

# IT 파이썬. 딥러닝

인공지능 예측

강사 이수현

## 머신러닝 문제해결 광고 플랫폼에 따른 판매량 예측

- 선형 회귀 알고리즘의 방법을 이해하고, 단순 선형 회귀와 다중 선형 회귀의 성능 평가 후 최적의 회귀식을 구하여 예측할 수 있다.

## 광고 플랫폼에 따른 판매량을 예측해 볼까?

이번 활동에서는 머신러닝 모델 중 **선형 회귀(Linear Regression)**를 이용하여 **광고 플랫폼에 따른 판매량을 예측**해 봅시다.

판매량을 예측하는 데 독립변수 개수를 늘려 단순 선형 회귀와 다중 선형 회귀를 서로 비교하며 광고 플랫폼에 따른 판매량을 예측하는 모델을 만들어 보겠습니다.

문제 정의하기

광고 플랫폼에 따른 판매량은 얼마나 될까요?

데이터 불러오기

캐글에서 광고(advertising) 데이터셋 불러오기

데이터 탐색하기

- 데이터 살펴보기
- 상관관계 분석하기

모델 학습하기

단순 선형 회귀

- 1개의 독립변수와 종속변수 설정하기

단순 선형 회귀로 학습하기

다중 선형 회귀

- 다수의 독립변수와 종속변수 설정하기

다중 선형 회귀로 학습하기

모델 테스트 및  
평가하기

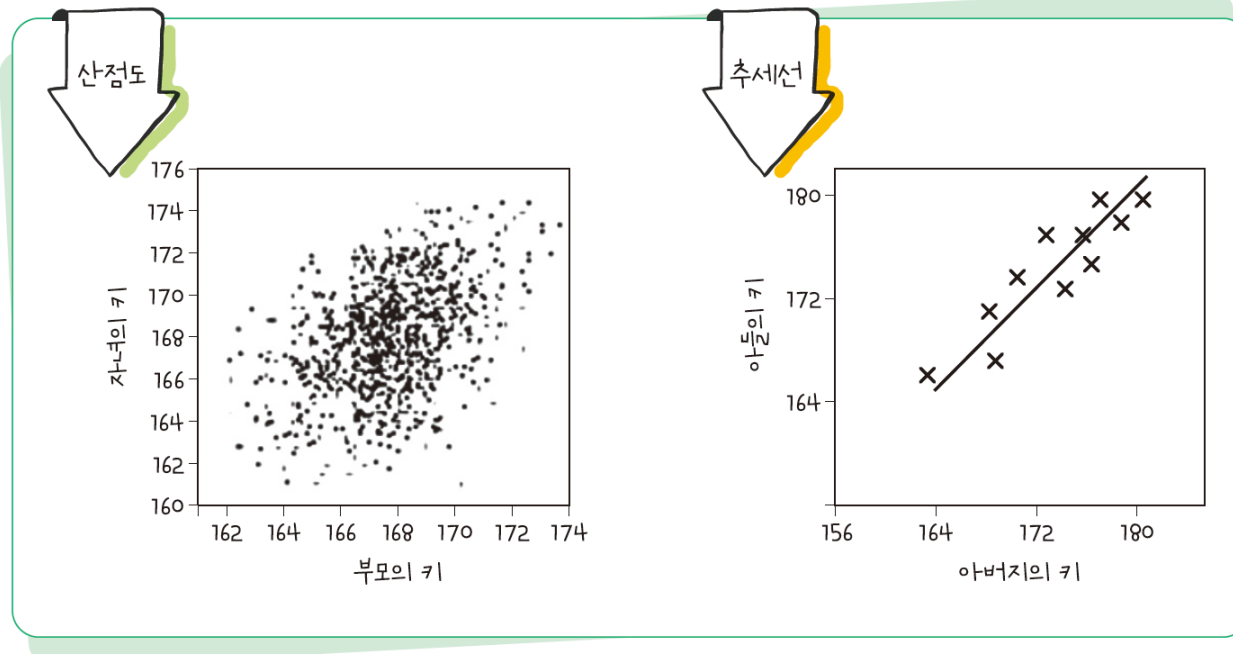
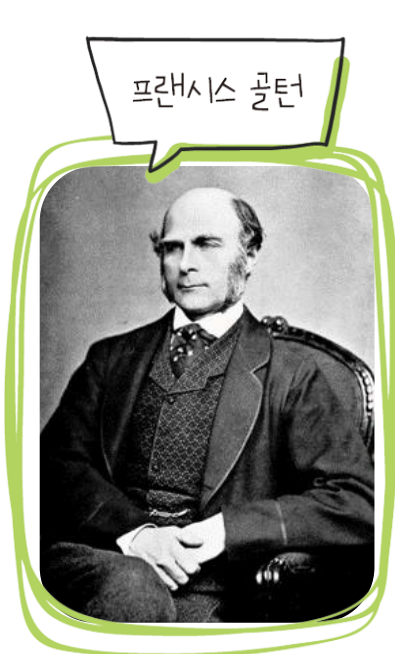
테스트 데이터로 평가하기

테스트 데이터로 평가하기

# 선형 회귀의 이해

## 회귀란?

- 회귀(regression) : '평균으로 돌아간다'라는 의미
- 영국의 인류학자 프랜시스 골턴(1822~1911)이 928명의 성인 자녀 키와 부모 키의 상관관계 연구를 통해 밝혀낸 자연 현상
- 골턴은 부모의 키와 자녀의 키를 산점도로 표현
- 추세선을 통해 부모와 자녀의 키 관계가 선형적인 관계임을 증명
- 부모의 키에 다른 자녀의 키는 평균으로 돌아가려는 경향(회귀)이 있다는 것을 알아냄



# 선형 회귀의 이해

## 회귀란?

- **회귀**(regression) :  $x$ 값에 대한 연속적인  $y$ 값을 예측하는 것
- **종속변수** : 자녀의 키( $y$ )를 예측하고자 하는  $y$ 값
- **독립변수** : 부모의 키( $x$ ), 종속변수를 설명하는 변수
- **분류**(classification) : 부모 대비 자녀의 키( $y$ )의 상태가 '크다'와 '작다'처럼 이산적인 경우

부모의 키( $x$ )	자녀의 키( $y$ )
165	168
172	175
180	177
190	180

▲  $y$ 값이 연속적이므로 회귀

부모의 키( $x$ )	부모 대비 자녀의 키( $y$ )
165	크다
172	크다
180	작다
190	작다

▲  $y$ 값이 이산적이므로 분류

# 선형 회귀의 이해

## 선형 회귀의 종류

- 데이터를 가장 잘 설명하는 직선
- 종속변수  $y$ 와 하나 이상의 독립변수  $x$ 와의 선형 상관관계를 모델링하는 기법

### 단순 선형 회귀 (Simple Linear Regression)

$$y = wx + b$$

$w$ : 가중치(weight)  
 $b$ : 편향(bias)

- 독립변수  $x$ 가 1개
- 직선의 형태

### 다중 선형 회귀 (Multiple Linear Regression)

$$y = w_1x_1 + w_2x_2 + \dots + w_nx_n + b$$

- 독립변수  $x$ 가 2개 이상

#### 단순 선형 회귀

$x$  (평균 온도)

25  
⋮

$y$  (와인 가격)

9.5  
⋮

#### 다중 선형 회귀

$x_1$  (겨울 강수량)

13  
⋮

$x_2$  (수확철 강수량)

35  
⋮

$x_3$  (평균 온도)

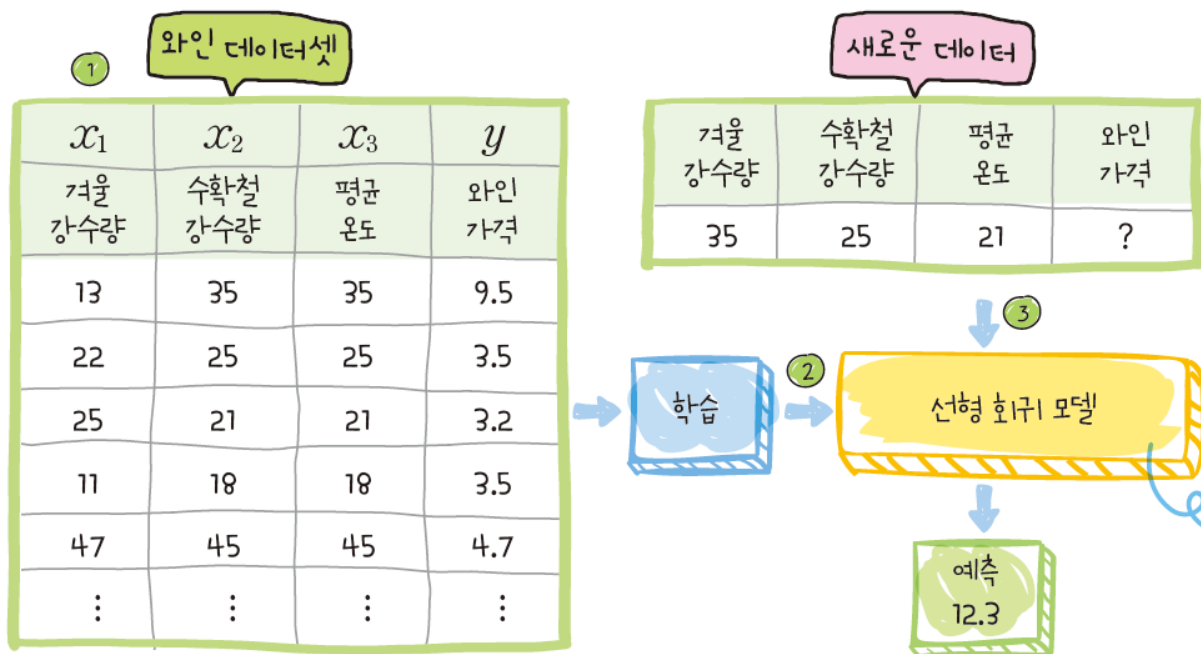
25  
⋮

$y$  (와인 가격)

9.5  
⋮

## 선형 회귀 알고리즘

와인 데이터셋에서 독립변수인 겨울 강수량( $x_1$ ), 수확철 강수량( $x_2$ ), 평균 온도( $x_3$ )일 때, 종속변수인 와인 가격( $y$ )을 선형 회귀 알고리즘으로 예측해 보자.



- ① 와인 데이터셋은 3개의 독립변수와 1개의 종속변수로 이루어져 있다.
- ② 데이터를 학습시켜, 와인 가격을 예측하는 선형 회귀 모델을 생성한다.
- ③ 새로운 데이터로 와인 가격을 예측한다.

$$y = w_1x_1 + w_2x_2 + w_3x_3 + b$$



# 문제 정의하기

## 문제 상황 이해하기

### 광고 플랫폼에 따른 판매량은 얼마나 될까요?

전통적인 광고 플랫폼인 TV, 라디오, 신문은 판매량에 어떤 영향을 주는지 알아보고,  
플랫폼별 광고비에 따른 판매량을 예측해 보겠습니다.



# 문제 정의하기

문제 해결에  
필요한 정보  
살펴보기

문제 해결 과정에서 필요한 정보를 미리 살펴봅시다.

**1** 이 활동에 필요한 데이터셋은 무엇이고, 이 데이터셋은 어디에서 수집할 수 있나요?

광고 데이터셋으로 캐글에서 다운로드할 수 있습니다.

**2** 모델 학습에 사용되는 알고리즘은 무엇인가요?

선형 회귀 알고리즘을 사용합니다. 선형 회귀는 회귀에서 가장 일반적인 알고리즘입니다. 단순 선형 회귀와 다중 선형 회귀가 있으며, 선형 회귀는 직선을 그려 연속적인 값을 예측하는 방법입니다.

# 문제 정의하기

문제 해결에  
필요한 정보  
살펴보기

## 3 모델 학습을 위해 어떤 처리를 해야 할까요?

광고 데이터셋에서 속성 간의 차이와 어떤 속성이 판매량에 영향을 미치는지 탐색하고, 데이터를 학습시킵니다.

## 4 선형 회귀 모델 성능을 나타내는 평가 지표는 무엇일까요?

테스트 데이터의 예측값과 실젯값을 비교하여 얼마나 차이가 나는지를 평가하는 지표로 평균 제곱 오차와 결정계수를 사용합니다.

# 데이터 불러오기

## 데이터셋 소개하기

### 광고 데이터셋 살펴보기

- 총 200개의 데이터
- 4개의 속성 : 텔레비전(TV) 광고비, 라디오(Radio) 광고비, 신문(Newspaper) 광고비, 판매량 (Sales)
- TV, Radio, Newspaper의 기본 단위 : 1,000 달러(\$)
- Sales 기본 단위 : 1,000개

TV의 230.1은 230,100\$이고,  
Sales의 22.1은 22,100개를  
의미합니다.

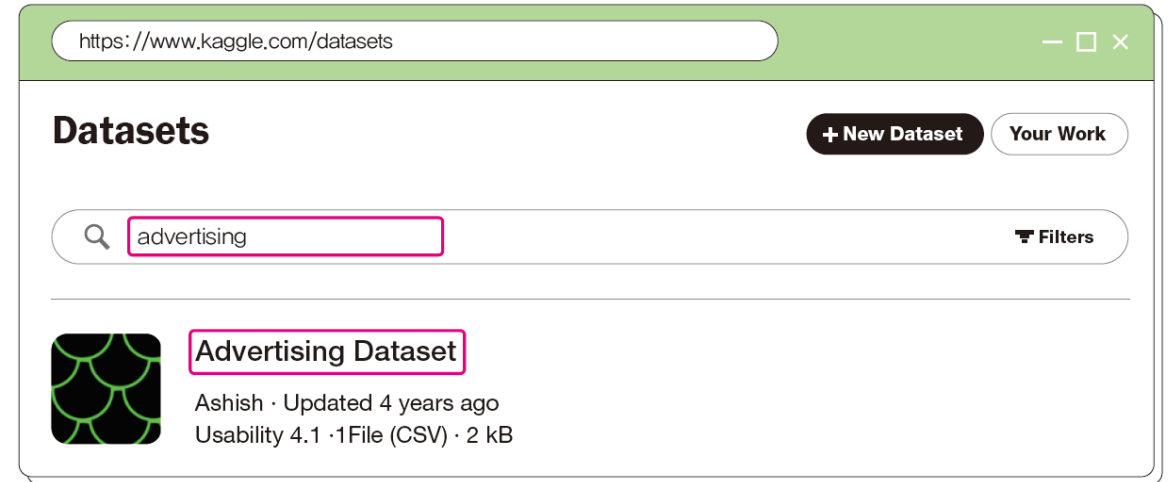
	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

# 데이터 불러오기

## 광고 데이터셋 다운로드하기

캐글에서 검색어 'advertising'을  
입력하여 Ashish가 등록한  
'Advertising Dataset' 선택하기

하단의 데이터셋 미리보기 부분  
에서 다운로드 아이콘(📄)을  
클릭하여 'advertising.csv'파일  
다운로드하기



# 데이터 불러오기

## 데이터셋 불러오기

## 파일 업로드하기


컴퓨터에 저장된 파일('advertising.csv')을 코랩으로 불러오는 방법을 사용하겠습니다.

- 1 google.colab 라이브러리에서 파일을 처리하는 데 사용하는 files를 불러온 후, 파일 선택 창에서 파일 업로드하기

```
1 from google.colab import files
2 filename = list(files.upload().keys())[0]
```

 파일 선택 선택된 파일 없음 Cancel upload

- 2 파일 선택 버튼을 클릭하여 파일을 다운로드받은 경로에서 'advertising.csv' 파일을 선택하면 자동으로 코랩의 저장 폴더로 업로드됨

 파일 선택 advertising.csv

- advertising.csv(text/csv)-4062 bytes, last modified: 2023. 6. 24.-100% done

Saving advertising.csv to advertising.csv

# 데이터 불러오기

## 파일 읽어 들이기

- 판다스 라이브러리를 사용하여 read\_csv( )를 통해 파일을 데이터프레임으로 변환
- sample( ) 메소드를 사용하여 3개의 데이터를 임의로 추출하여 출력

데이터프레임 객체 = 판다스 객체.read\_csv('파일명.csv')

# '파일명' 대신 파일을 저장할 변수로 설정 가능

데이터프레임 객체.sample()

# 임의의 데이터 샘플을 보여 주는 메소드

# 데이터 불러오기

## 파일 읽어 들이기



```
1 import pandas as pd
2 advertising = pd.read_csv(filename) # CSV 파일 읽어오기
3 advertising.sample(3) # 랜덤하게 데이터 3개 읽어오기
```



	TV	Radio	Newspaper	Sales
59	210.7	29.5	9.3	18.4
141	93.7	35.4	75.6	19.2
66	31.5	24.6	2.2	11.0

sample() 함수를 이용하면 랜덤한 샘플 데이터를 읽어올 수 있고, 매번 읽어올 때마다 데이터가 달라집니다.



# 데이터 탐색하기

## 데이터셋 살펴보기

어떤 속성이 포함되어 있는지, 각 속성의 값은 어떤 유형의 데이터인지, 결측치나 이상치가 있는지 등을 파악하고, 속성 간의 상관관계와 속성값의 분포를 시각화하는 작업을 합니다.

- 판다스 라이브러리의 `info()` 메소드를 사용하여 광고 데이터의 기초 정보 확인하기

### 데이터프레임 객체.info()

# 데이터 개수, 속성 개수, 속성명, 결측치, 속성의 데이터 유형 등 확인

# 데이터 탐색하기

## 데이터 기초 정보 확인하기

속성명	의미
TV	TV 광고비
Radio	라디오 광고비
Newspaper	신문 광고비
Sales	판매량



1

advertising.info()



```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 200 entries, 0 to 199
```

```
Data columns (total 4 columns):
```

#	Column	Non-Null Count	Dtype
0	TV	200 non-null	float64
1	Radio	200 non-null	float64
2	Newspaper	200 non-null	float64
3	Sales	200 non-null	float64

```
dtypes: float64(4)
```

```
memory usage: 6.4 KB
```



### 해석

이 데이터셋은 총 200개의 데이터로 구성되어 있으며, 속성은 4개입니다. 각 속성이 200개의 값을 가진 것으로 보아 결측치가 없음을 확인할 수 있습니다. 그리고 속성별 데이터 유형은 모두 실수형 (float64)으로 구성되어 있습니다.

# 데이터 탐색하기

## 데이터 통계치 살펴보기

- 광고 데이터 속성들의 통계치를 파악하기 위해 describe() 메소드 사용
- 수치형 데이터에 대해서만 통계치를 출력하며 결측치는 제외됨

### 데이터프레임 객체.describe().T

# 속성별 개수, 평균값, 표준편차, 최솟값, 사분위수, 최댓값을 한눈에 살펴봄.

# 데이터프레임 객체 T는 행과 열을 바꾸는 데 사용됨.

# 데이터 탐색하기

## 데이터 통계치 살펴보기

- 광고 데이터의 4개 속성에 대한 통계치 출력하기

```
1 advertising.describe().T # 데이터프레임의 행과 열 교환하기
```



	count	mean	std	min	25%	50%	75%	max
TV	200.0	147.0425	85.854236	0.7	74.375	149.75	218.825	296.4
Radio	200.0	23.2640	14.846809	0.0	9.975	22.90	36.525	49.6
Newspaper	200.0	30.5540	21.778621	0.3	12.750	25.75	45.100	114.0
Sales	200.0	15.1305	5.283892	1.6	11.000	16.00	19.050	27.0

TV 광고비는 700\$부터 296,400\$의 범위이고, Radio 광고비는 0\$부터 49,600\$입니다. 0\$인 것은 Radio 광고를 하지 않은 것으로 예상됩니다. Newspaper 광고비는 300\$부터 114,000\$까지입니다.

# 데이터 탐색하기

## 상관관계 분석하기

판매량 예측을 위하여 판매량(Sales)과 관련 있는 속성은 어떤 것들이 있는지 상관관계 분석을 통해 살펴보겠습니다.

## 속성별 상관관계 알아보기

- 판다스 라이브러리의 `corr()` 메소드를 사용하여 각 속성 간의 상관관계 파악하기

## 데이터프레임 객체.`corr()`

# 상관관계(correlation) 분석

# 데이터 탐색하기

## 속성별 상관관계 알아보기

```
1 corrMatrix = advertising.corr() # 상관관계 분석
2 corrMatrix
```



	TV	Radio	Newspaper	Sales
TV	1.000000	0.054809	0.056648	0.901208
Radio	0.054809	1.000000	0.354104	0.349631
Newspaper	0.056648	0.354104	1.000000	0.157960
Sales	0.901208	0.349631	0.157960	1.000000

대각선을 기준으로 위, 아래 비율이 같습니다. 어느 쪽을 기준으로 확인하든 상관없으며 우리는 가로를 기준으로 확인해 보면, Sales를 기준으로 상관관계가 가장 큰 것은 TV입니다. 상관관계수가 1에 가까울수록 높은 것이므로 Sales와 TV와의 관계는 약 0.9로 1에 상당히 가까운 것을 볼 수 있습니다.

# 데이터 탐색하기

## 히트맵으로 표현하기

- 각 속성 간의 관계를 색으로 나타내어 그 관계를 쉽게 파악할 수 있도록 히트맵 사용
- 히트맵은 시본(seaborn) 라이브러리를 사용하여 쉽게 구현가능

**시본 객체.heatmap(데이터프레임, annot = True/False,  
cmap = '색상명')**

# annot: 히트맵에 값(여기서는 상관계수) 포함 여부, 기본값은 False

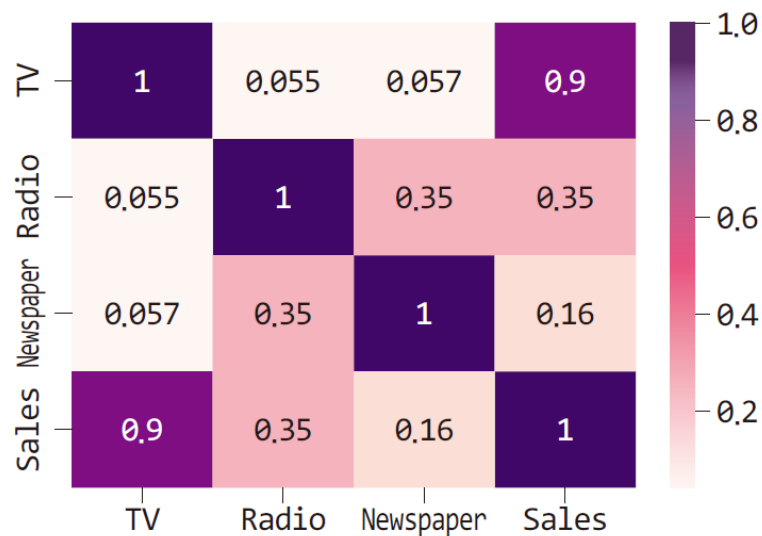
# cmap: matplotlib colormap 이름 또는 객체

# 데이터 탐색하기

## 히트맵으로 표현하기

```
1 import seaborn as sns
2 import matplotlib.pyplot as plt
3 sns.heatmap(corrMatrix, annot = True, cmap = 'RdPu')
4 plt.show()
```

'RdPu'는 "Red-Purple"의  
줄임말입니다.



값	상관관계 값의 의미
-1.0 ~ -0.7	강한 음의 상관관계
-0.7 ~ -0.3	음의 상관관계
-0.3 ~ -0.1	약한 음의 상관관계
-0.1 ~ 0.1	상관관계 거의 없음
0.1 ~ 0.3	약한 양의 상관관계
0.3 ~ 0.7	양의 상관관계
0.7 ~ 1.0	강한 양의 상관관계



# 데이터 탐색하기

## cmap의 다양한 색상 종류

히트맵의 옵션 중 cmap은 맵플롯립의 색상 종류를 설정합니다. 다음과 같이 코드를 입력하면 cmap의 종류를 출력해 볼 수 있습니다.

```
1 import matplotlib.pyplot as plt
2 from matplotlib import cm
3 cmaps = plt.colormaps()
4 print(len(cmaps))
5 print(cmaps)
```



178

['magma', 'inferno', 'plasma', ..., 'flare', 'flare\_r', 'crest', 'crest\_r']

## 속성 기준으로 데이터 정렬하기

- 히트맵을 기준으로 상관관계가 높은 속성을 내림차순으로 정렬하여 시각화
- 데이터를 정렬할 때는 `sort_values()` 메소드 사용

### 데이터프레임 객체.`sort_values`

(`by` = '정렬 기준이 되는 속성명', `ascending` = `True/False`)

# 정렬 기준이 되는 속성명을 기준으로 정렬

# 기본은 오름차순 정렬(`ascending` = `True`)이며, 내림차순 정렬은 `ascending` = `False`로 설정

# 데이터 탐색하기

속성 기준으로  
데이터 정렬하기

```
1 corr_sort = corrMatrix[['Sales']].sort_values(by = 'Sales', ascending = False)
2 corr_sort
```



	Sales
Sales	1.000000
TV	0.901208
Radio	0.349631
Newspaper	0.157960



해석

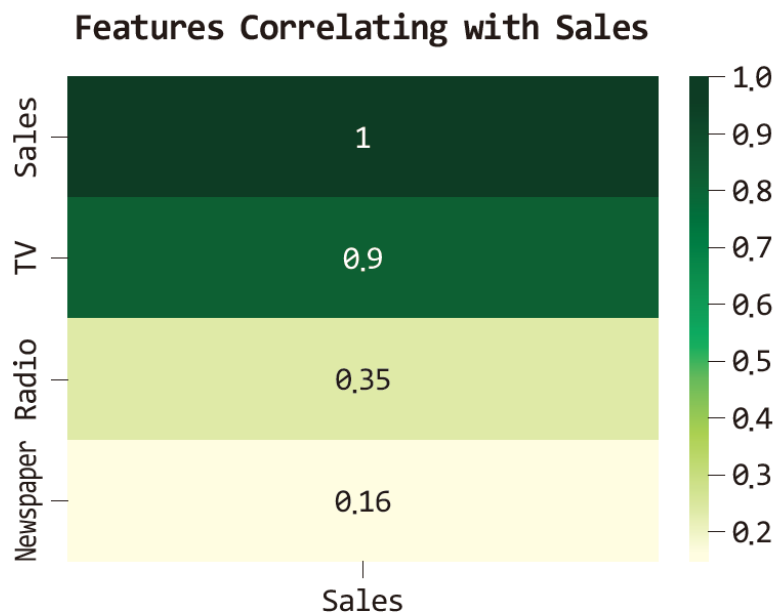
실행 결과를 통해 판매량(Sales) 속성을 기준으로 내림차순으로 정렬된 속성들을 확인할 수 있습니다.

# 데이터 탐색하기

## 속성 기준으로 데이터 정렬하기

- 내림차순으로 정렬된 속성 간의 관계를 히트맵을 사용하여 색으로 출력하기

```
1 heatmap = sns.heatmap(corr_sort, annot = True, cmap = 'YlGn')  
2 heatmap.set_title('Features Correlating with Sales') # 히트맵의 제목  
3 plt.show()
```



### 해석

실행 결과를 통해 판매량 (Sales)과 높은 상관관계를 갖는 속성을 시각적으로 확인할 수 있습니다.

# 데이터 탐색하기

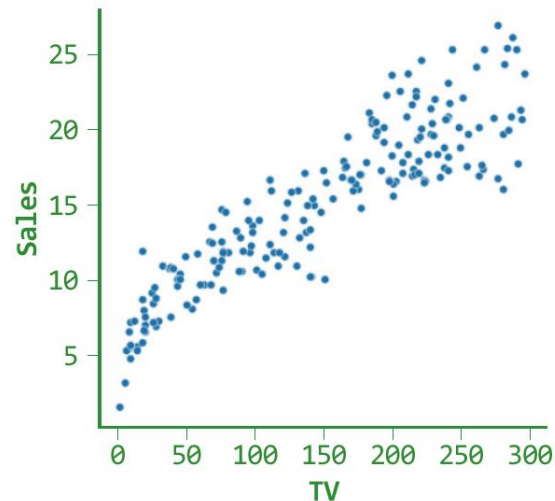
## 속성별 상관관계 알아보기

- 시본의 pairplot() 메소드를 사용하여 여러 개의 그래프로 속성 관계 살펴보기

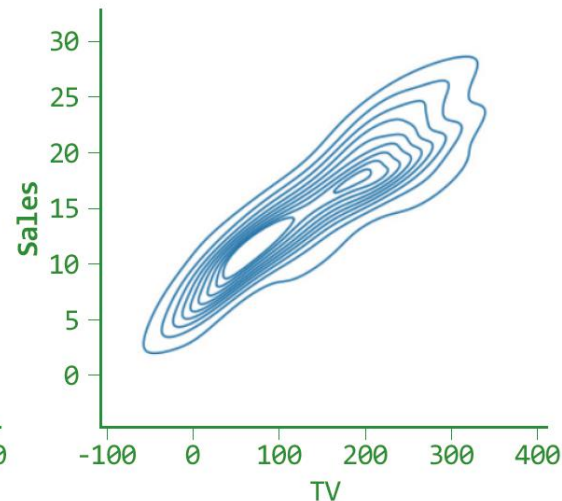
시본 객체.pairplot(data = 데이터프레임 객체, x\_vars = ['속성명'],  
y\_vars = ['속성명'], height = 크기, kind = '그래프 종류')

[그래프 종류]

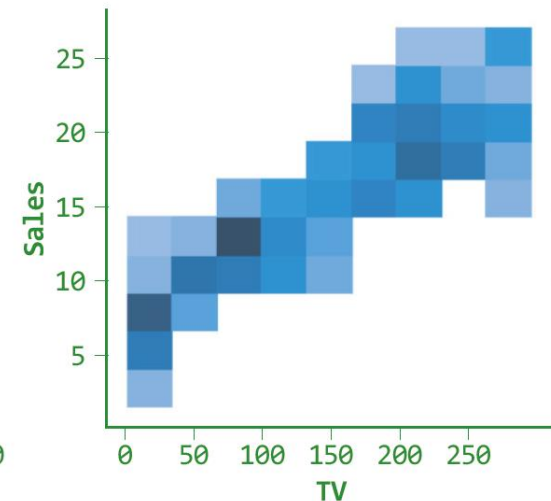
\*kind = 'scatter'(산점도)



\*kind = 'kde'(커널 밀도 추정)



\*kind = 'hist'(히스토그램)



# 데이터 탐색하기

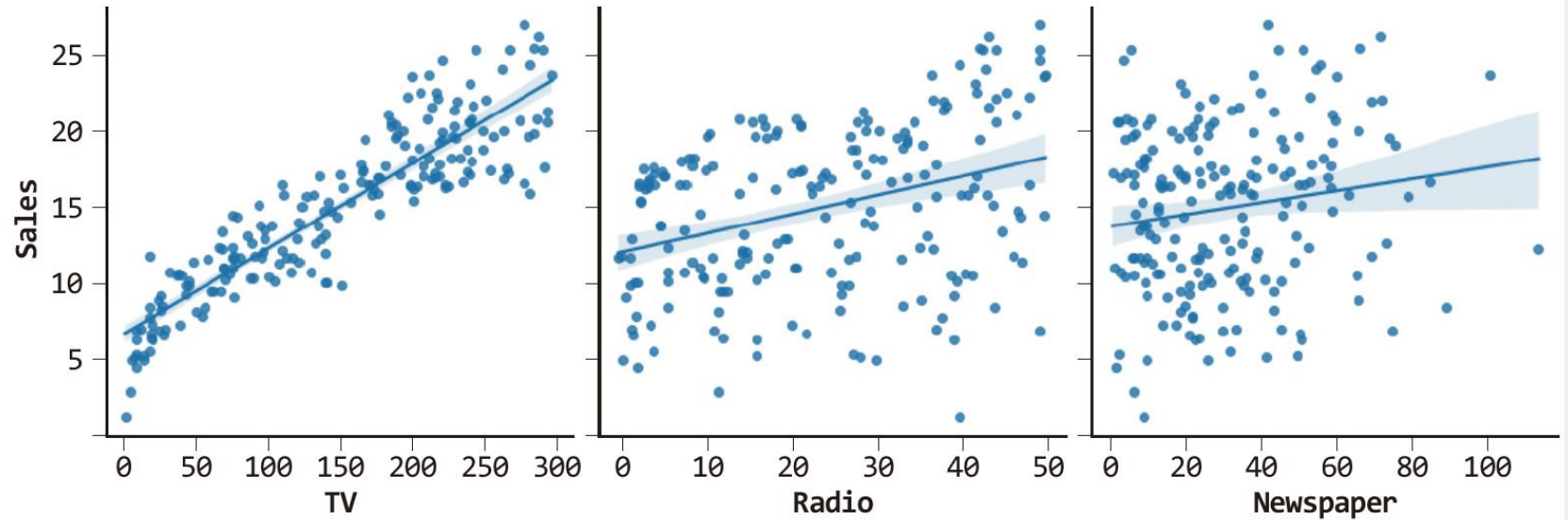
## 속성별 상관관계 알아보기

- x 축 : 'TV', 'Radio', 'Newspaper'
- y 축 : 'Sales'



1

```
sns.pairplot(data = advertising, x_vars = ['TV', 'Radio', 'Newspaper'],  
             y_vars = ['Sales'], height = 5, kind = 'reg')
```



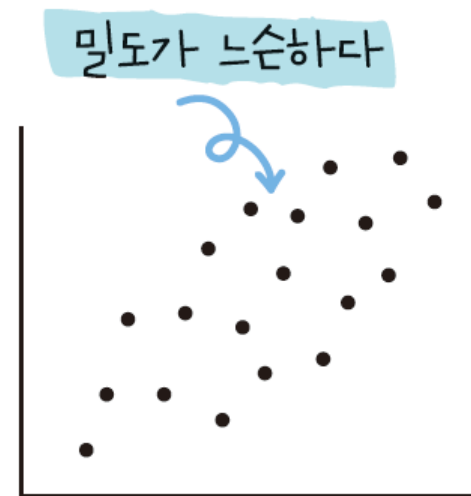
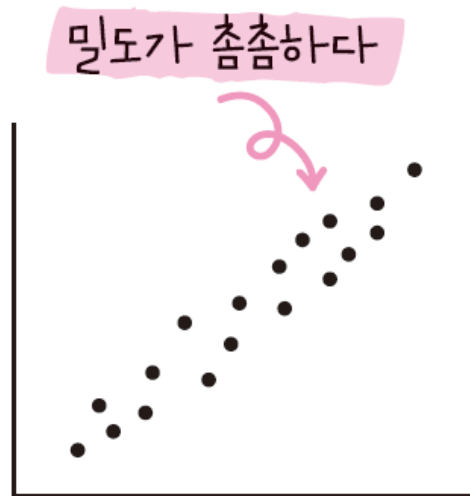
kind = 'reg'는  
kind = 'scatter'의  
경우에 추세선을 포함한  
경우입니다.

# 데이터 탐색하기

## 속성별 상관관계 알아보기

상관관계가 높은 TV와 Sales는 직선 부분에 데이터가 많이 몰려 있고 밀도가 촘촘합니다.  
이 '밀도를 표현한 숫자'를 '상관계수'라고 부릅니다.

상관관계는 -1부터 1사이의 값을 가지며 -1에 가까울수록 음(-)의 상관관계가 강하며, +1에 가까울수록 양(+)의 상관관계가 강하다는 의미입니다.



# 데이터 탐색하기

## 단순 선형 회귀

한 개의 독립변수  $x$  와 한 개의 종속변수  $y$  의 선형 상관관계

## 다중 선형 회귀

둘 이상의 독립변수  $x$  를 가지는 경우의 선형 상관관계

단순 선형 회귀와 다중 선형 회귀로 나누어 모델을 학습하고 평가해 봅시다.



# 모델 학습하기

## 단순 선형 회귀 데이터 나누기

- 단순 선형 회귀 :  $y = wx + b$  형태
- 독립변수 : TV 광고비(TV)
- 종속변수 : 판매량(Sales)

**독립변수 객체** = 데이터프레임 객체[['속성명1', '속성명2', ...]]

**종속변수 객체** = 데이터프레임 객체['속성명']

# 모델 학습하기

독립변수와  
종속변수로  
구분하기

```
1 X_data1 = advertising[['TV']] # 독립변수
2 y_data1 = advertising['Sales'] # 종속변수
3 display(X_data1)
4 display(y_data1)
```



	TV
0	230.1
1	44.5
2	17.2
3	151.5
..	...
198	283.6
199	232.1

[200 rows x 1 columns]

0	22.1
1	10.4
2	12.0
3	16.5
...	...
198	25.5
199	18.4

Name: Sales, Length: 200, dtype: float64

독립변수 X\_data1과  
종속변수 y\_data1을  
설정합니다.

# 모델 학습하기

## 표준화 적용하기

- 속성값의 차이가 있으므로 속성값의 범위를 일정하게 맞춰 주는 방법으로 표준화 적용

```
1 from sklearn.preprocessing import StandardScaler
2 scaler = StandardScaler()
3 X_scaled1 = scaler.fit_transform(X_data1) # 표준화 데이터
4 X_scaled1
```

```
array([[ 0.96985227],
       [-1.19737623],
        ...,
        [ 1.59456522],
        [ 0.99320602]])
```

표준화를 이용하여 평균이 0이고, 표준편차가 1인 정규분포로 만들었습니다.

# 모델 학습하기

## 훈련 데이터와 테스트 데이터 나누기

- 인공지능 모델을 학습시키기 위해서 광고 데이터셋을 훈련 데이터와 테스트 데이터를 7:3 비율로 나누기
- random\_state 옵션을 사용하여 훈련 데이터와 테스트 데이터를 동일한 패턴으로 추출하기

```
1 from sklearn.model_selection import train_test_split
2 X_train1, X_test1, y_train1, y_test1 = train_test_split(X_scaled1, y_data1,
3                                                         test_size = 0.3, random_state = 10)
4 print(X_train1.shape, X_test1.shape, y_train1.shape, y_test1.shape)
```

(140, 1) (60, 1) (140, ) (60, )

독립변수 X, 종속변수 Y로부터  
훈련 데이터 : 테스트 데이터 =  
7 : 3으로 분할하였으므로 훈련용  
독립변수 데이터의 개수는 140,  
테스트용 독립변수는 60개입니다.  
데이터를 추출하는 패턴(random\_  
state)은 10으로 설정하였으며,  
이 패턴에 따라 **훈련 데이터와  
테스트 데이터를 분할할 때마다  
동일하게 분할됩니다.**

데이터	독립변수(70%)	종속변수(30%)
훈련 데이터	X_train1 (140, 1)	y_train1 (140, )
테스트 데이터	X_test1 (60,1)	y_test1 (60, )

# 모델 학습하기

## 모델 생성 및 학습하기

- 사이킷런 라이브러리에서 제공하는 `LinearRegression()`을 통해 선형 회귀 모델 생성
- 훈련용 독립변수(`X_train1`)와 종속변수(`y_train1`)를 사용하여 `fit()` 메소드로 학습

```
1 from sklearn.linear_model import LinearRegression
2 lr_model1 = LinearRegression() # 선형 회귀 모델 생성
3 lr_model1.fit(X_train1, y_train1) # 선형 회귀 모델 학습
```

```
LinearRegression()
```

### 해석

`LinearRegression` 모듈을 불러와 모델 생성, 모델 학습의 순서로 코드를 구현합니다.  
지도학습이므로 독립변수(`X_train1`)와 종속변수(`y_train1`)를 같이 학습시킵니다.

# 모델 테스트 및 평가하기

단순 선형 회귀

## 모델 평가하기

회귀 모델 평가 지표로 평균 제곱 오차와 결정 계수를 사용해 봅니다.

### 평균 제곱 오차 (MSE)

- Mean Squared Error
- 예측값이 실제값과 얼마나 차이가 있는지를 평가
- 0에 가까울수록 좋은 모델

### 결정 계수 ( $R^2$ )

- 0부터 1 사이의 값을 가짐
- 1에 가까울수록 모델의 적합도가 좋음을 의미

# 모델 테스트 및 평가하기

단순 선형 회귀

## 실제값과 예측값 시각화하기

- 60개의 테스트 데이터로 성능을 평가하고 예측해 보기
- 실제값과 예측값을 그래프로 출력하기

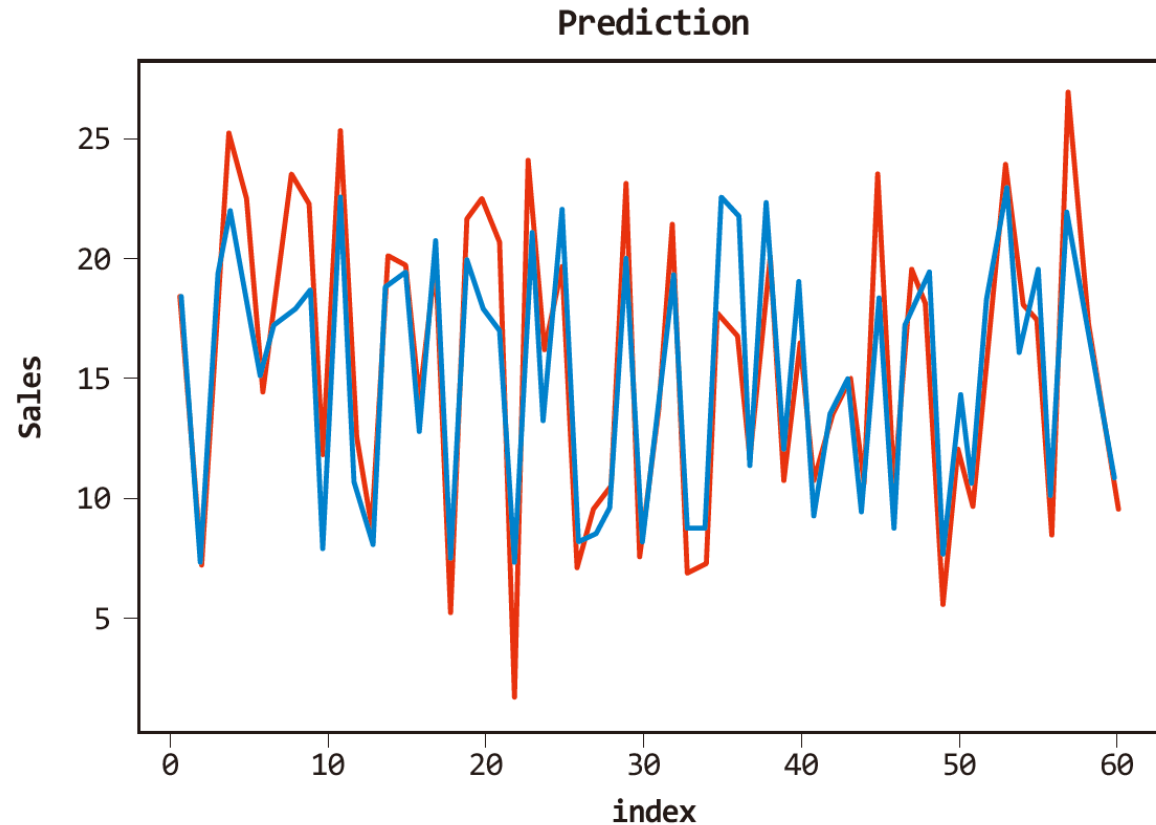
```
1 y_pred1 = lr_model1.predict(X_test1)
2 c = [i for i in range(1, 61, 1)]
3 plt.plot(c, y_test1, color = 'r') # 실제값
4 plt.plot(c, y_pred1, color = 'b') # 예측값
5 plt.xlabel('index') # X축 이름
6 plt.ylabel('Sales') # Y축 이름
7 plt.title('Prediction') # 그래프 제목
8 plt.show()
```

모델에 predict( ) 메소드를 사용하여 테스트용 독립변수 (X\_test1) 값을 넣어 값을 예측합니다.

# 모델 테스트 및 평가하기

단순 선형 회귀

실제값과 예측값  
시각화하기



빨간선: 실제값  
파란선: 예측값

## 해석

c는 1번부터 60번까지 1씩 증가하는 리스트입니다. 빨간선 그래프는 `y_test1`로 실제값이고, 파란선 그래프는 `y_pred1`로 선형 회귀 모델이 예측한 값입니다. 실제값과 예측값 사이에 어느 정도 차이(오차)가 있는지 직관적으로 확인이 가능합니다.



# 모델 테스트 및 평가하기

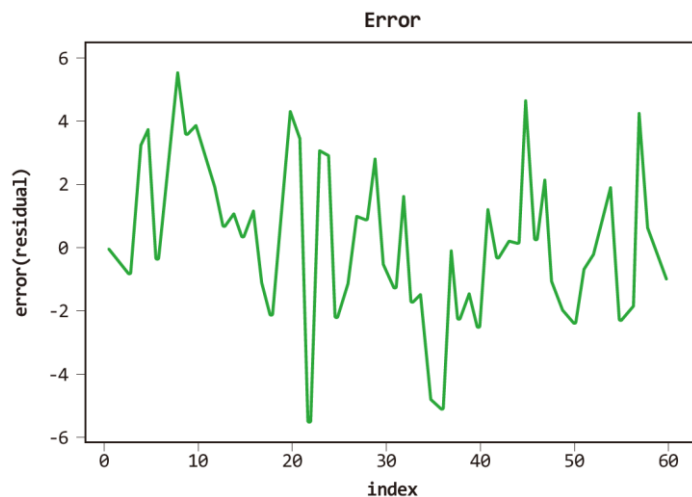
단순 선형 회귀

## 오차 시각화하기

- TV 광고비(TV)에 대한 판매량(Sales)의 실젯값과 예측값의 차이인 오차(error)를 그래프로 시각화



```
1 error = y_test1 - y_pred1 # 실젯값 - 예측값
2 plt.plot(c, error, color = 'g')
3 plt.xlabel('index')
4 plt.ylabel('error(residual)')
5 plt.title('Error')
6 plt.show()
```



이 그래프는 TV 광고비에 대한 판매량의 실젯값과 예측값 사이의 오차를 계산하여 index 별로 시각화한 그래프입니다. 오차가 0에 가까운 값도 있지만 +6과 -6에 가까운 차이를 보이는 값도 있습니다.

# 모델 테스트 및 평가하기

단순 선형 회귀

## 성능 평가하기

- 모델의 예측값( $y_{pred1}$ )과 실제값( $y_{test1}$ )을 이용하여 성능 평가 지표인 MSE와  $R^2$  구하기

```
1 from sklearn.metrics import mean_squared_error
2 from sklearn.metrics import r2_score
3 print(f 'MSE: {mean_squared_error(y_pred1, y_test1):.2f}') # 평균 제곱 오차(MSE)
4 print(f 'r2_score: {r2_score(y_pred1, y_test1):.2f}') # 결정계수( $R^2$ )
```



MSE: 6.46

r2\_score: 0.75



### 해석

MSE는 0에 가까울수록 성능이 좋은 것이고,  $r2\_score$ 는 1에 가까울수록 모형의 예측 능력이 좋다고 판단합니다.

# 모델 테스트 및 평가하기

단순 선형 회귀

판매량  
예측하기

- 데이터를 설명하는 선형 회귀선을 그리고, 회귀식을 구하여 TV 광고비와 판매량 사이의 관계 표현하기

```
파이플롯 객체.scatter(x, y, color = '점 색', label = 'scatter plot')
```

```
# x, y 모두 수치형 데이터인 산점도
```

```
파이플롯 객체.plot(x, y)
```

```
# 기본 그래프 그리기(직선)
```

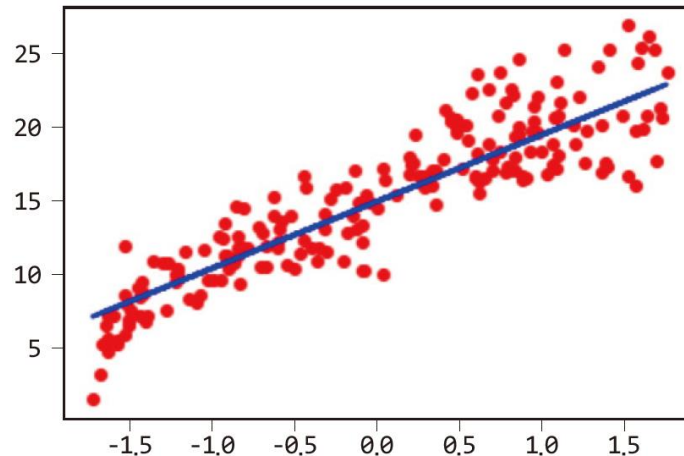
# 모델 테스트 및 평가하기

단순 선형 회귀

## 단순 선형 회귀선 그리기

- 산점도와 선형 회귀선 그리기
- X\_scaled1: X\_data1(TV 광고비)을 표준화한 데이터
- y\_data1: 판매량(Sales) 데이터

```
1 plt.scatter(X_scaled1, y_data1, color = 'red', label = 'scatter plot')
2 plt.plot(X_test1, y_pred1, color = 'blue', linewidth = 2,
3          label = 'Regression Line')
4 plt.show()
```



1. 광고 플랫폼에 따른 판매량을 예측해 볼까?

# 모델 테스트 및 평가하기

단순 선형 회귀

## 선형 회귀식 구하기

- TV 광고비와 판매량의 관계를 설명하는 단순 선형 회귀식인  $y = wx + b$ 의 기울기 계수( $w$ )와 절편( $b$ )을 구하여 선형 회귀식 완성하기

```
1  # Slope Coefficients(기울기 계수)
2  w1 = lr_model1.coef_ # 기울기
3  print(f 'slopes of TV: {w1[0]:.2f}')
4
5  # Intercept(절편)
6  b1 = lr_model1.intercept_ # y 절편
7  print('Intercept is:', b1.round(2))
```



```
slopes of TV: 4.57
Intercept is: 15.0
```

선형 회귀식은  $y = 4.57x + 15$  입니다.  
이 회귀식에 새로운  $x$ 를 입력하면  $y$ 를  
예측할 수 있습니다.

# 모델 학습하기

다중 선형 회귀

다중 선형 회귀  
데이터 나누기

독립변수와  
종속변수로  
구분하기

- 다중 선형 회귀 :  $y = w_1x_1 + w_2x_2 + w_3x_3 + b$  형태
- 독립변수 : TV 광고 비(TV), 라디오 광고비(Radio), 신문 광고비(Newspaper) 3개 속성
- 종속변수 : 판매량(Sales) 속성

```
1 X_data2 = advertising.drop(['Sales'], axis = 1) # 독립변수
2 y_data2 = advertising['Sales'] # 종속변수
```

X\_data2 객체에는 advertising 데이터프레임에서 Sales 열 데이터를 삭제한 후, 나머지 TV, Radio, Newspaper 열 데이터를 저장합니다. y\_data2 객체에는 Sales 열 데이터만 저장합니다.

# 모델 학습하기

다중 선형 회귀

## 데이터 표준화하기

- 속성값의 차이가 있으므로 범위를 일정하게 맞춰 주는 표준화 적용하기

```
1 from sklearn.preprocessing import StandardScaler
2 scaler = StandardScaler()
3 X_scaled2 = scaler.fit_transform(X_data2)
```

## 훈련 데이터와 테스트 데이터 나누기

- 훈련 데이터와 테스트 데이터를 7:3으로 나누기
- random\_state 옵션을 사용하여 훈련 데이터와 테스트 데이터를 동일한 패턴으로 추출하기

```
1 from sklearn.model_selection import train_test_split
2 X_train2, X_test2, y_train2, y_test2 = train_test_split(X_scaled2, y_data2,
3                                                         test_size = 0.3, random_state = 10)
4 print(X_train2.shape, X_test2.shape, y_train2.shape, y_test2.shape)
```

```
(140, 3) (60, 3) (140, ) (60, )
```

# 모델 학습하기

다중 선형 회귀

## 모델 생성 및 학습하기

- 모델을 생성하고 학습시키기

```
1 from sklearn.linear_model import LinearRegression
2 lr_model2 = LinearRegression()
3 lr_model2.fit(X_train2, y_train2) # 훈련 데이터
```

```
LinearRegression()
```



# 모델 테스트 및 평가하기

다중 선형 회귀

## 모델 평가하기

- 실제값과 예측값을 구해 데이터프레임 형태로 만들기

```
1 mlr = pd.DataFrame({'Actual_value':y_test2 ,  
2                       'Model prediction':lr_model2.predict(X_test2)})  
3 mlr.head()
```



	실제값 Actual_value	예측값 Model prediction
59	18.4	19.127479
5	7.2	10.658525
20	18.0	19.356496
198	25.5	24.315643
52	22.6	20.751037

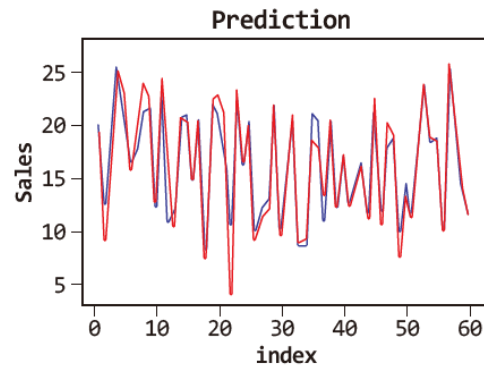
# 모델 테스트 및 평가하기

다중 선형 회귀

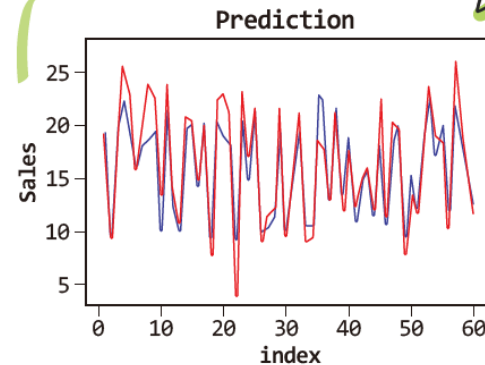
## 실제값과 예측값 시각화하기

- 단순 선형 회귀와 같이 다중 선형 회귀에서도 실제값과 예측값의 오차를 그래프로 표현하는 코드 동일

```
1 y_pred2 = lr_model2.predict(X_test2)
2 c = [i for i in range(1, 61, 1)]
3 plt.plot(c, y_test2, color='r')
4 plt.plot(c, y_pred2, color='b')
5 plt.xlabel('index')
6 plt.ylabel('Sales')
7 plt.title('Prediction')
8 plt.show()
```



다중 선형인 경우



단순 선형인 경우

220쪽  
그래프입니다.

## 해석

단순 선형 회귀에 비해  
오차가 다소 줄어든 것을  
확인할 수 있습니다.

# 모델 테스트 및 평가하기

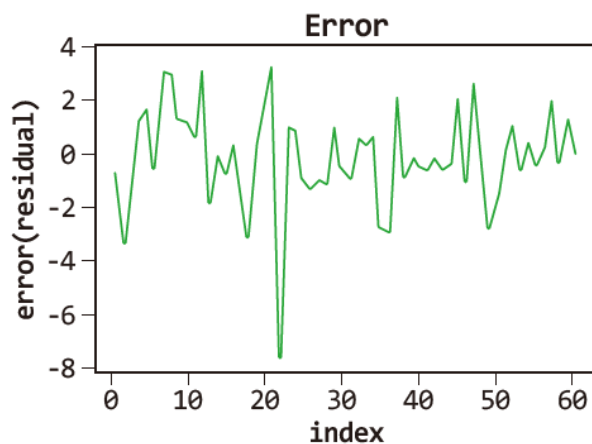
다중 선형 회귀

## 오차 시각화하기

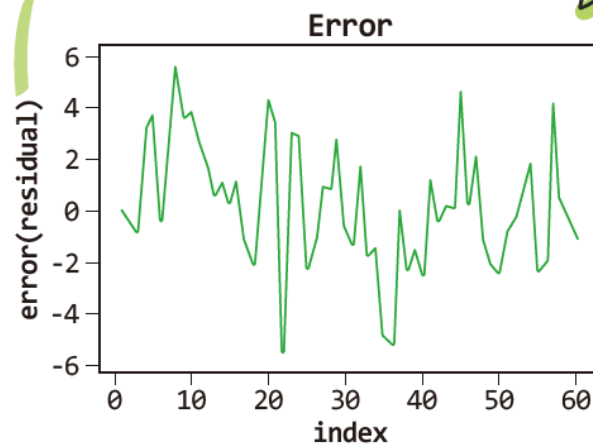
- 실제값과 예측값의 차이인 오차(error)를 구해 그래프로 시각화하기

이 그래프는 실제값과 예측값 사이의 오차를 계산하여 index별로 시각화한 그래프입니다. 오차가 0에 가까운 값도 있지만 +4와 -8에 가까운 차이도 보입니다.

```
1 error2 = y_test2 - y_pred2
2 plt.plot(c, error2, color = 'g')
3 plt.xlabel('index')
4 plt.ylabel('error(residual)')
5 plt.title('Error')
6 plt.show()
```



다중 선형인 경우



단순 선형인 경우

221쪽  
그래프입니다.

# 모델 테스트 및 평가하기

다중 선형 회귀

## 성능 평가하기

- 오차 수치로 표현하기
- MSE : 0에 가까울수록 성능이 좋음
- r2\_score : 1에 가까울수록 성능이 좋음

```
1 print(f 'MSE: {mean_squared_error(y_pred2, y_test2):.2f}')
```

```
2 print(f 'r2_score: {r2_score(y_pred2, y_test2):.2f}')
```



MSE: 3.66

r2\_score: 0.87



## 해석

단순 선형 회귀의 MSE는 6.46였는데 다중 선형 회귀에서는 3.66으로 줄어들었습니다.  
r2\_score를 보면 단순 선형 회귀에서는 0.75가 나왔으나 다중 선형 회귀에서는 0.87로 성능이 향상되었음을 알 수 있습니다.

# 모델 테스트 및 평가하기

다중 선형 회귀

판매량  
예측하기

- 다중 선형 회귀식:  $y = w_1x_1 + w_2x_2 + w_3x_3 + b$
- 다중 선형 회귀식의 기울기 계수( $w_1, w_2, w_3$ )와 절편( $b$ ) 구하기

```
1  # 기울기 계수
2  w2 = lr_model2.coef_
3  print(f 'slopes of TV: {w2[0]:.2f}')
4  print(f 'slopes of Radio: {w2[1]:.2f}')
5  print(f 'slopes of Newspaper: {w2[2]:.2f}')
6
7  # 절편
8  b2 = lr_model2.intercept_
9  print('Intercept is:', b2.round(2))
```

# 모델 테스트 및 평가하기

다중 선형 회귀

판매량  
예측하기



slopes of TV: 4.49  
slopes of Radio: 1.59  
slopes of Newspaper: 0.01  
Intercept is: 15.13



해석

$y = 4.49x_1 + 1.59x_2 + 0.01x_3 + 15.13$ 의 회귀식을 구할 수 있습니다. 다중 선형 회귀의 계수값을 보면 TV가 4.49로 가장 높고, Radio가 1.59, Newspaper가 0.01로 나옵니다. 즉, 판매는 계수값이 큰 TV의 영향을 가장 많이 받는 것을 확인할 수 있습니다. 이는 TV > Radio > Newspaper의 순으로 판매에 도움되는 광고 플랫폼임을 확인할 수 있습니다.

단순 선형 회귀에서는 회귀선을 2차원 평면상에 직선으로 그리기 쉬웠습니다.  
하지만 다중 선형 회귀에서는 차원이 높아짐에 따라 시각화하기 어렵습니다.

## </> Linear Regression 문제 해결 과정

문제 정의하기

광고 플랫폼에 따른 판매량을 예측해 볼까?

데이터 불러오기

캐글에서 광고 데이터셋 불러오기

데이터 처리하기

- 데이터 살펴보기(데이터 유형, 결측치, 속성명 등)
- 속성 간 상관관계 분석하기(히트맵으로 시각화)

모델 학습하기

- 선형 회귀 알고리즘으로 학습하기

모델 테스트 및 평가하기

- 테스트 데이터로 성능 평가하기
- 선형 회귀선 그리기/선형 회귀식 구하기



## 우리가 탐색한 정보

### 1. 이 활동에 필요한 데이터셋은 무엇이고, 이 데이터셋은 어디에서 수집할 수 있었나요?

데이터셋은 광고 데이터셋으로, 캐글에서 다운로드할 수 있습니다.

### 2. 모델 학습에 사용한 알고리즘은 무엇이었나요?

선형 회귀 알고리즘을 사용합니다. 선형 회귀는 회귀에서 가장 일반적인 알고리즘입니다. 단순 선형 회귀와 다중 선형 회귀가 있으며 단순 선형 회귀는 직선을 그려 연속적인 값을 예측하는 방법입니다.



### 3. 선형 회귀 모델 성능을 나타내는 성능 평가 지표는 무엇이었나요

테스트 데이터의 예측값과 실젯값을 비교하여 얼마나 차이가 나는지를 평가하는 지표로 평균 제곱 오차와 결정계 수를 산출합니다.

### 4. 이 활동에서 새롭게 알게 된 정보는 무엇이었나요?

경사 하강법(Gradient descent): 선형 회귀 알고리즘에서 실젯값과 예측값 사이의 오차를 최소화 할 수 있는 이상 적인 값을 찾는 방법입니다. 한 번에 오차를 최소화할 수 없기에 여러 번 반복을 통해 이상적인 값을 찾습니다.

지금까지 선형 회귀(Linear Regression)를 살펴보았습니다.

선형 회귀는 데이터를 가장 잘 설명하는 직선으로, 종속변수와 독립변수와의 선형 상관관계를 모델링하는 기법입니다. 선형 회귀에서 독립변수의 개수가 1개인 경우를 단순 선형 회귀, 2개 이상인 경우를 다중 선형 회귀라고 합니다.

이 활동에서는 광고 플랫폼에 따른 판매량을 예측하기 위하여 TV 광고비와 판매량에 대한 단순 선형 회귀 방법과 TV, Radio, Newspaper 광고비와 판매량에 대한 다중 선형 회귀 방법을 모두 다루어 보았습니다. 히트맵과 pairplot( ) 메소드를 이용한 시각화를 통하여 TV 광고비와 판매량이 강한 양의 상관관계가 있음을 확인할 수 있었으며, 실젯값과 예측값의 차이인 오차(error)를 그래프로 시각화하여 확인하였습니다.

성능 평가 지표로 평균 제곱 오차(MSE)와 결정계수( $R^2$ )를 살펴보았을 때, 단순 선형 회귀보다 다중 선형 회귀의 성능이 좀 더 향상되었음을 보았습니다. 최적의 선형 회귀식을 구하기 위해서 오차를 최소화하는 이상적인 가중치(w) 값을 구하는 경사 하강법을 구현해 보며 마무리하였습니다.