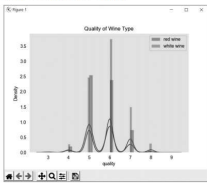
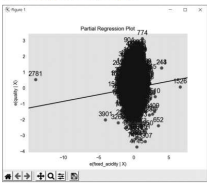


01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 분석 미리보기

와인 품질 등급 예측하기	
목표	와인 속성을 분석하여 품질 등급을 예측한다.
핵심 개념	기술 통계, 회귀 분석, t-검정, 히스토그램
데이터 수집	레드 와인/화이트 와인 데이터셋: 캘리포니아 어바인 대학의 마신러닝 저장소에서 다운로드
데이터 준비	수집한 데이터 파일 병합
데이터 탐색	1. 정보 확인: info() 2. 기술 통계 확인: describe(), unique(), value_counts()
데이터 모델링	1. 데이터를 두 그룹으로 비교 분석 • 그룹별 기술 통계 분석: describe() • t-검정: scipy 패키지의 ttest_ind() • 회귀 분석: statsmodels.formula.api 패키지의 ols() 2. 품질 등급 예측 • 샘플을 독립 변수(x)로 지정 → 회귀 분석 모델 적용 → 종속 변수(y)인 품질(quality) 예측
결과 시각화	
1. 히스토그램을 이용한 시각화	2. 부분 회귀 플롯을 이용한 시각화
	

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 목표 설정

- 목표: 와인의 속성을 분석한 뒤 품질 등급을 예측하는 것
- 데이터의 기술 통계를 구함
- 레드 와인과 화이트 와인 그룹의 품질에 대한 t-검정을 수행
- 와인 속성을 독립 변수로, 품질 등급을 종속 변수로 선형 회귀 분석을 수행

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 핵심 개념 이해

- 기술 통계 (요약 통계)
 - 데이터의 특성을 나타내는 수치를 이용해 분석하는 기본적인 통계 방법
 - 평균, 중앙값, 최빈값 등을 구할 수 있음
- 회귀 분석
 - 독립 변수, x와 종속 변수, y 간의 상호 연관성 정도를 파악하기 위한 분석 기법
 - 하나의 변수가 변함에 따라 대응 되는 변수가 어떻게 변하는지를 측정하는 것
 - 변수 간의 인과관계를 분석 할 때 많이 사용
 - 독립 변수가 한 개이면 단순 회귀 분석, 두 개 이상이면 다중 회귀 분석
 - 독립 변수와 종속 변수의 관계에 따라 선형 회귀 분석과 비선형 회귀 분석으로 나뉨
 - 선형 회귀 분석 식: $y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$
- t-검정
 - 데이터에서 찾은 평균으로 두 그룹에 차이가 있는지 확인하는 방법
 - 예) A와 B의 품질이 1등급인지 2등급인지에 따라 가격에 차이가 있는지를 확인할 때 사용
- 히스토그램
 - 데이터 값의 범위를 몇 개 구간으로 나누고 각 구간에 해당하는 값의 숫자나 상대적 빈도 크기를 차트로 나타낸 것

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 수집

- 캘리포니아 어바인 대학의 머신러닝 저장소에서 제공하는 오픈 데이터를 사용
- 다운로드한 파일은 My_Python/7장_data 폴더를 만든 후에 저장

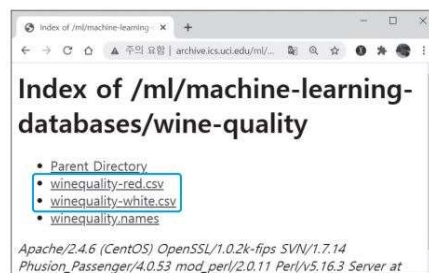


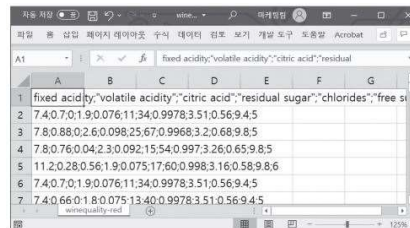
그림 7-1 와인 데이터 다운로드

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

1. 다운로드한 CSV 파일 정리하기

- 엑셀은 CSV 파일을 열 때 쉼표를 열 구분자로 사용하므로 열이 깨진 것처럼 보임
- 엑셀에서 세미콜론을 열 구분자로 인식하도록 파일을 다시 저장해야 함



	A	B	C	D	E	F	G
1	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	
2	7.4	0.7	1.9	0.076	11.34	0.9978	3.51
3	7.8	0.88	0.26	0.098	25.67	0.9968	3.2
4	7.8	0.76	0.04	2.3	0.092	15.54	0.997
5	11.2	0.28	0.56	1.9	0.075	17.60	0.998
6	7.4	0.7	1.9	0.076	11.34	0.9978	3.51
7	7.4	0.66	0.18	0.075	13.40	0.9978	3.51

그림 7-2 다운로드한 CSV 파일을 엑셀에서 열기

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

1. 다운로드한 CSV 파일 정리하기

1. 엑셀에서 열 구분자를 세미콜론으로 인식시키기

```
01 >>> import pandas as pd
02 >>> red_df = pd.read_csv('C:/Users/kmj/My_Python/7장_data/winequality-red.csv', sep = ';', header = 0, engine = 'python')
03 >>> white_df = pd.read_csv('C:/Users/kmj/My_Python/7장_data/winequality-white.csv', sep = ';', header = 0, engine = 'python')
04 >>> red_df.to_csv('C:/Users/kmj/My_Python/7장_data/winequality-red2.csv', index = False)
05 >>> white_df.to_csv('C:/Users/kmj/My_Python/7장_data/winequality-white2.csv', index = False)
```

- 01행 테이블 형태의 CSV 파일을 다루기 위해 pandas 라이브러리 패키지를 pd 이름으로 로드
- 02~03행 pandas의 read_csv() 함수를 사용해 CSV 파일을 읽어온음 이때 CSV 파일 데이터의 열 구분자를 세미콜론으로 지정하기 위해 sep 매개변수 값을 ';'으로 지정
- 04~05행 pandas로 읽은 CSV 데이터는 테이블 형태의 DataFrame 객체(red_df, white_df)에 있음 이 상태 그대로 CSV 파일로 저장

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

1. 다운로드한 CSV 파일 정리하기
2. 파이썬에서 저장한 CSV 파일을 엑셀에서 열어 상태를 확인

	A	B	C	D	E	F	G	H	I	J	K	L
1	acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	pH	sulphates	alcohol	quality	
2	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
3	7.8	0.88	0	2.6	0.086	25	67	0.9968	3.2	0.68	9.8	5
4	7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	5
5	11.2	0.28	0.56	1.9	0.075	17	60	0.998	3.16	0.58	9.8	6

(a) winequality-red2.csv

	A	B	C	D	E	F	G	H	I	J	K	L
1	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	pH	sulphates	alcohol	quality	
2	7	0.27	0.36	20.7	0.045	45	170	1.001	3	0.45	8.8	6
3	6.3	0.3	0.34	1.6	0.049	14	132	0.994	3.3	0.49	9.5	6
4	6.1	0.28	0.4	6.9	0.05	30	97	0.9951	3.26	0.44	10.1	6
5	7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.4	9.9	6

(b) winequality-white2.csv

그림 7-3 엑셀의 열 구분자를 수정한 뒤 저장한 파일 다시 열기

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

2. 데이터 병합하기
1. 레드 와인과 화이트 와인 파일 합치기

```
01 >>> red_df.head()
fixed acidity volatile acidity citric acid ... sulphates alcohol quality
0 7.4 0.70 0.00 ... 0.56 9.4 5
1 7.8 0.88 0.00 ... 0.68 9.8 5
2 7.8 0.76 0.04 ... 0.65 9.8 5
3 11.2 0.28 0.56 ... 0.58 9.8 6
4 7.4 0.70 0.00 ... 0.56 9.4 5

[5 rows x 12 columns]
02 >>> red_df.insert(0, column = 'type', value = 'red')
03 >>> red_df.head()
type fixed acidity volatile acidity ... sulphates alcohol quality
0 red 7.4 0.70 ... 0.56 9.4 5
1 red 7.8 0.88 ... 0.68 9.8 5
2 red 7.8 0.76 ... 0.65 9.8 5
3 red 11.2 0.28 ... 0.58 9.8 6
4 red 7.4 0.70 ... 0.56 9.4 5

[5 rows x 13 columns]
04 >>> red_df.shape
(1599, 13)
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

2. 데이터 병합하기

1. 레드 와인과 화이트 와인 파일 합치기

```

05 >>> white_df.head()
fixed acidity volatile acidity citric acid ... sulphates alcohol quality
0 7.0 0.27 0.36 ... 0.45 8.8 6
1 6.3 0.30 0.34 ... 0.49 9.5 6
2 8.1 0.28 0.40 ... 0.44 10.1 6
3 7.2 0.23 0.32 ... 0.40 9.9 6
4 7.2 0.23 0.32 ... 0.40 9.9 6

[5 rows x 12 columns]
06 >>> white_df.insert(0, column = 'type', value = 'white')
07 >>> white_df.head()
type fixed acidity volatile acidity ... sulphates alcohol quality
0 white 7.0 0.27 ... 0.45 8.8 6
1 white 6.3 0.30 ... 0.49 9.5 6
2 white 8.1 0.28 ... 0.44 10.1 6
3 white 7.2 0.23 ... 0.40 9.9 6
4 white 7.2 0.23 ... 0.40 9.9 6

[5 rows x 13 columns]
08 >>> white_df.shape
(4898, 13)
09 >>> wine = pd.concat([red_df, white_df])
10 >>> wine.shape
(6497, 13)
11 >>> wine.to_csv('C:/Users/kmj/My_Python/7장_data/wine.csv', index = False)

```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

2. 데이터 병합하기

1. 레드 와인과 화이트 와인 파일 합치기

- [01~04행] 레드 와인 파일을 읽고 데이터프레임에 'type' 열 삽입하기
 - 01행 red_df에 저장된 내용을 위에서부터 5개(0번~4번) 행만 출력하여 확인
 - 02행 이름이 'type'이고 값이 'red'인 열을 만들어 index = 0(첫 번째 열) 자리에 삽입
 - 03행 red_df에 저장된 내용을 위에서부터 5개(0번~4번) 행만 다시 출력해 삽입된 'type' 열을 확인
 - 04행 red_df.shape를 이용하여 현재 red_df의 크기를 '(행의 개수, 열의 개수)' 형태로 확인
- [05~08행] 화이트 와인 파일을 읽고 데이터프레임에 'type' 열 삽입하기
 - 05행 white_df에 저장된 내용을 위에서부터 5개(0번~4번) 행만 출력하여 확인
 - 06행 이름이 'type'이고 값이 'white'인 열을 만들어 index = 0(첫 번째 열) 자리에 삽입
 - 07행 white_df에 저장된 내용을 위에서부터 5개(0번~4번) 행만 다시 출력해 삽입된 'type' 열을 확인
 - 08행 white_df.shape를 이용하여 현재 white_df의 크기를 '(행의 개수, 열의 개수)' 형태로 확인
- [09~10행] red_df와 white_df를 하나의 데이터프레임 형식으로 결합하기
 - 09행 pd.concat() 함수를 이용하여 red_df와 white_df를 결합
 - 10행 결합된 wine의 현재 크기를 '(행의 개수, 열의 개수)' 형태로 확인
 - 11행 wine을 CSV 파일로 저장

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 준비

2. 데이터 병합하기
2. 결합된 파일 확인하기

	A	B	C	D	E	F	G	H	I	J	K	L	M
1596	red	6.2	0.6	0.08	2	0.09	32	44	0.9949	3.45	0.58	10.5	5
1597	red	5.9	0.55	0.1	2.2	0.062	39	51	0.99512	3.52	0.76	11.2	6
1598	red	6.3	0.51	0.13	2.3	0.076	29	40	0.99574	3.42	0.75	11	6
1599	red	5.9	0.645	0.12	2	0.075	32	44	0.99547	3.57	0.71	10.2	5
1600	red	6	0.31	0.47	3.6	0.067	18	42	0.99549	3.39	0.66	11	6
1601	white	7	0.27	0.36	20.7	0.045	45	170	1.001	3	0.45	8.8	6
1602	white	6.3	0.3	0.34	1.6	0.049	14	132	0.994	3.3	0.49	9.5	6
1603	white	8.1	0.28	0.4	6.9	0.05	30	97	0.9951	3.26	0.44	10.1	6
1604	white	7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.4	9.9	6

그림 7-4 레드 와인 데이터셋과 화이트 와인 데이터셋이 결합된 wine.csv 파일

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 탐색

1. 기본 정보 확인하기

```
01 >>> print(wine.info())
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6497 entries, 0 to 4897
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  -
0   type                   6497 non-null   object
1   fixed acidity           6497 non-null   float64
2   volatile acidity        6497 non-null   float64
3   citric acid             6497 non-null   float64
4   residual sugar          6497 non-null   float64
5   chlorides               6497 non-null   float64
6   free sulfur dioxide     6497 non-null   float64
7   total sulfur dioxide    6497 non-null   float64
8   density                 6497 non-null   float64
9   pH                     6497 non-null   float64
10  sulphates               6497 non-null   float64
11  alcohol                 6497 non-null   float64
12  quality                 6497 non-null   int64
dtypes: float64(11), int64(1), object(1)
memory usage: 710.6+ KB
None
```

- 전체 샘플은 6,497개이고 속성을 나타내는 열은 13개, 각 속성의 이름은 type부터 quality까지
- 속성 중에서 실수 타입(float64)은 11개, 정수 타입(int64)은 1개(quality), 객체 타입(object)이 1개(type)
- 독립 변수(x)는 type부터 alcohol 까지 12개, 종속 변수(y)는 1개(quality)

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 탐색

2. 함수를 사용해 기술 통계 구하기

```
01 >>> wine.columns = wine.columns.str.replace(' ', '_')
02 >>> wine.head()
   type  fixed acidity  volatile acidity  ...  sulphates  alcohol  quality
0  red         7.4         0.70         ...     0.56      9.4        5
1  red         7.8         0.88         ...     0.68      9.8        5
2  red         7.8         0.76         ...     0.65      9.8        5
3  red        11.2         0.28         ...     0.58      9.8        6
4  red         7.4         0.70         ...     0.56      9.4        5
[5 rows x 13 columns]
03 >>> wine.describe()
      fixed_acidity  volatile_acidity  ...  alcohol  quality
count  6497.000000    6497.000000  ...  6497.000000  6497.000000
mean     7.215307     0.339666  ...  10.491801    5.818378
std     1.296434     0.164636  ...   1.192712    0.873255
Min      3.800000     0.080000  ...   8.000000    3.000000
25%      6.400000     0.230000  ...   9.500000    5.000000
50%      7.000000     0.290000  ...  10.300000    6.000000
75%      7.700000     0.400000  ...  11.300000    6.000000
max     15.900000     1.580000  ...  14.900000    9.000000
[8 rows x 12 columns]
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 탐색

2. 함수를 사용해 기술 통계 구하기

```
04 >>> sorted(wine.quality.unique())
[3, 4, 5, 6, 7, 8, 9]
05 >>> wine.quality.value_counts()
6    2836
5    2138
7    1079
4     216
8     193
3       30
9         5
Name: quality, dtype: int64
9         5
Name: quality, dtype: int64
```

- [01~04행] 열이름 정렬하기
 - 01행 열 이름에 공백이 있으면 밑줄로 바꾼 뒤 한 단어로 연결
 - 02행 변경된 열 이름을 확인
 - 03행 describe() 함수를 사용하여 속성별 개수, 평균, 표준편차, 최소값, 전체 데이터 백분율에 대한 25번째 백분위수(25%), 중앙값인 50번째 백분위수(50%), 75번째 백분위수(75%) 그리고 100번째 백분위수인 최대값max를 출력
 - 04행 wine.quality.unique() 함수를 사용하여 quality 속성값 중에서 유일한 값을 출력
이를 통해 와인 품질 등급quality은 3, 4, 5, 6, 7, 8, 9의 7개 등급이 있다는 것을 알 수 있음
 - 05행 quality.value_counts() 함수는 quality 속성값에 대한 빈도수를 보여줌
6등급인 샘플이 2,826개로 가장 많고, 9등급인 샘플이 5개로 가장 적은 것을 알 수 있음

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

1. describe() 함수로 그룹 비교하기

```
01 >>> wine.groupby('type')['quality'].describe()
      count      mean      std  min  25%  50%  75%  max
type
Red    1599.0  5.636023  0.807569  3.0  5.0  6.0  6.0  8.0
White  4898.0  5.877909  0.885639  3.0  5.0  6.0  6.0  9.0
02 >>> wine.groupby('type')['quality'].mean()
type
red      5.636023
white    5.877909
Name: quality, dtype: float64
03 >>> wine.groupby('type')['quality'].std()
type
red      0.807569
white    0.885639
Name: quality, dtype: float64
04 >>> wine.groupby('type')['quality'].agg(['mean', 'std'])
      mean      std
type
red      5.636023  0.807569
white    5.877909  0.885639
```

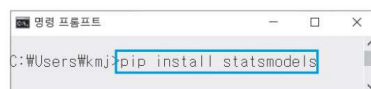
- 01행 레드 와인과 화이트 와인을 구분하는 속성인 type을 기준으로 그룹을 나눈 뒤 그룹 안에서 quality 속성을 기준으로 기술 통계를 구함
- 02~04행 기술 통계 전부를 구할 때는 describe() 함수를 사용하지만 mean() 함수로 평균만 구하거나 std() 함수로 표준편차만 따로 구할 수도 있음
mean() 함수와 std() 함수를 묶어서 한 번에 사용하려면 04행과 같이 agg() 함수를 사용

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

2. t-검정과 회귀 분석으로 그룹 비교하기

- t-검정을 위해서는 scipy 라이브러리 패키지를 사용
- 회귀 분석을 위해서는 statsmodels 라이브러리 패키지를 사용
- 명령 프롬프트 창에서 다음과 같이 입력하여 statsmodels 패키지를 설치



```
C:\Users\Wknj>pip install statsmodels
```

```
01 >>> from scipy import stats
02 >>> from statsmodels.formula.api import ols, glm
03 >>> red_wine_quality = wine.loc[wine['type'] == 'red', 'quality']
04 >>> white_wine_quality = wine.loc[wine['type'] == 'white', 'quality']
05 >>> stats.ttest_ind(red_wine_quality, white_wine_quality, equal_var = False)
Ttest_indResult(statistic = -10.149363059143164, pvalue = 8.168348870049682e-24)
06 >>> Rformula = 'quality ~ fixed_acidity + volatile_acidity + citric_acid +
residual_sugar + chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
density + pH + sulphates + alcohol
07 >>> regression_result = ols(Rformula, data = wine).fit()
08 >>> regression_result.summary()
```


01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

2. t-검정과 회귀 분석으로 그룹 비교하기

```
<class 'statsmodels.iolib.summary.Summary'>
```

OLS Regression Results						
Dep. Variable:	quality	R-squared:	0.292			
Model:	OLS	Adj. R-squared:	0.291			
Method:	Least Squares	F-statistic:	243.3			
Date:	Tue, 28 Apr 2020	Prob (F-statistic):	0.00			
Time:	16:19:39	Log-Likelihood:	-7215.5			
No. Observations:	6497	AIC:	1.445e+04			
Df Residuals:	6485	BIC:	1.454e+04			
Df Model:	11					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	55.7627	11.894	4.688	0.000	32.447	79.079
fixed_acidity	0.0677	0.016	4.346	0.000	0.037	0.098
volatile_acidity	-1.3279	0.077	-17.162	0.000	-1.480	-1.176
citric_acid	-0.1097	0.080	-1.377	0.168	-0.266	0.046
residual_sugar	0.0436	0.005	8.449	0.000	0.033	0.054
chlorides	-0.4837	0.333	-1.454	0.146	-1.136	0.168
free_sulfur_dioxide	0.0060	0.001	7.948	0.000	0.004	0.007
total_sulfur_dioxide	-0.0025	0.000	-8.969	0.000	-0.003	-0.002
density	-54.9669	12.137	-4.529	0.000	-78.760	-31.173
pH	0.4393	0.090	4.861	0.000	0.262	0.616
sulphates	0.7683	0.076	10.092	0.000	0.619	0.917
alcohol	0.2670	0.017	15.963	0.000	0.234	0.300

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

2. t-검정과 회귀 분석으로 그룹 비교하기

Omnibus:	144.075	Durbin-Watson:	1.646
Prob(Omnibus):	0.000	Jarque-Bera (JB):	324.712
Skew:	-0.006	Prob(JB):	3.09e-71
urtosis:	4.095	Cond. No.	2.49e+05

```
=====
```

Warnings:

[1] Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 2.49e+05. This might indicate that there are strong multicollinearity or other numerical problems.

```
=====
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

2. t-검정과 회귀 분석으로 그룹 비교하기

- 01~02행
 - t-검정에 필요한 scipy 패키지의 stats 함수와 회귀 분석에 필요한 statsmodels.formula.api 패키지의 ols, glm 함수를 로드
- [03~04행] 그룹 분리하기
 - 03행 레드 와인 샘플의 quality 값만 찾아서 red_wine에 저장
 - 04행 화이트 와인 샘플의 quality 값만 찾아서 white_wine에 저장
 - 05행 scipy 패키지의 stats.ttest_ind() 함수를 사용하여 t-검정을 하고 두 그룹 간 차이를 확인
- [06~08행] 선형 회귀 분석 수행하기
 - 06행 선형 회귀 분석식의 종속 변수(y)와 독립 변수(x1~x10)를 구성
 - 07행 선형 회귀 모델 중에서 OLS Ordinary Least Squares 모델을 사용
 - 08행 선형 회귀 분석과 관련된 통계값을 출력

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

3. 회귀 분석 모델로 새로운 샘플의 품질 등급 예측하기

```
01 >>> sample1 = wine[wine.columns.difference(['quality', 'type'])]
02 >>> sample1 = sample1[0:5][:]
03 >>> sample1_predict = regression_result.predict(sample1)
04 >>> sample1_predict
0    4.997607
1    4.924993
2    5.034663
3    5.680333
4    4.997607
dtype: float64
05 >>> wine[0:5]['quality']
0    5
1    5
2    5
3    6
4    5
Name: quality, dtype: int64
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

3. 회귀 분석 모델로 새로운 샘플의 품질 등급 예측하기

```
06 >>> data = {"fixed_acidity": [8.5, 8.1], "volatile_acidity": [0.8, 0.5],
"citric_acid": [0.3, 0.4], "residual_sugar": [6.1, 5.8], "chlorides": [0.055,
0.04], "free_sulfur_dioxide": [30.0, 31.0], "total_sulfur_dioxide": [98.0,
99], "density": [0.996, 0.91], "pH": [3.25, 3.01], "sulphates": [0.4, 0.35],
"alcohol": [9.0, 0.88]}
07 >>> sample2 = pd.DataFrame(data, columns= sample1.columns)
08 >>> sample2
   alcohol chlorides ... total_sulfur_dioxide volatile_acidity
0      9.00    0.055 ...             98.0                0.8
1      0.88    0.040 ...             99.0                0.5
[2 rows x 11 columns]
09 >>> sample2_predict = regression_result.predict(sample2)
10 >>> sample2_predict
-0      4.809094
1      7.582129
dtype: float64
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 데이터 모델링

3. 회귀 분석 모델로 새로운 샘플의 품질 등급 예측하기

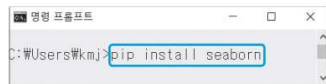
- [01~02행] 예측에 사용할 첫 번째 샘플 데이터 만들기
 - 01행 wine에서 quality와 type 열은 제외하고, 회귀 분석 모델에 사용할 독립 변수만 추출하여 sample1에 저장
 - 02행 sample1에 있는 샘플 중에서 0번~4번 5개 샘플만 추출하고, sample1에 다시 저장하여 예측에 사용할 샘플을 제작
- [03~05행] 첫 번째 샘플의 quality 예측하기
 - 03행 샘플 데이터를 회귀 분석 모델 regression_result의 예측 함수 predict()에 적용하여 수행한 뒤 결과 예측값을 sample1_predict에 저장
 - 04행 sample1_predict를 출력하여 예측한 quality를 확인
 - 05행 wine에서 0번부터 4번까지 샘플의 quality 값을 출력하여 sample1_predict이 맞게 예측되었는지 확인
- [06~08행] 예측에 사용할 두 번째 샘플 데이터 만들기
 - 06행 회귀식에 사용한 독립 변수에 대입할 임의의 값을 딕셔너리 형태로 제작
 - 07행 딕셔너리 형태의 값과 sample1의 열 이름만 뽑아 데이터프레임으로 묶은 sample2를 제작
 - 08행 sample2를 출력하여 제대로 구성되었는지 확인
- [09~10행] 두 번째 샘플의 quality 예측하기
 - 09행 샘플 데이터를 회귀 분석 모델 regression_result의 예측 함수 predict()에 적용하여 수행한 뒤 결과 예측값을 sample2_predict에 저장
 - 10행 sample2_predict를 출력하여 예측한 quality를 확인

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 결과 시각화

1. 와인 유형에 따른 품질 등급 히스토그램 그리기

1. 명령 프롬프트 창에서 다음 명령을 입력하여 seaborn 라이브러리 패키지를 설치
그 후 파이썬 셸 창으로 돌아와 임포트



2. 파이썬 셸 창에 명령을 입력

```
01 >>> import matplotlib.pyplot as plt
02 >>> import seaborn as sns
03 >>> sns.set_style('dark')
04 >>> sns.distplot(red_wine_quality, kde = True, color = "red", label = 'red
    wine')
    <matplotlib.axes_subplots.AxesSubplot object at 0x000001C843F36348>
05 >>> sns.distplot(white_wine_quality, kde = True, label = 'white wine')
    <matplotlib.axes_subplots.AxesSubplot object at 0x000001C843F36348>
06 >>> plt.title("Quality of Wine Type")
    Text(0.5, 1.0, 'Quality of Wine Type')
07 >>> plt.legend()
    <matplotlib.legend.Legend object at 0x0000014F20DFA548>
08 >>> plt.show()
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 결과 시각화

1. 와인 유형에 따른 품질 등급 히스토그램 그리기

2. 파이썬 셸 창에 명령을 입력
 - 01~02행 시각화에 필요한 패키지를 로드
 - [03~08행] 커널 밀도 추정(kde)을 적용한 히스토그램 그리기
 - 03행 히스토그램 차트의 배경색 스타일을 설정
 - 04행 레드 와인에 대한 distplot 객체를 생성
 - 05행 화이트 와인에 대한 distplot 객체를 생성
 - 06행 차트 제목을 설정
 - 07행 차트 범례를 설정
 - 08행 설정한 내용대로 차트를 표시

- x축: qualit
- y축: 확률 밀도 함수값

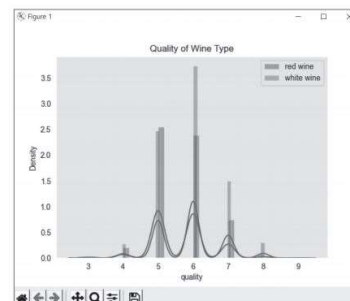


그림 7-5 와인 유형에 따른 품질 등급 히스토그램

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 결과 시각화

2. 부분 회귀 플롯으로 시각화하기

- 독립 변수가 2개 이상인 경우에는 부분 회귀 플롯을 사용하여 하나의 독립 변수가 종속 변수에 미치는 영향력을 시각화 함으로써 결과를 분석할 수 있음

```
01 >>> import statsmodels.api as sm
02 >>> others = list(set(wine.columns).difference(set(["quality", "fixed_acidity"])))
03 >>> p, resid = sm.graphics.plot_partregress("quality", "fixed_acidity", others, data = wine, ret_coors = True)
04 >>> plt.show()
05 >>> fig = plt.figure(figsize = (8, 13))
06 >>> sm.graphics.plot_partregress_grid(regression_result, fig = fig)
    <Figure size 800x1300 with 12 Axes>
07 >>> plt.show()
```

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 결과 시각화

2. 부분 회귀 플롯으로 시각화하기

- 01행 부분 회귀 계산을 위해 statsmodels.api를 로드
- [02~04행] fixed_acidity가 종속 변수 quality에 미치는 영향력을 시각화하기
 - 02행 부분 회귀에 사용한 독립 변수와 종속 변수를 제외한 나머지 변수 이름을 리스트 others로 추출
 - 03행 나머지 변수는 고정하고 fixed_acidity가 종속 변수 quality에 미치는 영향에 부분회귀를 수행
 - 04행 부분 회귀의 결과를 플롯으로 시각화하여 나타냄

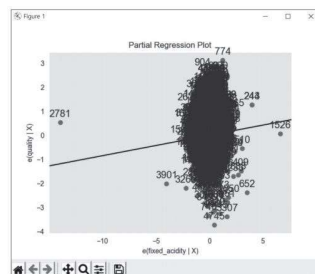


그림 7-6 독립 변수 fixed_acidity와 종속 변수 quality에 대한 부분 회귀 시각화

01. [기술 통계 분석 + 그래프] 와인 품질 예측하기

■ 결과 시각화

2. 부분 회귀 플롯으로 시각화하기

- [05~07행] 각 독립 변수가 종속 변수 quality에 미치는 영향력을 시각화하기
 - 05행 차트의 크기를 지정
 - 06행 다중 선택 회귀 분석 결과를 가지고 있는 regression_result를 이용해 각 독립 변수의 부분 회귀 플롯을 구함
 - 07행 부분 회귀 결과를 플롯으로 시각화하여 나타냄

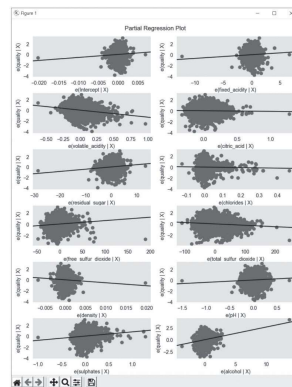


그림 7-7 각자의 독립 변수와 종속 변수 quality에 대한 부분 회귀 시각화

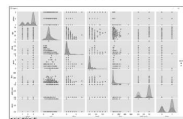
02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 분석 미리보기

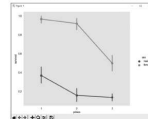
타이타닉호 생존을 분석하기	
목표	타이타닉호 승객 변수를 분석하여 생존율과의 상관관계를 찾는다.
핵심 개념	상관 분석, 상관 계수, 피어슨 상관 계수, 히트맵
데이터 수집	타이타닉 데이터: seaborn 내장 데이터셋
데이터 준비	결측치 치환, 중앙값 치환, 최빈값 치환
데이터 탐색	1. 정보 확인: info() 2. 차트를 통한 데이터 탐색: pie(), countplot()
데이터 모델링	1. 모든 변수 간 상관 계수 구하기 2. 지정된 두 변수 간 상관 계수 구하기

결과 시각화

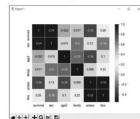
1. 산점도를 이용한 시각화



2. 특정 변수 간 상관관계 시각화



3. 히트맵을 이용한 시각화



02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 분석 미리보기

- 타이타닉호의 생존자와 관련된 변수의 상관관계를 찾아봄
- 생존과 가장 상관도가 높은 변수는 무엇인지 분석
- 상관 분석을 위해 피어슨 상관 계수를 사용
- 변수 간의 상관관계는 시각화하여 분석

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 핵심 개념 이해

■ 상관 분석

- 두 변수가 어떤 선형적 관계에 있는지를 분석하는 방법
- 두 변수는 서로 독립적이거나 상관된 관계일 수 있는데, 두 변수의 관계의 강도를 상관관계 라고함
- 상관 분석에서는 상관관계의 정도를 나타내는 단위로 모상관 계수 ρ 를 사용
- 상관 계수는 두 변수가 연관된 정도를 나타낼 뿐 인과 관계를 설명하지 않으므로 정확한 예측치를 계산할 수는 없음

• 단순 상관 분석

- 두 변수가 어느 정도 강한 관계에 있는지 측정

• 다중 상관 분석

- 세 개 이상의 변수 간 관계의 강도를 측정
- 편상관 분석: 다른 변수와의 관계를 고정하고 두 변수 간 관계의 강도를 나타내는 것

■ 상관 계수 ρ

- 변수 간 관계의 정도(0~1)와 방향(+, -)을 하나의 수치로 요약해주는 지수로 -1에서 +1 사이의 값을 가짐
- 상관 계수가 +이면 양의 상관관계이며 한 변수가 증가하면 다른 변수도 증가
- 상관 계수가 -이면 음의 상관관계이며 한 변수가 증가할 때 다른 변수는 감소
- 0.0 ~ 0.2: 상관관계가 거의 없음
- 0.2 ~ 0.4: 약한 상관관계가 있음
- 0.4 ~ 0.6: 상관관계가 있음
- 0.6 ~ 0.8: 강한 상관관계가 있음
- 0.8 ~ 1.0: 매우 강한 상관관계가 있음

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 수집

```
01 >>> import seaborn as sns
02 >>> import pandas as pd
03 >>> titanic = sns.load_dataset("titanic")
04 >>> titanic.to_csv('C:/Users/kmj/My_Python/7장_data/titanic.csv', index = False)
```

- 01행 seaborn 패키지를 로드
- 03행 titanic 데이터를 로드
- 04행 데이터를 CSV 파일로 저장

■ 데이터 준비

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embarked_town	alive	alone
0	3	male	22	1	0	7.25	S	Third	man	TRUE		Southampton	no	FALSE
1	1	female	38	1	0	53.00	C	First	woman	FALSE	C	Cherbourg	yes	FALSE
1	3	female	26	0	0	5.12	S	Third	woman	FALSE		Southampton	yes	TRUE
1	1	female	35	1	0	51.00	S	First	woman	FALSE	C	Southampton	yes	FALSE
0	3	male	35	0	0	8.00	S	Third	man	TRUE		Southampton	no	TRUE
0	3	male	0	0	0	51.00	Q	Third	man	TRUE		Queensbury	no	TRUE

그림 7-8 다운로드한 파일(titanic.csv) 열기

- 저장한 titanic.csv 파일을 열어서 데이터 정리 작업이 필요한지 확인

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 준비

```
01 >>> titanic.isnull().sum()
survived      0
pclass        0
sex           0
age          177
sibsp         0
parch         0
fare          0
embarked      2
class         0
who           0
adult_male    0
deck         688
embark_town   2
alive         0
alone         0
dtype: int64
02 >>> titanic['age'] = titanic['age'].fillna(titanic['age'].median())
03 >>> titanic['embarked'].value_counts()
S    644
C    168
Q     77
Name: embarked, dtype: int64
06 >>> titanic['embark_town'] = titanic['embark_town'].fillna('Southampton')
07 >>> titanic['deck'].value_counts()
C     59
B     47
D     33
E     32
A     15
F     13
G      4
Name: deck, dtype: int64
08 >>> titanic['deck'] = titanic['deck'].fillna('C')
09 >>> titanic.isnull().sum()
survived      0
pclass        0
sex           0
age           0
sibsp         0
parch         0
fare          0
embarked      0
class         0
who           0
adult_male    0
deck          0
embark_town   0
alive         0
alone         0
dtype: int64
```


02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 탐색

1. 데이터의 기본 정보 탐색하기

```
01 >>> titanic.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
#   Column      Non-Null Count  Dtype
---  -
0    survived    891 non-null    int64
1    pclass      891 non-null    int64
2    sex         891 non-null    object
3    age         891 non-null    float64
4    sibsp       891 non-null    int64
5    parch       891 non-null    int64
6    fare        891 non-null    float64
7    embarked    891 non-null    object
8    class       891 non-null    category
9    who         891 non-null    object
10   adult_male  891 non-null    bool
11   deck        891 non-null    category
12   embark_town  891 non-null    object
13   alive        891 non-null    object
14   alone       891 non-null    bool
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.6+ KB

02 >>> titanic.survived.value_counts()
0    549
1    342
Name: survived, dtype: int64
```

- 01행 타이타닉 데이터의 기본 정보를 확인
- 02행 survived 속성값의 빈도를 확인
- 전체 샘플의 수: 891개이고 속성은 15개
- 샘플 891명 중에서 생존자는 342명이고 사망자는 549명
- pclass, class: 객실 등급
- sibsp: 함께 탑승한 형제자매와 배우자 수
- parch: 함께 탑승한 부모/자식 수
- embarked, embark_town: 탑승 항구
- adult_male: 성인 남자 여부
- alone: 동행 여부를 True/False로 나타냄

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 탐색

2. 차트를 그려 데이터를 시각적으로 탐색하기

```
01 >>> import matplotlib.pyplot as plt
02 >>> [ax = plt.subplots(1, 2, figsize = (10, 5))]
03 >>> titanic['survived'][titanic['sex'] == 'male'].value_counts().plot.
pie(explode = [0,0.1], autopct = '%1.1f%%', ax = ax[0], shadow = True)
<matplotlib.axes_subplots.AxesSubplot object at 0x000001DD48E0C648>
04 >>> titanic['survived'][titanic['sex'] == 'female'].value_counts().plot.
pie(explode = [0,0.1], autopct = '%1.1f%%', ax = ax[1], shadow = True)
<matplotlib.axes_subplots.AxesSubplot object at 0x000001DD491C35C8>
05 >>> ax[0].set_title('Survived (Male)')
Text(0.5, 1.0, 'Survived (Male)')
06 >>> ax[1].set_title('Survived (Female)')
Text(0.5, 1.0, 'Survived (Female)')
07 >>> plt.show()
```

- 01행 차트를 그리기 위해 matplotlib.pyplot를 로드
- [02~07행] 남자 승객과 여자 승객의 생존율을 pie 차트로 그리기
 - 02행 한 줄에 두 개의 차트를 그리도록 하고 크기를 설정
 - 03행 첫 번째 pie 차트는 남자 승객의 생존율을 나타내도록 설정
 - 04행 두 번째 pie 차트는 여자 승객의 생존율을 나타내도록 설정
 - 05행 첫 번째 차트의 제목을 설정
 - 06행 두 번째 차트의 제목을 설정
 - 07행 구성된 차트를 나타낸다.

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 탐색

2. 차트를 그려 데이터를 시각적으로 탐색하기

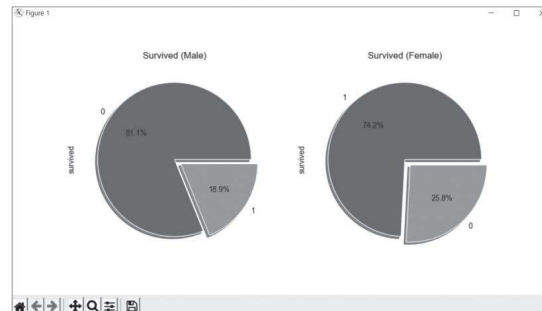


그림 7-9 성별에 따른 생존율 차트

- 남자 승객의 생존율: 18.9%
- 여자 승객의 생존율 74.2%

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 탐색

3. 등급별 생존자 수를 차트로 나타내기

```
01 >>> sns.countplot('pclass', hue = 'survived', data = titanic)
    <matplotlib.axes_subplots.AxesSubplot object at 0x000001DD48E03EC8>
02 >>> plt.title('Pclass vs Survived')
    Text(0.5, 1.0, 'Pclass vs Survived')
03 >>> plt.show()
```

- 01행 pclass 유형 1,2,3을 x축으로 하고 survived =0과 survived =1의 개수를 계산하여 y축으로 하는 countplot을 설정
- 02행 차트 제목을 설정
- 03행 구성한 차트를 나타냄
- 생존자(1)는 1등급에서 가장 많음
- 사망자(0)는 3등급에서 월등히 많음

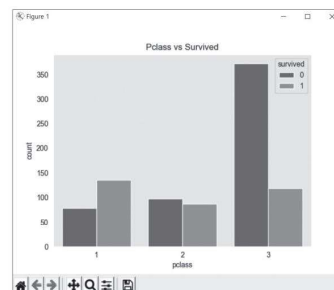


그림 7-10 객실 등급에 따른 생존자 수

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 모델링

1. 상관 분석을 위한 상관 계수 구하고 저장하기

```
01 >>> titanic_corr = titanic.corr(method = 'pearson')
02 >>> titanic_corr
survived    pclass    age    ...    fare    adult_male    alone
survived    1.000000  -0.338481 -0.064910 ... 0.257307 -0.557080 -0.203367
pclass      -0.338481  1.000000  -0.339898 ... -0.549500  0.094035  0.135207
age          -0.064910 -0.339898  1.000000 ... 0.096688  0.247704  0.171647
sibsp       -0.035322  0.083081 -0.233296 ... 0.159651 -0.253586 -0.584471
parch       0.081629  0.018443 -0.172482 ... 0.216225 -0.349943 -0.583398
fare         0.257307 -0.549500  0.096688 ... 1.000000 -0.182024 -0.271832
adult_male -0.557080  0.094035  0.247704 ... -0.182024  1.000000  0.404744
alone        0.203367  0.135207  0.171647 ... -0.271832  0.404744  1.000000
[8 rows x 8 columns]
03 >>> titanic_corr.to_csv('C:/Users/kmj/My_Python/7장_data/titanic_corr.csv',
index = False)
```

- 01행 피어슨 상관 계수를 적용하여 상관 계수를 구함
- 02행 상관 계수를 출력
- 03행 상관 계수를 CSV 파일로 저장

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 데이터 모델링

2. 상관 계수 확인하기

- 남자 성인(adult_male): 생존(survived)과 음의 상관관계
- 객실 등급(pclass): 음의 상관
- 관계, 객실 요금fare은 양의 상관관계
- 동행 없이 혼자 탑승한 경우(alone): 생존율이 떨어진다는 상관관계

	A	B	C	D	E	F	G	H
1 survived	1							
2 pclass	-0.338481	1						
3 age	-0.064910	-0.339898	1					
4 sibsp	-0.035322	0.083081	-0.233296	1				
5 parch	0.081629	0.018443	-0.172482	0.159651	1			
6 fare	0.257307	-0.549500	0.096688	0.159651	-0.253586	1		
7 adult_male	-0.557080	0.094035	0.247704	-0.182024	0.159651	-0.253586	1	
8 alone	0.203367	0.135207	0.171647	-0.271832	0.404744	0.404744	1.000000	1

그림 7-11 titanic_corr.csv 파일에서 상관 계수 확인

3. 특정 변수 사이의 상관 계수 구하기

```
01 >>> titanic['survived'].corr(titanic['adult_male'])
-0.5570800422053259
02 >>> titanic['survived'].corr(titanic['fare'])
0.2573065223849622
```

- [01~02행] 두 변수 사이의 상관 계수 구하기
 - 01행 survived와 adult_male 변수 사이의 상관 계수를 구함
 - 02행 survived와 fare 변수 사이의 상관 계수를 구함

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 결과 시각화

1. 산점도로 상관 분석 시각화하기

```
01 >>> sns.pairplot(titanic, hue = 'survived')
    <seaborn.axisgrid.PairGrid object at 0x000001710D852A58>
02 >>> plt.show()
```

- [01~02행] 변수 간의 상관 분석 시각화를 위해 pairplot() 그리기
 - 01행 pairplot() 함수를 사용하여 타이타닉 데이터의 차트를 그림, hue는 종속 변수를 지정
 - 02행 pairplot을 나타냄

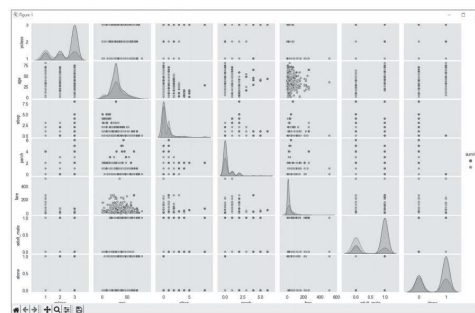


그림 7-12 pairplot() 함수를 이용한 산점도

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 결과 시각화

1. 산점도로 상관 분석 시각화하기

```
01 >>> sns.pairplot(titanic, hue = 'survived')
    <seaborn.axisgrid.PairGrid object at 0x000001710D852A58>
02 >>> plt.show()
```

- [01~02행] 변수 간의 상관 분석 시각화를 위해 pairplot() 그리기
 - 01행 pairplot() 함수를 사용하여 타이타닉 데이터의 차트를 그림, hue는 종속 변수를 지정
 - 02행 pairplot을 나타냄

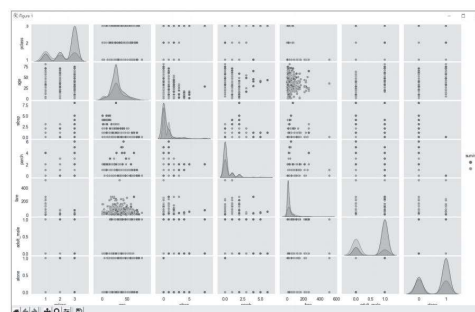


그림 7-12 pairplot() 함수를 이용한 산점도

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 결과 시각화

2. 두 변수의 상관관계 시각화하기

```
01 >>> sns.catplot(x = 'pclass', y = 'survived', hue = 'sex', data = titanic, kind = 'point')
<seaborn.axisgrid.FacetGrid object at 0x000001DD44EB4888>
02 >>> plt.show()
```

- [01~02행] 생존자의 객실 등급과 성별 관계를 catplot()로 그리기
 - 01행 catplot() 함수를 사용하여 pclass와 survived 변수의 관계를 차트로 그림
hue인자를 이용하여 종속 변수를 sex로 지정
 - 02행 catplot을 나타냄

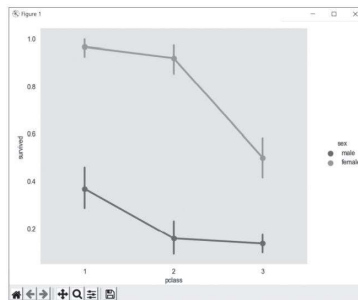


그림 7-13 객실 등급과 생존의 상관관계를 나타내는 catplot 차트

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 결과 시각화

2. 변수 사이의 상관 계수를 히트맵으로 시각화하기

```
01 >>> def category_age(x):
    if x < 10:
        return 0
    elif x < 20:
        return 1
    elif x < 30:
        return 2
    elif x < 40:
        return 3
    elif x < 50:
        return 4
    elif x < 60:
        return 5
    elif x < 70:
        return 6
    else:
        return 7

02 >>> titanic['age2'] = titanic['age'].apply(category_age)
03 >>> titanic['sex'] = titanic['sex'].map({'male':1, 'female':0})
04 >>> titanic['family'] = titanic['sibsp'] + titanic['parch'] + 1
05 >>> titanic.to_csv('C:/Users/kmj/My_Python/7장_data/titanic3.csv', index =
    False)
06 >>> heatmap_data = titanic[['survived', 'sex', 'age2', 'family', 'pclass',
    'fare']]
07 >>> colormap = plt.cm.RdBu
08 >>> sns.heatmap(heatmap_data.astype(float).corr(), linewidths = 0.1, vmax
    = 1.0, square = True, cmap = colormap, linecolor = 'white', annot = True,
    annot_kws = {'size': 10})
<matplotlib.axes._subplots.AxesSubplot object at 0x000001DD4C8DBF88>
09 >>> plt.show()
```

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 결과 시각화

3. 변수 사이의 상관 계수를 히트맵으로 시각화하기

- [01~02행] age를 카테고리 값으로 바꾸어 age2 변수로 추가하기
 - 01행 10살 단위로 등급을 나누어 0~7의 값으로 바꿔주는 category_age 함수를 작성
 - 02행 category_age 함수를 적용하여 새로운 age2 열을 만들어 추가
- 03행 성별을 male/female에서 1/0으로 치환
- 04행 가족의 수를 구하여 family 열을 추가
- 05행 수정된 데이터프레임을 titanic3.csv로 저장

survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_tolive	alone	age2	family
0	3	1	22	1	0	7.25	S	Third	man	TRUE	C	Southamp	no	FALSE	2
1	1	1	38	1	0	71.2833	C	First	woman	FALSE	C	Cherbourg	yes	FALSE	3
1	3	0	26	0	0	7.925	S	Third	woman	FALSE	C	Southamp	yes	TRUE	2
1	1	0	35	1	0	53.1	S	First	woman	FALSE	C	Southamp	yes	FALSE	3
0	3	1	35	0	0	8.05	S	Third	man	TRUE	C	Southamp	no	TRUE	3
0	3	1	28	0	0	8.4583	C	Third	man	TRUE	C	Queenston	no	TRUE	2
0	1	1	54	0	0	51.8625	S	First	man	TRUE	E	Southamp	no	TRUE	5
0	3	1	2	3	1	21.075	S	Third	child	FALSE	C	Southamp	no	FALSE	0

그림 7-14 titanic3.csv 파일에서 치환된 내용과 추가된 내용 확인

02. [상관 분석 + 히트맵] 타이타닉호 생존을 분석하기

■ 결과 시각화

3. 변수 사이의 상관 계수를 히트맵으로 시각화하기

- [06~09행] 상관 분석 결과를 히트맵으로 나타내기
 - 06행 히트맵에 사용할 데이터를 추출
 - 07행 히트맵에 사용할 색상맵을 지정
 - 08행 corr() 함수로 구한 상관 계수로 히트맵을 생성
 - 09행 생성한 히트맵을 나타냄

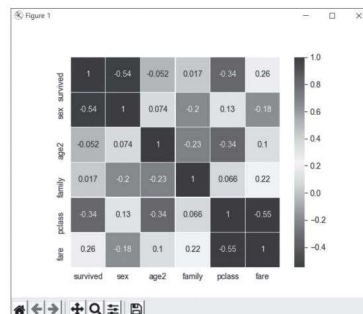


그림 7-15 상관 분석에 대한 히트맵 시각화