

▼ Extra2 データ分析

```
# 演習に必要なライブラリのインストール
!pip install matplotlib
!pip install pandas
!pip install scikit-learn
!pip install seaborn

import pprint
from matplotlib import pyplot as plt
%matplotlib inline
import pandas as pd
import seaborn as sns
from sklearn import datasets

Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.49.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.4)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas) (1.25.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.25.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.3.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.13.1)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /usr/local/lib/python3.10/dist-packages (from seaborn) (1.25.2)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn) (1.5.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /usr/local/lib/python3.10/dist-packages (from seaborn) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.49)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn) (2023.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn)
```

▼ 問題1 データ読み込み

今回の演習では UCI Machine Learning Repository が提供している Wine のデータセットを使用します。Ref このデータセットはイタリアの同じ地域で栽培され、3つの異なる品種に由来するワインの化学分析の結果です。

なおデータ定義は以下の通りです。

target

変数名	定義
target	ワインの種類

feature_names

変数名	定義
alcohol	アルコール濃度
malic_acid	リンゴ酸濃度
ash	灰分濃度
alcalinity_of_ash	灰のアルカリ性
magnesium	マグネシウム量

変数名	定義
total_phenol	フェノール量
flavanoids	フラバノイド量
nonflavanoid_phenols	非フラバノイドフェノール類の量
proanthocyanins	プロアントシアニン
color_intensity	色の強度
hue	色調
od280/OD315_of_diluted_wines	薄めたワインの280nmと315nmの波長の光に対する濁度
proline	プロリン量

ワインの味と成分についての[参考資料](#)

▼ Wine データセットの読み込み

Wine データセットは上で import した sklearn.datasets に格納されています。 sklearn.datasets からデータセットを読み込むには次のコードを実行します。[Ref](#)

```
wine_data = datasets.load_wine()
```

実際に読み込んだデータの中身を見ていきましょう。 Python の標準ライブラリである pprint モジュールを使うと、リスト型や辞書型などのオブジェクトを、整形してきれいに表示できます。[Ref](#)

```
pprint.pprint(wine_data)
```

```

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2]),

'target_names': array(['class_0', 'class_1', 'class_2'], dtype='|U7')}

```

Wine データセットの詳細、data、特徴量名などが辞書型で格納されていることが分かります。

▼ 問題1-1

上で読み込んだデータセット wine_data から data のみを抽出して、出力して下さい。(※ヒントは[こちら](#))

```

# ===== 以下に解答のコードを入力して下さい =====
data = wine_data.data
print(data)

[[1.423e+01 1.710e+00 2.430e+00 ... 1.040e+00 3.920e+00 1.065e+03]
 [1.320e+01 1.780e+00 2.140e+00 ... 1.050e+00 3.400e+00 1.050e+03]
 [1.316e+01 2.360e+00 2.670e+00 ... 1.030e+00 3.170e+00 1.185e+03]
 ...
 [1.327e+01 4.280e+00 2.260e+00 ... 5.900e-01 1.560e+00 8.350e+02]
 [1.317e+01 2.590e+00 2.370e+00 ... 6.000e-01 1.620e+00 8.400e+02]
 [1.413e+01 4.100e+00 2.740e+00 ... 6.100e-01 1.600e+00 5.600e+02]]

```

▼ 問題1-2

問題1-1 で読み込んだデータを pandas の dataframe に変換して下さい。その際変数名は wine_df、カラム名は wine_data.feature_names を使用して下さい。(※ヒントは[こちら](#))

```

# ===== 以下に解答のコードを入力して下さい =====
wine_df = pd.DataFrame(data, columns = wine_data.feature_names)
print(wine_df)

alcohol malic_acid ash alcalinity_of_ash magnesium total_phenols \
0    14.23      1.71   2.43          15.6     127.0       2.80
1    13.20      1.78   2.14          11.2     100.0       2.65
2    13.16      2.36   2.67          18.6     101.0       2.80
3    14.37      1.95   2.50          16.8     113.0       3.85
4    13.24      2.59   2.87          21.0     118.0       2.80
...
173   13.71      5.65   2.45          20.5      95.0       1.68
174   13.40      3.91   2.48          23.0     102.0       1.80
175   13.27      4.28   2.26          20.0     120.0       1.59
176   13.17      2.59   2.37          20.0     120.0       1.65
177   14.13      4.10   2.74          24.5      96.0       2.05

flavanoids nonflavanoid_phenols proanthocyanins color_intensity hue \
0        3.06            0.28          2.29      5.64  1.04
1        2.76            0.26          1.28      4.38  1.05
2        3.24            0.30          2.81      5.68  1.03
3        3.49            0.24          2.18      7.80  0.86
4        2.69            0.39          1.82      4.32  1.04
...
173     0.61            0.52          1.06      7.70  0.64
174     0.75            0.43          1.41      7.30  0.70
175     0.69            0.43          1.35     10.20  0.59
176     0.68            0.53          1.46      9.30  0.60
177     0.76            0.56          1.35      9.20  0.61

od280/od315_of_diluted_wines proline
0                  3.92    1065.0
1                  3.40    1050.0
2                  3.17    1185.0
3                  3.45    1480.0
4                  2.93    735.0
...
173                 1.74    740.0
174                 1.56    750.0
175                 1.56    835.0
176                 1.62    840.0
177                 1.60    560.0

[178 rows x 13 columns]

```

▼ 問題1-3

問題1-2 で変換した wine_df について、上から10行を Notebook 上に表示してみましょう。(※ヒントは[こちら](#))

===== 以下に解答のコードを入力して下さい ======
wine_df.head(10)

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	
5	14.20	1.76	2.45	15.2	112.0	3.27	3.39	
6	14.39	1.87	2.45	14.6	96.0	2.50	2.52	
7	14.06	2.15	2.61	17.6	121.0	2.60	2.51	
8	14.83	1.64	2.17	14.0	97.0	2.80	2.98	
9	13.86	1.35	2.27	16.0	98.0	2.98	3.15	

Next steps: [Generate code with wine_df](#) [View recommended plots](#)

▼ 問題1-4

wine_df の行数、列数を確認しましょう。(※ヒントは[こちら](#))

===== 以下に解答のコードを入力して下さい ======
wine_df.shape

(178, 13)

▼ 問題1-5

wine_df の列名の一覧を確認しましょう。(※ヒントは[こちら](#))

===== 以下に解答のコードを入力して下さい ======
wine_df.columns

```
Index(['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium',
       'total_phenols', 'flavanoids', 'nonflavanoid_phenols',
       'proanthocyanins', 'color_intensity', 'hue',
       'od280/od315_of_diluted_wines', 'proline'],
      dtype='object')
```

▼ 問題1-6

wine_df から、列 alcohol を取得してみましょう。

===== 以下に解答のコードを入力して下さい ======
alcohol_column = wine_df[['alcohol']]
alcohol_column.head(10)

	alcohol	grid
0	14.23	grid
1	13.20	
2	13.16	
3	14.37	
4	13.24	
5	14.20	
6	14.39	
7	14.06	
8	14.83	
9	13.86	

▼ 問題1-7

wine_df から、列 color_intensity, hue を取得してみましょう。

```
# ===== 以下に解答のコードを入力して下さい =====
selected_columns = wine_df[['color_intensity', 'hue']]
selected_columns.head(10)
```

	color_intensity	hue	grid
0	5.64	1.04	grid
1	4.38	1.05	
2	5.68	1.03	
3	7.80	0.86	
4	4.32	1.04	
5	6.75	1.05	
6	5.25	1.02	
7	5.05	1.06	
8	5.20	1.08	
9	7.22	1.01	

Next steps: [Generate code with selected_columns](#)

[View recommended plots](#)

▼ 問題2 データ操作

dataframe に対する操作を実践してみましょう。

▼ 問題2-1

wine_df から magnesium の値が 100 以上の行を抽出して、Notebook に表示してみましょう。 (※ヒントは[こちら](#))

また magnesium の値が 100 以上のものはいくつありますか？

```
# ===== 以下に解答のコードを入力して下さい =====
magnesium_over_100 = wine_df[wine_df['magnesium'] >= 100]
magnesium_over_100.head(len(magnesium_over_100))
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoic
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	
...
168	13.58	2.58	2.69	24.5	105.0	1.55	0.84	
169	13.40	4.60	2.86	25.0	112.0	1.98	0.96	
174	13.40	3.91	2.48	23.0	102.0	1.80	0.75	
175	13.27	4.28	2.26	20.0	120.0	1.59	0.69	
176	13.17	2.59	2.37	20.0	120.0	1.65	0.68	

31 rows × 13 columns

Next steps: [Generate code with magnesium_over_100](#) [View recommended plots](#)

```
# ===== 以下に解答のコードを入力して下さい =====
num_rows_over_100 = magnesium_over_100.shape[0]
# または num_rows_over_100 = len(magnesium_over_100)
print("Magnesiumの値が100以上の行の数:", num_rows_over_100)
```

Magnesiumの値が100以上の行の数: 81

▼ 問題2-2

wine_df から ash の値の降順で並べ替え、上から10行を Notebook に表示してみましょう。(※ヒントは[こちら](#))

```
# ===== 以下に解答のコードを入力して下さい =====
sorted_df = wine_df.sort_values(by='ash', ascending=False)
sorted_df.head(10)
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoic
121	11.56	2.05	3.23	28.5	119.0	3.18	5.08	
25	13.05	2.05	3.22	25.0	124.0	2.63	2.68	
112	11.76	2.68	2.92	20.0	103.0	1.75	2.03	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	
169	13.40	4.60	2.86	25.0	112.0	1.98	0.96	
36	13.28	1.64	2.84	15.5	110.0	2.60	2.68	
28	13.87	1.90	2.80	19.4	107.0	2.95	2.97	
127	11.79	2.13	2.78	28.5	92.0	2.13	2.24	
152	13.11	1.90	2.75	25.5	116.0	2.20	1.28	
177	14.13	4.10	2.74	24.5	96.0	2.05	0.76	

Next steps: [Generate code with sorted_df](#) [View recommended plots](#)

▼ 問題2-3

wine_df を target すなわちワインの種類によってグループに分け、flavanoids の平均値を計算しましょう。(※ヒントは[こちら](#))

またその結果、平均値が一番小さい target はどれですか？

```
# wine_df に新しい列 target を追加する
wine_df['target'] = pd.DataFrame(wine_data.target)

# ===== 以下に解答のコードを入力して下さい ======
average_flavanoids_by_target = wine_df.groupby('target')['flavanoids'].mean()
min_average_flavanoids_target = average_flavanoids_by_target.idxmin()

# ===== 以下に解答のコードを入力して下さい ======
print("平均値が一番小さい target:", min_average_flavanoids_target)

    平均値が一番小さい target: 2
```

▼ 問題2-4

wine_df の列 alcohol に対して、

13%以上は 1 13%未満は 0

と出力する関数 is_alcohol_high を適用し、新しく alcohol_high という変数を作成してみましょう。(※ヒントは[こちら](#))

作成したら、結果を下から10行出力して確認してみましょう。

```
# ===== 以下に解答のコードを入力して下さい ======
def is_alcohol_high(alcohol):
    if alcohol > 13:
        return 1
    else:
        return 0

# ===== 以下に解答のコードを入力して下さい ======
wine_df['alcohol_high'] = wine_df['alcohol'].apply(is_alcohol_high)
wine_df.head(len(wine_df['alcohol_high']))
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoids	proline
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	2.69	148
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	2.35	132
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	2.59	132
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	2.57	142
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	2.52	132
...
173	13.71	5.65	2.45	20.5	95.0	1.68	0.61	1.9	132
174	13.40	3.91	2.48	23.0	102.0	1.80	0.75	1.9	132
175	13.27	4.28	2.26	20.0	120.0	1.59	0.69	1.9	132
176	13.17	2.59	2.37	20.0	120.0	1.65	0.68	1.9	132
177	14.13	4.10	2.74	24.5	96.0	2.05	0.76	2.2	142

178 rows × 15 columns

Next steps: [Generate code with wine_df](#) [View recommended plots](#)

▼ 問題3 統計分析

作成したデータに対して統計分析をしてみましょう。

▼ 問題3-1

dataframe の describe メソッドを使用し、数値データの各種統計量を確認してみましょう。(※ヒントは[こちら](#))

特徴量の中で一番標準偏差が大きいものはどれですか？

```
# ===== 以下に解答のコードを入力して下さい ======
print(wine_df.describe())
```

```
alcohol malic_acid      ash alkalinity_of_ash magnesium \
count 178.000000 178.000000 178.000000    178.000000 178.000000
mean   13.000618  2.336348  2.366517     19.494944  99.741573
std    0.811827  1.117146  0.274344     3.339564  14.282484
min   11.030000  0.740000  1.360000    10.600000  70.000000
25%  12.362500  1.602500  2.210000    17.200000  88.000000
50%  13.050000  1.865000  2.360000    19.500000  98.000000
75%  13.677500  3.082500  2.557500    21.500000 107.000000
max   14.830000  5.800000  3.230000    30.000000 162.000000
```

```
total_phenols flavanoids nonflavonoid_phenols proanthocyanins \
count 178.000000 178.000000    178.000000 178.000000
mean   2.295112  2.029270     0.361854  1.590899
std    0.625851  0.998859     0.124453  0.572359
min   0.980000  0.340000     0.130000  0.410000
25%  1.742500  1.205000     0.270000  1.250000
50%  2.355000  2.135000     0.340000  1.555000
75%  2.800000  2.875000     0.437500  1.950000
max   3.880000  5.080000     0.660000  3.580000
```

```
color_intensity      hue od280/od315_of_diluted_wines      proline \
count 178.000000 178.000000    178.000000 178.000000
mean   5.058090  0.957449     2.611685  746.893258
std    2.318286  0.228572     0.709990  314.907474
min   1.280000  0.480000     1.270000  278.000000
25%  3.220000  0.782500     1.937500  500.500000
50%  4.690000  0.965000     2.780000  673.500000
75%  6.200000  1.120000     3.170000  985.000000
max   13.000000 1.710000     4.000000 1680.000000
```

```
target alcohol_high
count 178.000000 178.000000
mean   0.938202  0.516854
std    0.775035  0.501126
min   0.000000  0.000000
25%  0.000000  0.000000
50%  1.000000  1.000000
75%  2.000000  1.000000
max   2.000000 1.000000
```

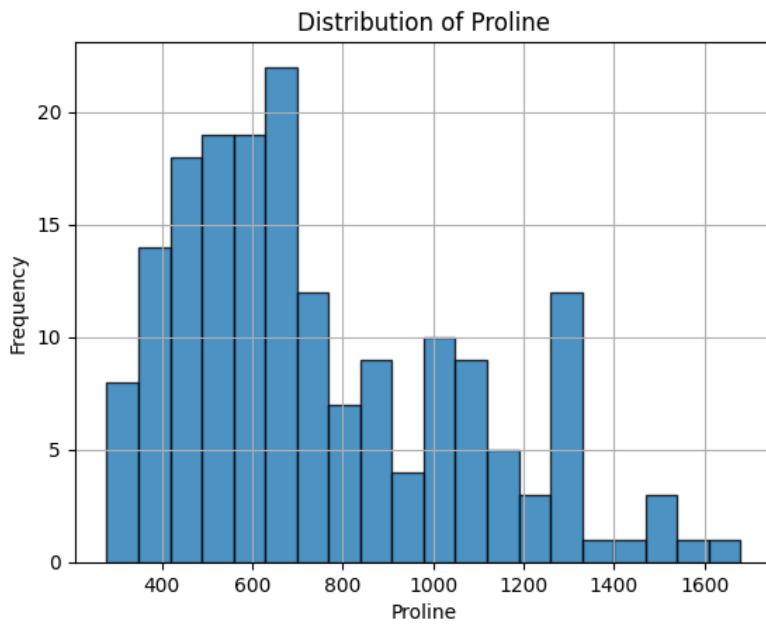
```
# ===== 以下に解答のコードを入力して下さい =====
std_max_column = wine_df.describe().loc['std'].idxmax()
print("標準偏差が最も大きい特徴量:", std_max_column)
```

標準偏差が最も大きい特徴量: proline

▼ 問題3-2

dataframeの plot.hist メソッドを使用し、proline の分布を可視化しましょう。(*ヒントは[こちら](#))

```
# ===== 以下に解答のコードを入力して下さい =====
wine_df['proline'].plot.hist(bins=20, edgecolor='black', alpha=0.8)
# 細かい設定
plt.title('Distribution of Proline')
plt.xlabel('Proline')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

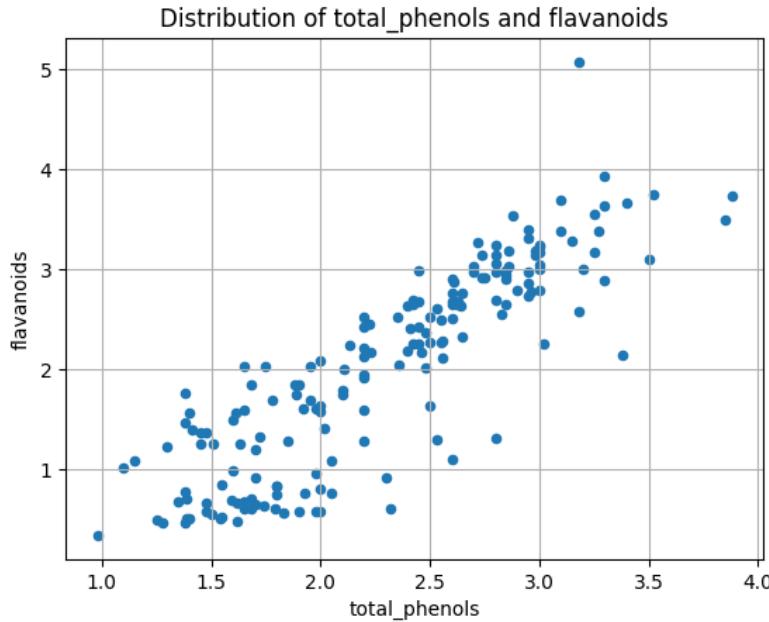


▼ 問題3-3

dataframeの `plot.scatter` メソッドを使用し、`total_phenols`, `flavanoids` の分布を可視化しましょう。(※ヒントは[こちら](#))

また可視化した結果、どんなことが言えそうですか？

```
# ===== 以下に解答のコードを入力して下さい =====
wine_df.plot.scatter(x="total_phenols", y="flavanoids")
plt.title("Distribution of total_phenols and flavanoids")
plt.grid(True)
```



```
# ===== 以下に解答のコードを入力して下さい =====
```

```
print("total_phenolsとflavanoidsには正の相関がある。つまり、どちらか一方が増えれば増えるほど、もう片方も増加する。")
```

total_phenolsとflavanoidsには正の相関がある。つまり、どちらか一方が増えれば増えるほど、もう片方も増加する。

▼ 問題3-4

dataframeの `corr` メソッドを使用し、`total_phenols`, `flavanoids` の相関係数を計算しましょう。(※ヒントは[こちら](#))

```
# ===== 以下に解答のコードを入力して下さい =====
corr_df = wine_df[['total_phenols','flavanoids']]
print(corr_df.corr())
print("相関係数が0.864564より、かなり強い正の相関がある。")
```

```
total_phenols  flavanoids
total_phenols    1.000000  0.864564
flavanoids      0.864564  1.000000
相関係数が0.864564より、かなり強い正の相関がある。
```

問題4 探索的データ解析 (EDA)

EDAを行い、データの特徴や傾向を特定ていきましょう。

▼ 問題4-1

ワインの種類 target ごとに特徴量の平均を算出してみましょう。またその結果、ワインの種類 target ごとに差がある特徴量を2つ選択して下さい。

```
# ===== 以下に解答のコードを入力して下さい =====
X = wine_data.data # 特徴量
y = wine_data.target # ターゲット

# ワインの種類(=target)ごとに特徴量の平均を計算
mean_by_target = wine_df.groupby('target').mean()

print("ワインの種類ごとの特徴量の平均:")
print(mean_by_target)

ワインの種類ごとの特徴量の平均:
alcohol  malic_acid    ash  alkalinity_of_ash  magnesium \
target
0       13.744746  2.010678  2.455593        17.037288 106.338983
1       12.278732  1.932676  2.244789        20.238028 94.549296
2       13.153750  3.333750  2.437083        21.416667 99.312500

total_phenols  flavanoids  nonflavanoid_phenols  proanthocyanins \
target
0           2.840169  2.982373          0.290000      1.899322
1           2.258873  2.080845          0.363662      1.630282
2           1.678750  0.781458          0.447500      1.153542

color_intensity    hue  od280/od315_of_diluted_wines    proline \
target
0           5.528305  1.062034          3.157797 1115.711864
1           3.086620  1.056282          2.785352 519.507042
2           7.396250  0.682708          1.683542 629.895833

alcohol_high
target
0       0.966102
1       0.112676
2       0.562500

# ===== 以下に解答のコードを入力して下さい =====
# 差がある特徴量を選択
feature_diffs = mean_by_target.diff().abs().max() # 差が最大の特徴量を選択
top_diff_indices = feature_diffs.nlargest(2).index # 最も差が大きい2つの特徴量を選択

print("\n差がある特徴量: " + top_diff_indices[0] + ", " + top_diff_indices[1])
```

差がある特徴量: proline, magnesium

▼ 問題4-2

問題4-1で選択した2つの特徴量で、上手くワインの種類が分類できそうかを散布図を用いて確認してみましょう。

ここでは seaborn の scatterplot を用いて描画してみましょう。(※ヒントは[こちら](#))

```
# ===== 以下に解答のコードを入力して下さい =====
plt.figure(figsize=(10, 6)) # グラフのサイズを調整
sns.scatterplot(x=top_diff_indices[0], y=top_diff_indices[1], hue='target', data=wine_df, palette='Set1', s=100) # マーク
plt.xlabel(top_diff_indices[0], fontsize=12) # x軸ラベルのフォントサイズを設定
```

```
plt.ylabel(top_diff_indices[1], fontsize=12) # y軸ラベルのフォントサイズを設定
plt.title('Scatter plot of two selected features', fontsize=14) # グラフタイトルのフォントサイズを設定
plt.legend(title='target', loc='upper right') # 凡例の位置を右上に設定
plt.grid(True) # グリッド線を表示
plt.tight_layout() # グラフがはみ出さないようにレイアウトを調整
plt.show()
```

