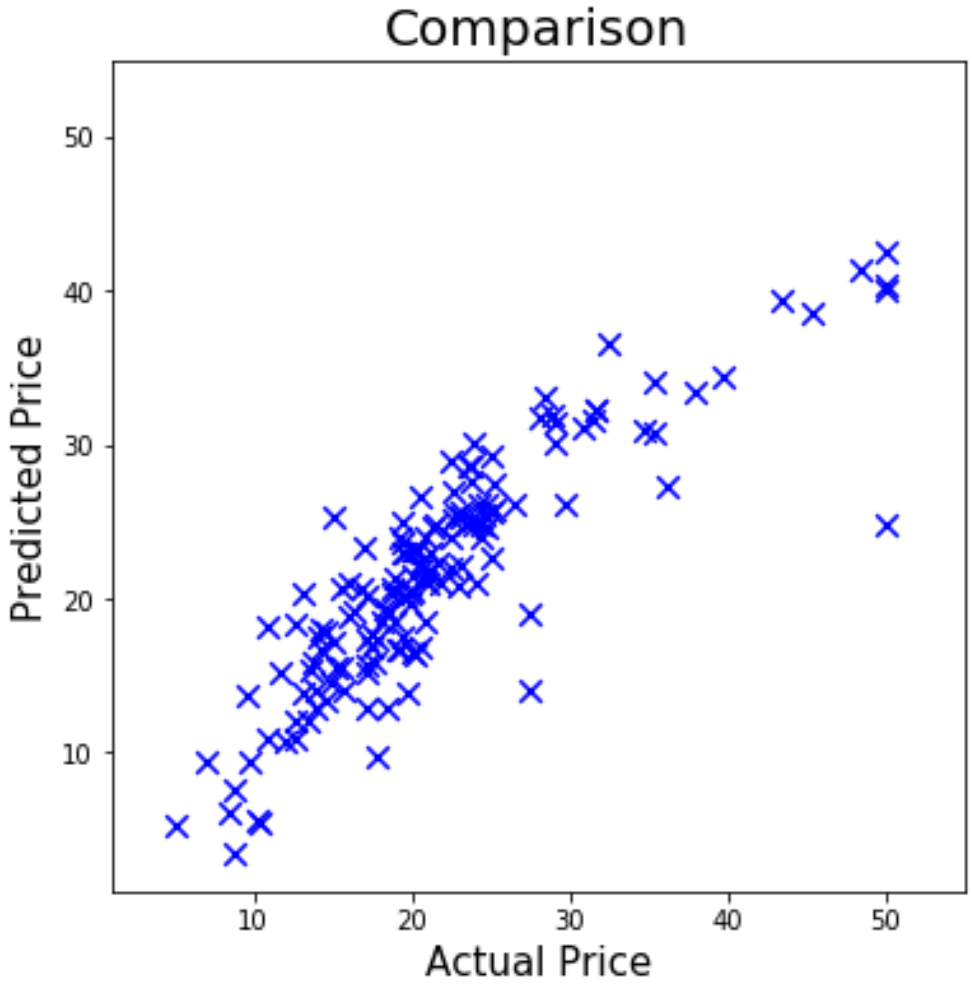
- 사이킷런으로 다중 선형 회귀 수행
  - 회귀 계수를 각 독립변수 이름과 함께 정렬하여 확인한다.

```
# 각 독립변수 이름이 인덱스가 되도록 회귀 계수를 Series로 생성한다.
coefs = pd.Series(data=np.round(reg.coef_, 3), index=X.columns)

# 생성한 Series를 값의 내림차순으로 정렬한다.
coefs_sort = coefs.sort_values(ascending=False)

# 결과를 확인한다.
print(coefs_sort)
```

```
RM          4.057
CHAS        3.120
RAD         0.243

LSTAT      -0.547
PTRATIO    -0.911
DIS        -1.386
NOX       -15.417
```

다중 선형 회귀

- 사이킷런으로 다중 선형 회귀 수행
  - 실제값과 예측값의 분포 차이를 플롯으로 표현해 본다.



다중 선형 회귀

- 스탯츠모델로 다중 선형 회귀 수행
  - 최소제곱법으로 학습, 예측, 평가를 진행한다.

```
1  import statsmodels.api as sm
2
3  # 훈련 데이터로 학습을 수행한다.
4  X_train = sm.add_constant(X_train)
5  reg = sm.OLS(y_train, X_train).fit()
6
7  # 검증 데이터로 예측을 수행한다.
8  X_test = sm.add_constant(X_test)
9  y_pred = reg.predict(X_test)
10
11 # 평가 지표 값들을 출력한다.
12 print(reg.summary())
```

다중 선형 회귀

- 스탯츠모델로 다중 선형 회귀 수행
  - 수정 결정 계수, 회귀 계수, 절편 값 등을 확인한다.

OLS Regression Results

Dep. Variable:	PRICE	R-squared:	0.743
Model:	OLS	Adj. R-squared:	0.734
Method:	Least Squares	F-statistic:	75.81

	coef	std err	t	P>|t|	[0.025	0.975]
const	31.6311	6.056	5.223	0.000	19.720	43.542
CRIM	-0.1335	0.041	-3.271	0.001	-0.214	-0.053
ZN	0.0358	0.018	2.029	0.043	0.001	0.071
INDUS	0.0495	0.073	0.680	0.497	-0.094	0.193
CHAS	3.1198	1.037	3.010	0.003	1.081	5.159
NOX	-15.4171	4.750	-3.246	0.001	-24.759	-6.075
RM	4.0572	0.496	8.181	0.000	3.082	5.033
AGE	-0.0108	0.016	-0.671	0.503	-0.043	0.021