

機械学習 講義で用いるデータセット

管理工学科
篠沢佳久

講義で用いるデータセット

- scikit-learnで扱えるデータセット(7種類)

- 分類問題

- アイリス(Iris dataset)
- ワイン(Wine dataset)
- 手書き数字(Digits dataset)
- 乳がん(Breast cancer dataset)

多クラス分類問題

二クラス分類問題

- 回帰問題

- ボストンの住宅価格(Boston dataset)
- 糖尿病患者(Diabetes dataset)
- 生理学的特徴と運動能力の関係(Linnerrud dataset)

Iris dataset

Iris dataset

■ アヤメの分類問題

用途	クラス分類
データ数	150
特徴量	4
目的変数	3

クラス名	データ数
setosa	50
versicolor	50
virginica	50



Iris.py

```
import numpy as np
```

```
from sklearn import datasets
```

データセットを用いるために必要

```
from matplotlib import pyplot as plt
```

散布図の描画のために必要

```
# データのロード
```

```
iris = datasets.load_iris()
```

`datasets.load_iris()`
Irisデータセットのロード

```
# データの説明
```

```
print(iris.DESCR)
```

`DESCR`
Irisデータセットの説明

`iris.DESCR`
「iris」は変数名

```
# 特徴量(4次元)
```

```
feature_name = iris.feature_names
```

```
print( "¥n [ 特徴量 ]" )
```

```
print( feature_name )
```

`feature_names`
特徴量の名前

```
# 特徴量のデータ
```

```
data = iris.data
```

`data`
特徴量

大きさ
(150,4)

目的変数

```
class_name = iris.target_names  
print( "¥n [ クラス名 ]" )  
print( class_name )
```

target_names

目的変数(クラス)の名前

目的変数の値

```
label = iris.target
```

target

目的変数の値(0,1,2)

個数

150

クラスごとのデータ数

```
label_count = []  
for i in range( np.min(label) , np.max(label)+1 ):  
    label_count.append( len( label[label==i] ) )  
print( "¥n [ クラスごとのデータ数 ]" )  
print( label_count )
```

label==0

labelの要素が0の場合はTrue
その他はFalse

label[label==0]

labelの要素が0のみを取り出す

特徴量, クラス番号の表示

```
print( "¥n 特徴量      : クラス" )  
for i in range(len(label)):  
    print( data[i] , ":" , label[i] )
```

データ(特徴量, クラス番号)の表示

散布図の表示

```
fig = plt.figure(figsize=(10,8))
```

グラフの描画領域の大きさ

```
plt.subplots_adjust(wspace=0.4, hspace=0.6)
```

```
color = ['r', 'g', 'b']
```

```
count = 1
```

```
for i in range(4):
```

```
    for j in range(4):
```

```
        plt.subplot(4,4,count)
```

subplot(4,4,番号)

番号の欄にグラフを表示

```
total = 0
```

```
for k in range(3):
```

```
    x0 = [d[i] for d in data[total:total+label_count[k]]]
```

x軸のデータ

```
    x1 = [d[j] for d in data[total:total+label_count[k]]]
```

y軸のデータ

```
    plt.scatter(x0, x1, c=color[k], label=class_name[k])
```

```
    total += label_count[k]
```

```
    plt.xlabel(feature_name[i])
```

```
    plt.ylabel(feature_name[j])
```

scatter(xの値, yの値, c=色, label=凡例名)
plt.legend()で凡例が表示(コメントしています)

```
count += 1
```

xlabel(x軸の説明)
ylabel(y軸の説明)

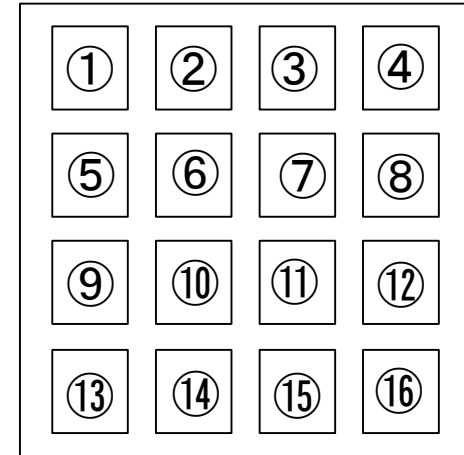
```
plt.suptitle("Iris")
```

```
plt.show()
```

suptitle(タイトル)
グラフのタイトル

show()
グラフの表示

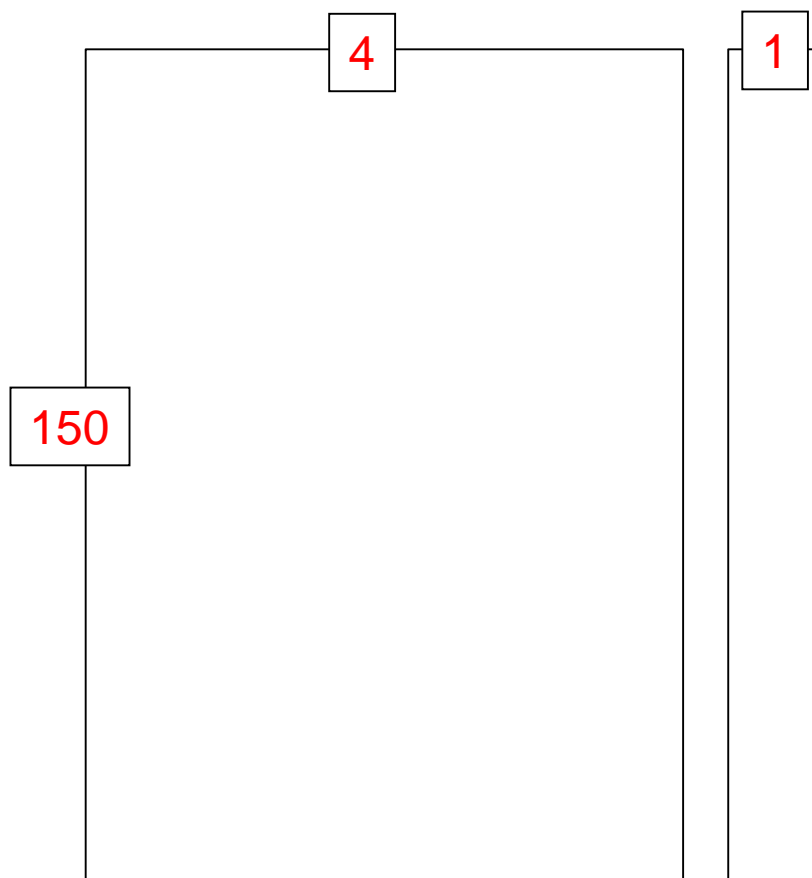
(4×4)個の散布図の表示



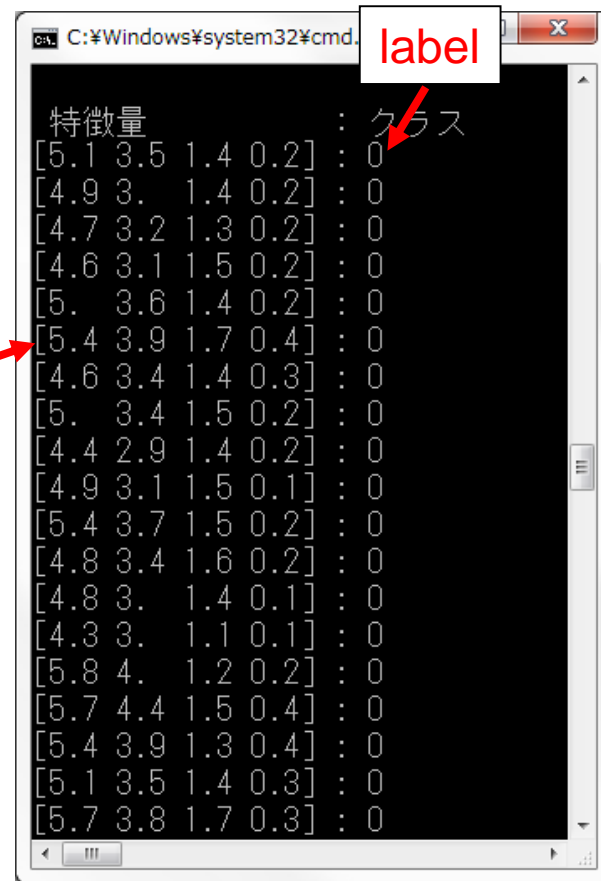
特徴量と教師ラベル

`data = iris.data`

`label = iris.target`



`data`



A screenshot of a Windows command prompt window showing the Iris dataset. The window title is 'C:\Windows\system32\cmd...'. The output displays the features (特徴量) and labels (教師ラベル) for 150 samples. A red arrow points from the 'data' label to the first column of features, and another red arrow points from the 'label' label to the second column of labels.

特徴量	教師ラベル
[5.1 3.5 1.4 0.2]	0
[4.9 3. 1.4 0.2]	0
[4.7 3.2 1.3 0.2]	0
[4.6 3.1 1.5 0.2]	0
[5. 3.6 1.4 0.2]	0
[5.4 3.9 1.7 0.4]	0
[4.6 3.4 1.4 0.3]	0
[5. 3.4 1.5 0.2]	0
[4.4 2.9 1.4 0.2]	0
[4.9 3.1 1.5 0.1]	0
[5.4 3.7 1.5 0.2]	0
[4.8 3.4 1.6 0.2]	0
[4.8 3. 1.4 0.1]	0
[4.3 3. 1.1 0.1]	0
[5.8 4. 1.2 0.2]	0
[5.7 4.4 1.5 0.4]	0
[5.4 3.9 1.3 0.4]	0
[5.1 3.5 1.4 0.3]	0
[5.7 3.8 1.7 0.3]	0

実行結果

```
C:\Windows\system32\cmd.exe
C:\home\shino\ML-2019\データセット\program>python Iris.py
Iris Plants Database
=====
Notes
-----
Data Set Characteristics:
: Number of Instances: 150 (50 in each of three classes)
: Number of Attributes: 4 numeric, predictive attributes and the class
: Attribute Information:
  - sepal length in cm
  - sepal width in cm
  - petal length in cm
  - petal width in cm
  - class:
    - Iris-Setosa
    - Iris-Versicolour
    - Iris-Virginica
: Summary Statistics:

=====
              Min  Max   Mean   SD   Class Correlation
=====
sepal length:  4.3  7.9   5.84   0.83    0.7826
sepal width:   2.0  4.4   3.05   0.43   -0.4194
petal length:   1.0  6.9   3.76   1.76    0.9490 (high!)
petal width:    0.1  2.5   1.20   0.76    0.9565 (high!)
=====
```

Irisデータセットの説明

特徴量の名前

```
[ 特徴量 ]  
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

目的変数(クラス)の名前

```
[ クラス名 ]  
['setosa', 'versicolor', 'virginica']
```

クラスごとのデータ数

```
[ クラスごとのデータ数 ]  
[50, 50, 50]
```

```
特徴量      : クラス
```

```
[5.1 3.5 1.4 0.2] : 0
```

```
[4.9 3.  1.4 0.2] : 0
```

```
[4.7 3.2 1.3 0.2] : 0
```

```
[4.6 3.1 1.5 0.2] : 0
```

```
[5.  3.6 1.4 0.2] : 0
```

```
[5.4 3.9 1.7 0.4] : 0
```

```
[4.6 3.4 1.4 0.3] : 0
```

```
[5.  3.4 1.5 0.2] : 0
```

```
[4.4 2.9 1.4 0.2] : 0
```

```
[4.9 3.1 1.5 0.1] : 0
```

```
[5.4 3.7 1.5 0.2] : 0
```

```
[4.8 3.4 1.6 0.2] : 0
```

```
[4.8 3.  1.4 0.1] : 0
```

```
[4.3 3.  1.1 0.1] : 0
```

```
[5.8 4.  1.2 0.2] : 0
```

```
[5.7 4.4 1.5 0.4] : 0
```

```
[5.4 3.9 1.3 0.4] : 0
```

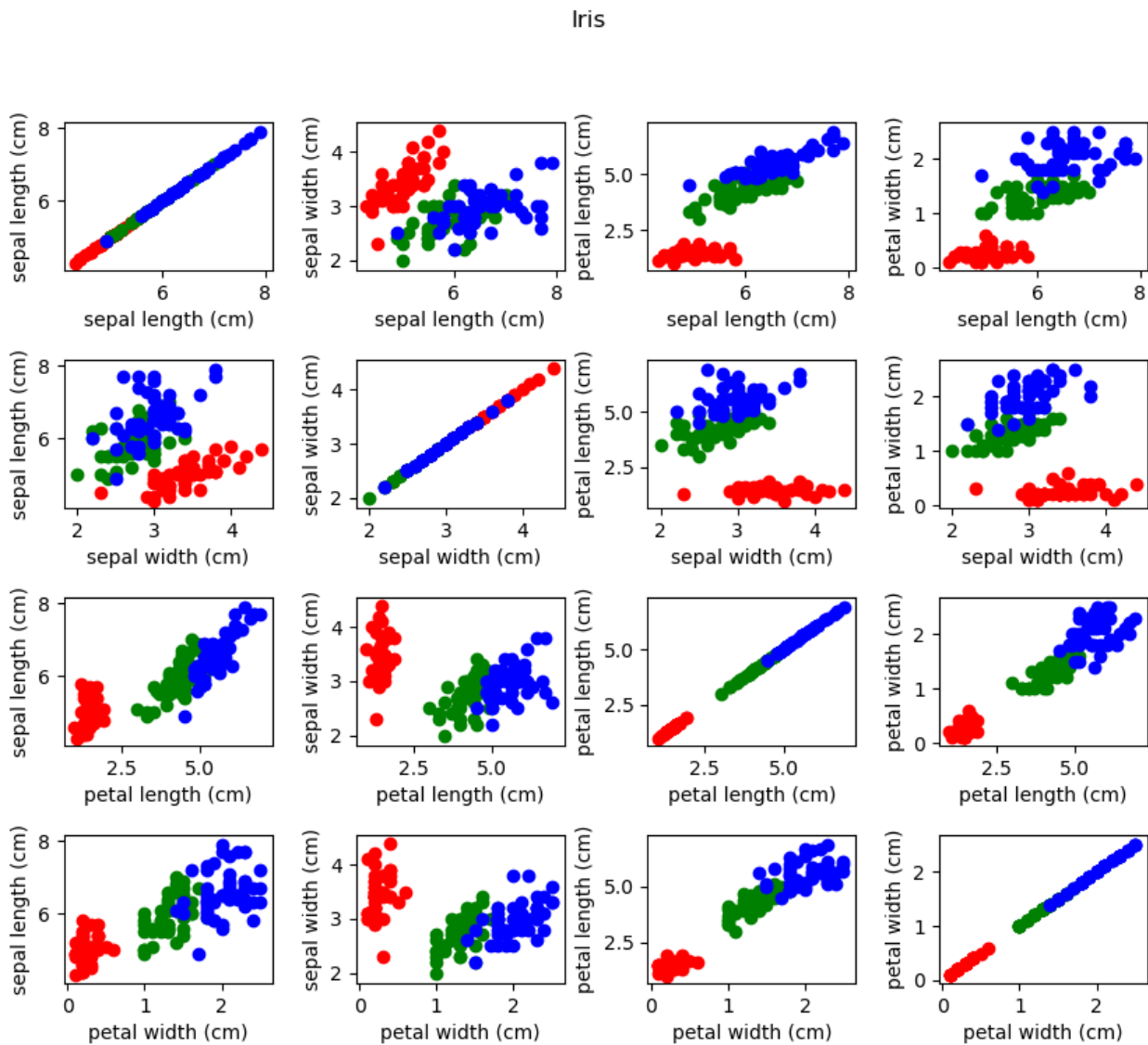
```
[5.1 3.5 1.4 0.3] : 0
```

特徴量の値
大きさ: (150,4)

クラス
0 → setosa
1 → versicolor
2 → virginica

個数: 150

散布図



Wine dataset

Wine dataset

■ ワインの分類問題

用途	クラス分類
データ数	178
特徴量	13
目的変数	3

クラス名	データ数
class_0	59
class_1	71
class_2	50



wine.py

```
import numpy as np
```

```
from sklearn import datasets
```

データセットを用いるために必要

```
from matplotlib import pyplot as plt
```

散布図の描画のために必要

```
# データのロード
```

```
wine = datasets.load_wine()
```

datasets.load_wine()

wineデータセットのロード

```
# データの説明
```

```
print(wine.DESCR)
```

DESCR

wineデータセットの説明

```
# 特徴量(13次元)
```

```
feature_name = wine.feature_names
```

feature_names

特徴量の名前

```
print( "¥n [ 特徴量 ]" )
```

```
print( feature_name )
```

```
# 特徴量のデータ
```

```
data = wine.data
```

data
特徴量

大きさ
(178,13)

目的変数

```
class_name = wine.target_names
```

target_names

目的変数(クラス)の名前

```
print( "¥n [ クラス名 ]" )
```

```
print( class_name )
```

目的変数の値

```
label = wine.target
```

target

目的変数の値(0,1,2)

個数

178

クラスごとのデータ数

```
label_count = []
```

各クラスのデータ数

```
for i in range( np.min(label) , np.max(label)+1 ):
```

```
    label_count.append( len( label[label==i] ) )
```

```
print( "¥n [ クラスごとのデータ数 ]" )
```

```
print( label_count )
```

特徴量, クラス番号の表示

```
print( "¥n 特徴量      : クラス" )
```

データ(特徴量, クラス番号)の表示

```
for i in range(len(label)):
```

```
    print( data[i] , ":" , label[i] )
```

散布図の表示

```
fig = plt.figure(figsize=(10,8))
```

グラフの描画領域の大きさ

```
plt.subplots_adjust(wspace=0.4, hspace=0.6)
```

```
color = ['r', 'g', 'b']
```

```
count = 1
```

```
for i in range(4):
```

```
    for j in range(4):
```

```
        plt.subplot(4,4,count)
```

subplot(4,4,番号)

番号の欄にグラフを表示

```
total = 0
```

```
for k in range(3):
```

```
    x0 = [d[i] for d in data[total:total+label_count[k]]]
```

x軸のデータ

```
    x1 = [d[j] for d in data[total:total+label_count[k]]]
```

y軸のデータ

```
    plt.scatter(x0, x1, c=color[k], label=class_name[k])
```

```
    total += label_count[k]
```

```
    plt.xlabel(feature_name[i])
```

```
    plt.ylabel(feature_name[j])
```

scatter(xの値, yの値, c=色, label=凡例名)
plt.legend()で凡例が表示(コメントしています)

```
count += 1
```

xlabel(x軸の説明)
ylabel(y軸の説明)

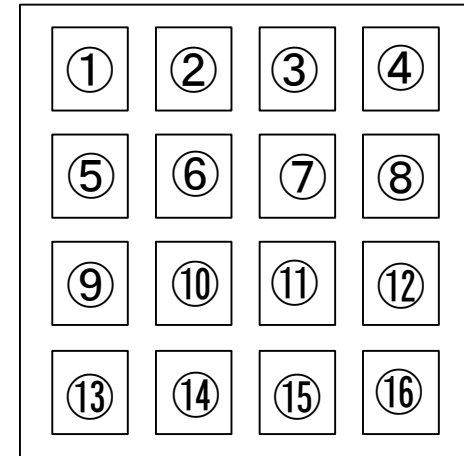
```
plt.suptitle("wine")
```

```
plt.show()
```

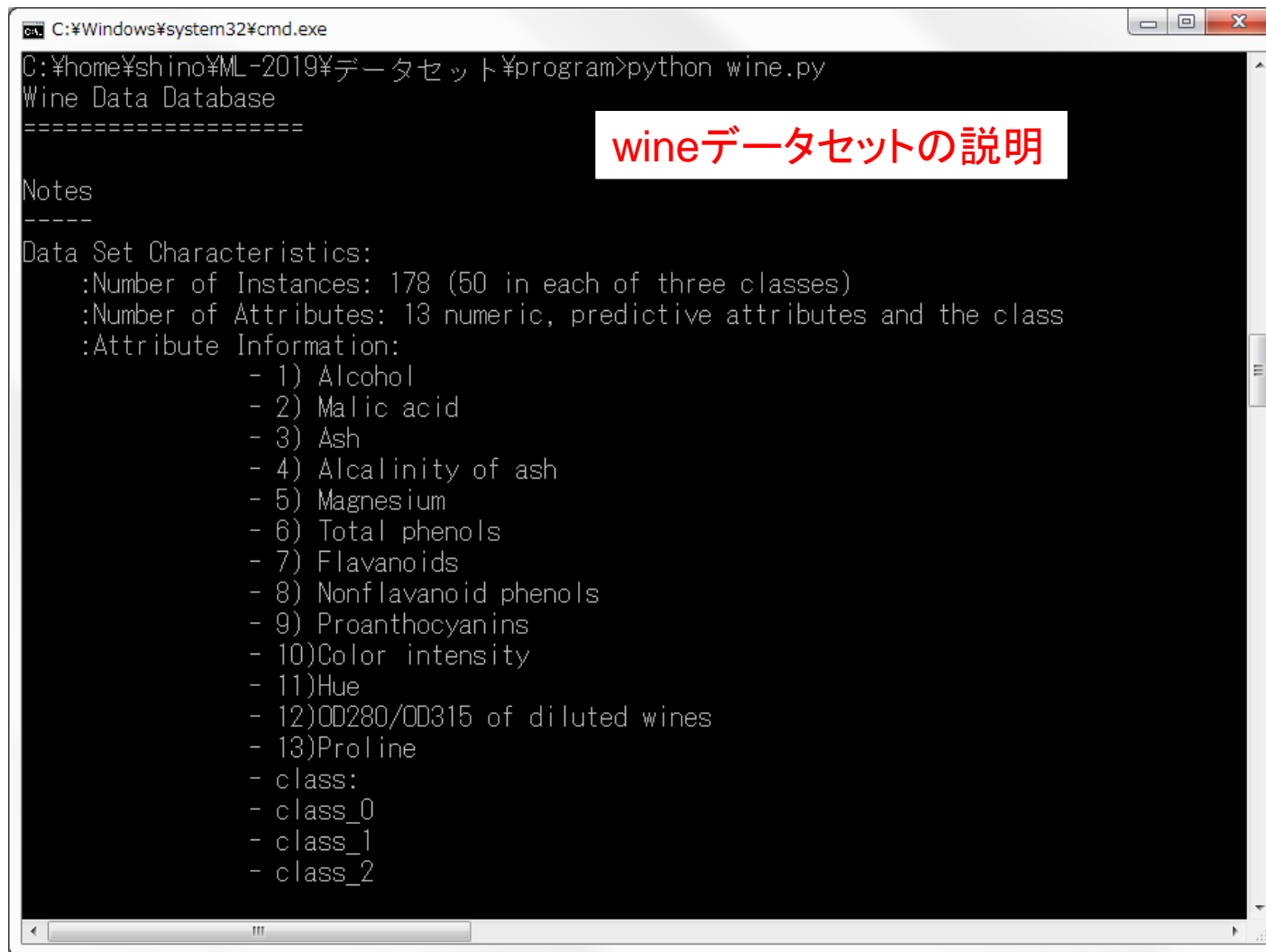
suptitle(タイトル)
グラフのタイトル

show()
グラフの表示

(4×4)個の散布図の表示



実行結果



```
C:\Windows\system32\cmd.exe
C:\home\shino\ML-2019\データセット\program>python wine.py
Wine Data Database
=====

Notes
-----

Data Set Characteristics:
: Number of Instances: 178 (50 in each of three classes)
: Number of Attributes: 13 numeric, predictive attributes and the class
: Attribute Information:
  - 1) Alcohol
  - 2) Malic acid
  - 3) Ash
  - 4) Alcalinity of ash
  - 5) Magnesium
  - 6) Total phenols
  - 7) Flavanoids
  - 8) Nonflavanoid phenols
  - 9) Proanthocyanins
  - 10) Color intensity
  - 11) Hue
  - 12) OD280/OD315 of diluted wines
  - 13) Proline
  - class:
  - class_0
  - class_1
  - class_2
```

特徴量の名前

```
[ 特徴量 ]
['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols', 'fl
```

目的変数(クラス)の名前

```
[ クラス名 ]
['class_0', 'class_1', 'class_2']
[59, 71, 48]
```

クラスごとのデータ数

特徴量の値
大きさ: (178,13)

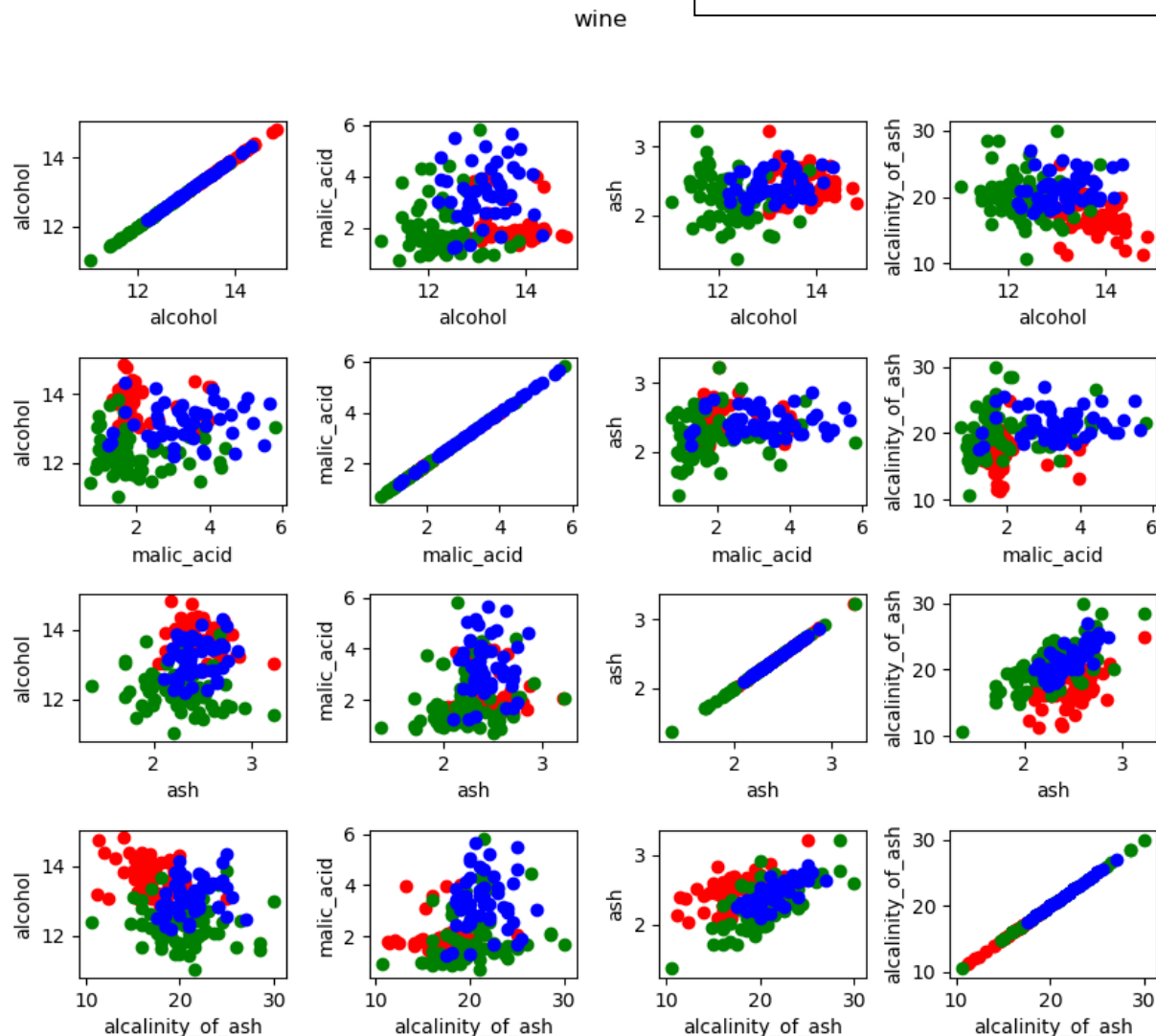
クラスの値
0 → class_0
1 → class_1
2 → class_2

個数: 178

```
特徴量      : クラス
[1.423e+01 1.710e+00 2.430e+00 1.560e+01 1.270e+02 2.800e+00 3.060e+00
 2.800e-01 2.290e+00 5.640e+00 1.040e+00 3.920e+00 1.065e+03] 0
[1.32e+01 1.78e+00 2.14e+00 1.12e+01 1.00e+02 2.65e+00 2.76e+00 2.60e-01
 1.28e+00 4.38e+00 1.05e+00 3.40e+00 1.05e+03] 0
[1.316e+01 2.360e+00 2.670e+00 1.860e+01 1.010e+02 2.800e+00 3.240e+00
 3.000e-01 2.810e+00 5.680e+00 1.030e+00 3.170e+00 1.185e+03] 0
[1.437e+01 1.950e+00 2.500e+00 1.680e+01 1.130e+02 3.850e+00 3.490e+00
 2.400e-01 2.180e+00 7.800e+00 8.600e-01 3.450e+00 1.480e+03] 0
[1.324e+01 2.590e+00 2.870e+00 2.100e+01 1.180e+02 2.800e+00 2.690e+00
 3.900e-01 1.820e+00 4.320e+00 1.040e+00 2.930e+00 7.350e+02] 0
[1.42e+01 1.76e+00 2.45e+00 1.52e+01 1.12e+02 3.27e+00 3.39e+00 3.40e-01
 1.97e+00 6.75e+00 1.05e+00 2.85e+00 1.45e+03] 0
[1.439e+01 1.870e+00 2.450e+00 1.460e+01 9.600e+01 2.500e+00 2.520e+00
 3.000e-01 1.980e+00 5.250e+00 1.020e+00 3.580e+00 1.290e+03] 0
[1.406e+01 2.150e+00 2.610e+00 1.760e+01 1.210e+02 2.600e+00 2.510e+00
 3.100e-01 1.250e+00 5.050e+00 1.060e+00 3.580e+00 1.295e+03] 0
[1.483e+01 1.640e+00 2.170e+00 1.400e+01 9.700e+01 2.800e+00 2.980e+00
 2.900e-01 1.980e+00 5.200e+00 1.080e+00 2.850e+00 1.045e+03] 0
[1.386e+01 1.350e+00 2.270e+00 1.600e+01 9.800e+01 2.980e+00 3.150e+00
```

散布図

特徴量は13次元
その中の4次元について表示



Digits dataset

Digits dataset

■ 手書き数字の分類問題

用途	クラス分類
データ数	1797
特徴量	画素数: $64(8 \times 8)$ 値: 0~16
目的変数	10

数字	データ数
0	178
1	182
2	177
3	183
4	181
5	182
6	181
7	179
8	174
9	180

digits.py

```
import random
```

```
import numpy as np
```

```
from sklearn import datasets
```

データセットを用いるために必要

```
from matplotlib import pyplot as plt
```

散布図の描画のために必要

```
# データのロード
```

```
digits = datasets.load_digits()
```

datasets.load_digits()

digitsデータセットのロード

```
# データの説明
```

```
print(digits.DESCR)
```

DESCR

digitsデータセットの説明

```
# 特徴量 (1797, 8, 8)
```

```
image = digits.images
```

images

1797枚, 画素数(8×8)

```
# 目的変数
```

```
label = digits.target
```

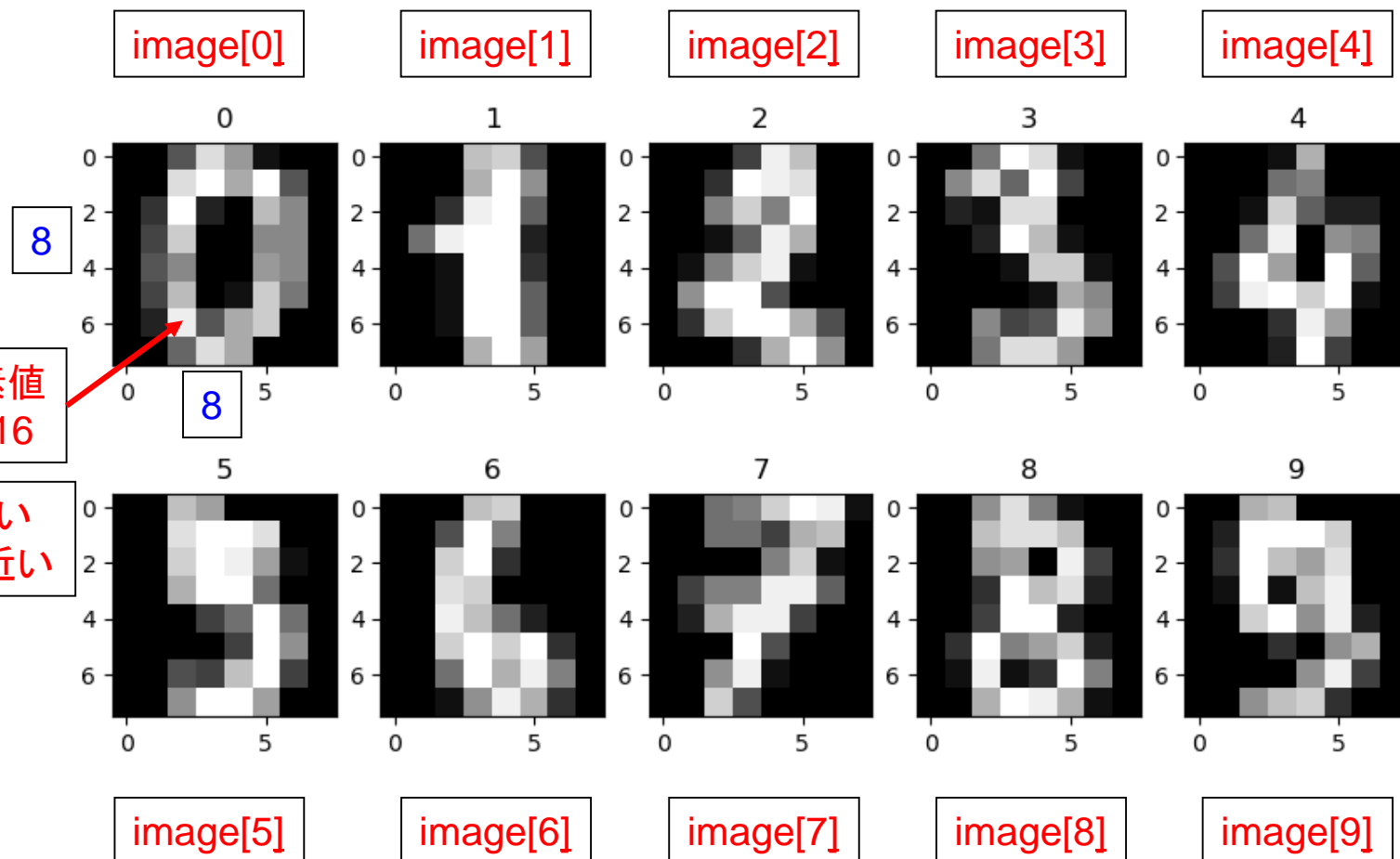
target

目的変数の値(0~9)

個数

1797

配列image



配列image

(8,8)の大きさの二次元配列が1797個(三次元配列)

数字ごとのデータ数

```
print( "¥n [ 数字ごとのデータ数 ]" )
```

数字ごとのデータ数

```
for i in range(10):
```

```
    print( i , ":" , np.sum(label==i) )
```

ave_image: 平均ベクトル
(8,8)の大きさの二次元配列を10個確保
要素は0で初期化

平均ベクトル (10,8,8)

```
ave_image = np.zeros((10,8,8))
```

数字の画像表示

乱数で文字を選択

```
fig1 = plt.figure(figsize=(10,5))
```

```
for i in range(10):
```

```
    rnd = random.randint(0,len(image))
```

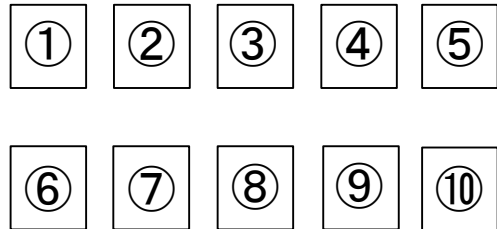
```
    plt.subplot(2, 5, i+1)
```

```
    plt.title('{0}'.format(label[rnd]))
```

```
    plt.imshow(image[rnd], cmap=plt.cm.gray, interpolation='none')
```

subplot(2,5,番号)

2×5の行列にグラフ(画像)を描画



imshow(二次元配列)
二次元配列を視覚化

cmap=plt.cm.gray
グレースケールで表示

interpolation='none'
補間はしない

平均ベクトルの画像表示

```
fig2 = plt.figure(figsize=(10,5))
```

```
for i in range(10):
```

数字ごとに平均ベクトルを計算

```
    ave_image[i] = image[label==i].mean(axis=0)
```

```
    plt.subplot(2, 5, i+1)
```

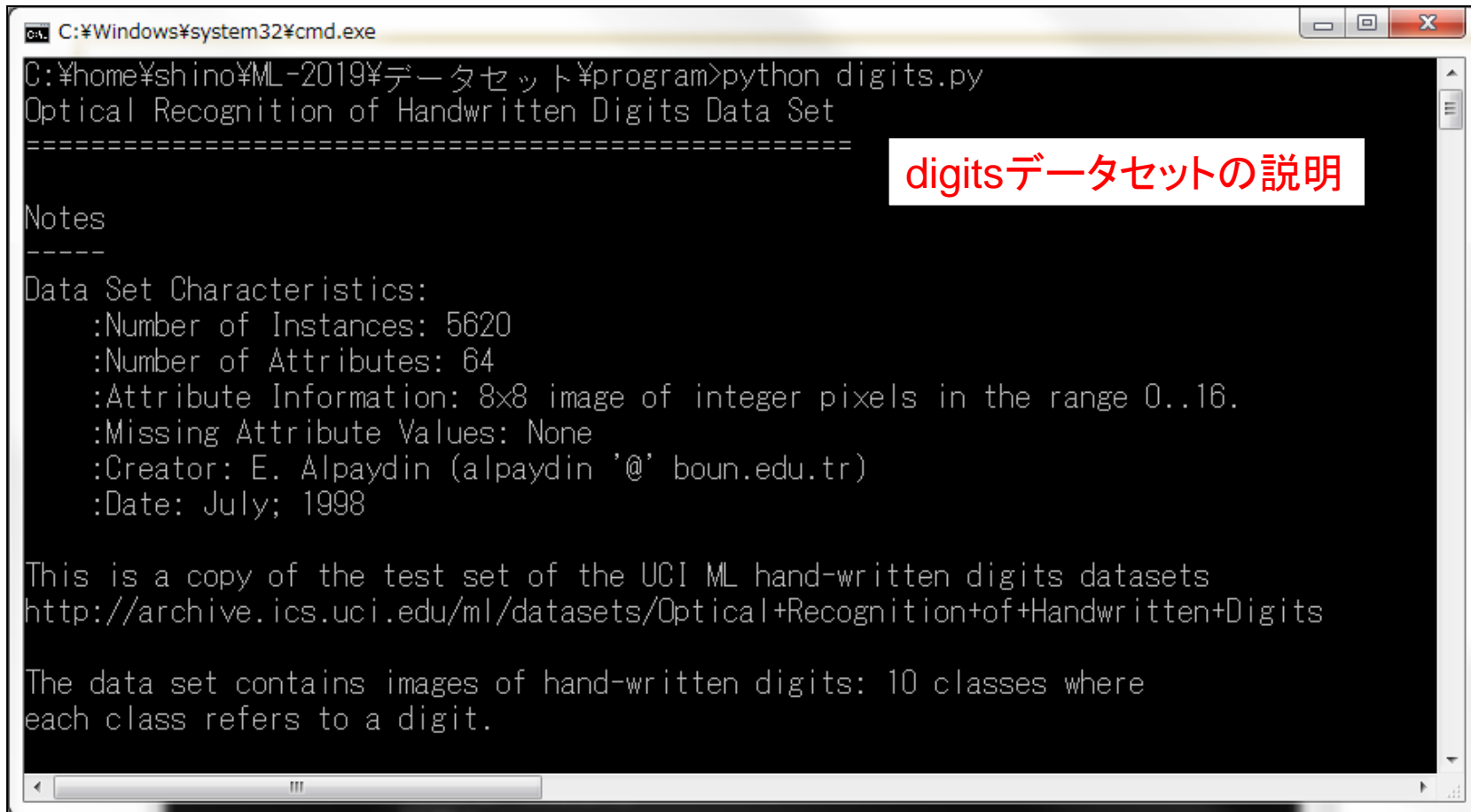
```
    plt.title('{0} ({1})'.format(i, np.sum(label==i)))
```

```
    plt.imshow(ave_image[i], cmap=plt.cm.gray, interpolation='none')
```

```
plt.show()
```

平均ベクトルを視覚化

実行結果①



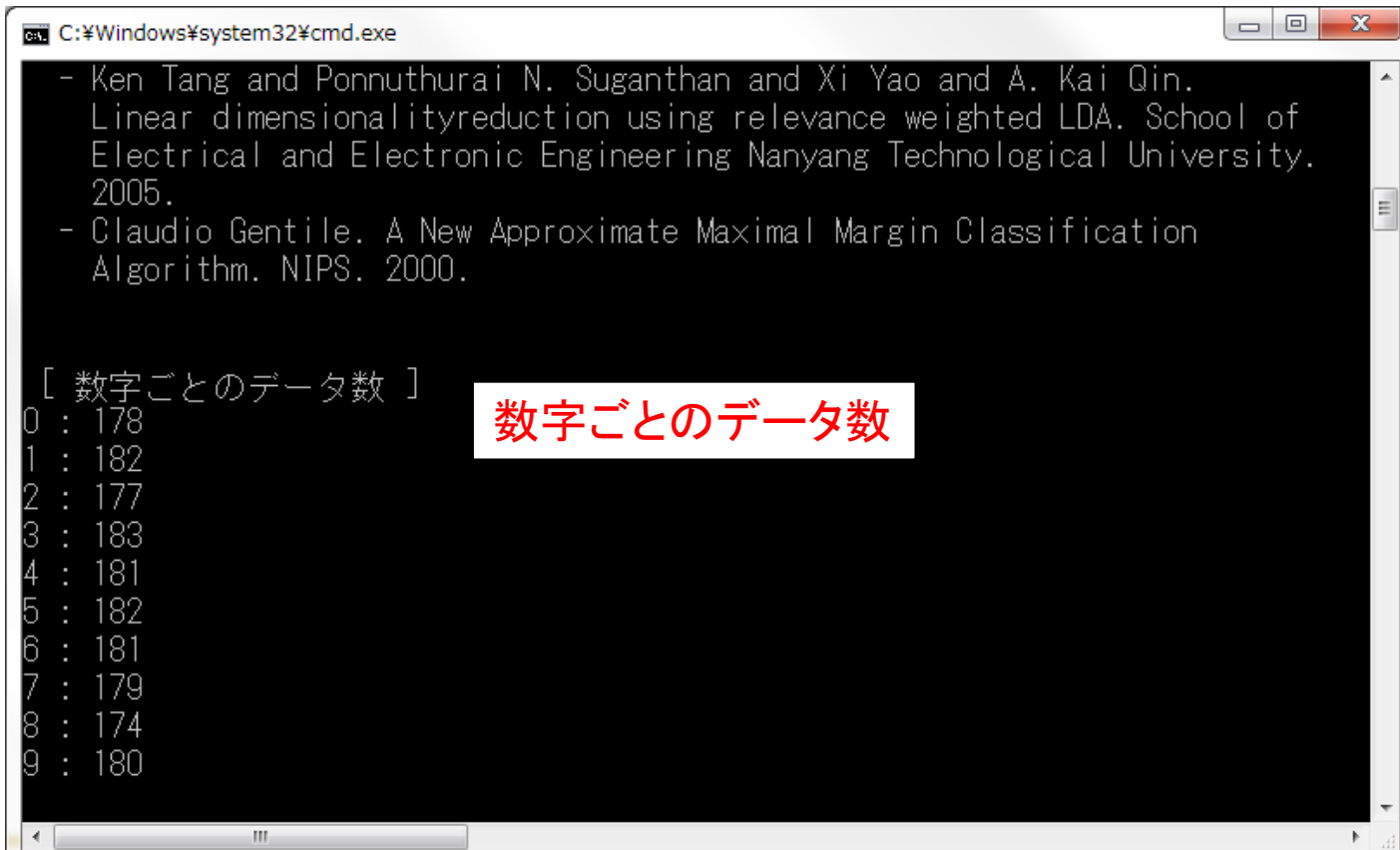
```
C:\Windows\system32\cmd.exe
C:\home\shino\ML-2019\データセット\program>python digits.py
Optical Recognition of Handwritten Digits Data Set
=====
Notes
-----
Data Set Characteristics:
: Number of Instances: 5620
: Number of Attributes: 64
: Attribute Information: 8x8 image of integer pixels in the range 0..16.
: Missing Attribute Values: None
: Creator: E. Alpaydin (alpaydin '@' boun.edu.tr)
: Date: July; 1998

This is a copy of the test set of the UCI ML hand-written digits datasets
http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

The data set contains images of hand-written digits: 10 classes where
each class refers to a digit.
```

digitsデータセットの説明

実行結果②



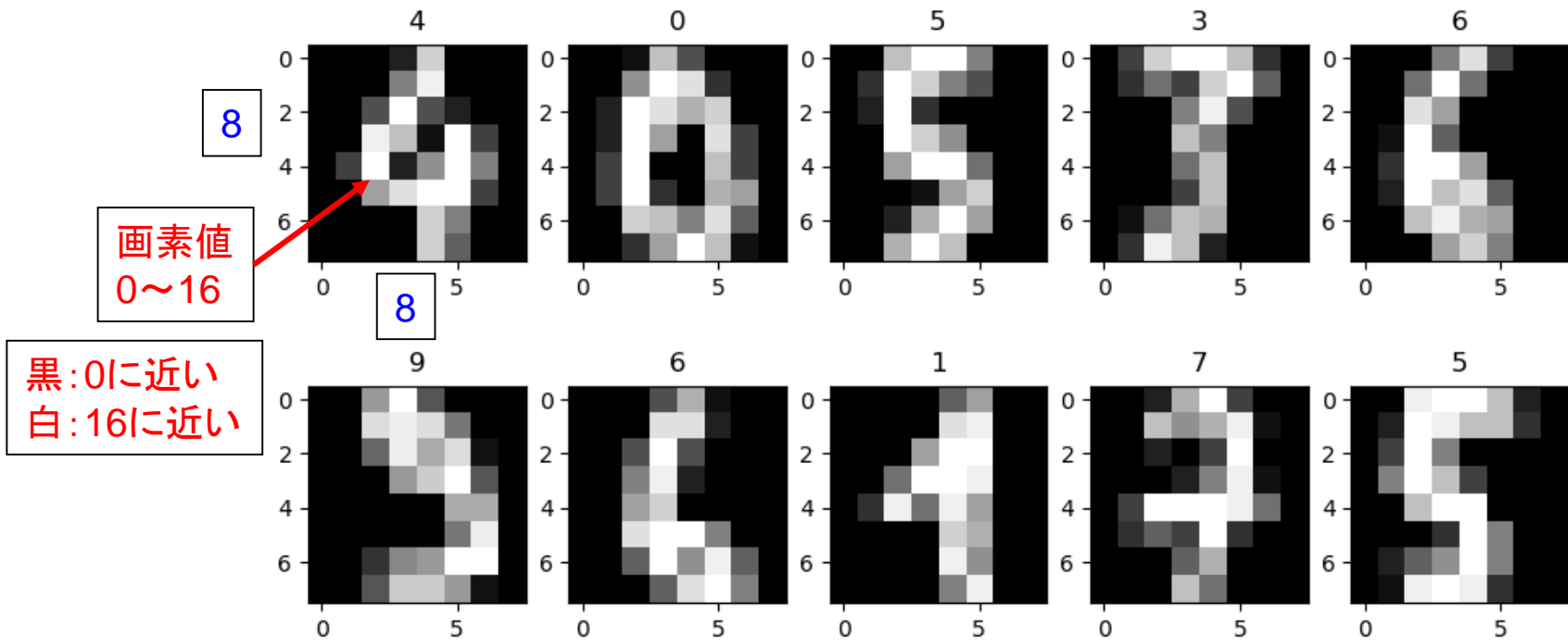
A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window contains two sections of text. The first section is a list of references, each preceded by a hyphen. The second section is a table showing the number of data points for each digit from 0 to 9. A red rectangular box with white text is overlaid on the table, highlighting the header and the first few rows.

```
- Ken Tang and Ponnuthurai N. Suganthan and Xi Yao and A. Kai Qin.  
  Linear dimensionality reduction using relevance weighted LDA. School of  
  Electrical and Electronic Engineering Nanyang Technological University.  
  2005.  
- Claudio Gentile. A New Approximate Maximal Margin Classification  
  Algorithm. NIPS. 2000.
```

[数字ごとのデータ数]

0	: 178
1	: 182
2	: 177
3	: 183
4	: 181
5	: 182
6	: 181
7	: 179
8	: 174
9	: 180

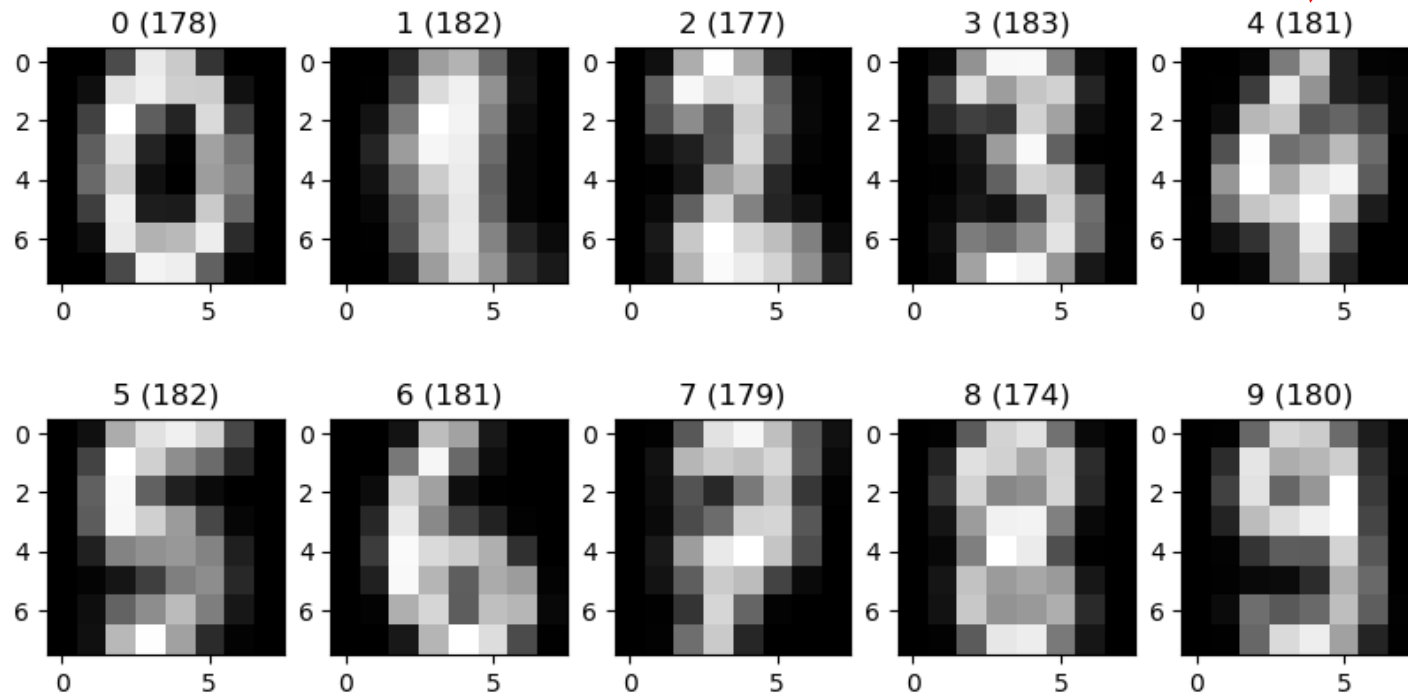
文字データの表示①



文字データの表示②

数字ごとの平均ベクトルの表示

数字(データ数)



Breast cancer dataset

Breast cancer dataset

- 乳がんの分類問題
 - 二値分類問題

用途	クラス分類
データ数	569
特徴量	30
目的変数	2

クラス名	データ数
malignant	212
benign	357

cancer.py

```
import numpy as np
```

```
from sklearn import datasets
```

```
from matplotlib import pyplot as plt
```

データセットを用いるために必要

散布図の描画のために必要

```
# データのロード
```

```
breast_cancer = datasets.load_breast_cancer()
```

`datasets.load_breast_cancer()`

cancerデータセットのロード

```
# データの説明
```

```
print(breast_cancer.DESCR)
```

`DESCR`

cancerデータセットの説明

```
# 特徴量(30次元)
```

```
feature_name = breast_cancer.feature_names
```

```
print( "¥n [ 特徴量 ]" )
```

```
print( feature_name )
```

`feature_names`

特徴量の名前

```
# 特徴量のデータ
```

```
data = breast_cancer.data
```

`data`

特徴量

大きさ

(569,30)

目的変数

```
class_name = breast_cancer.target_names
```

target_names

目的変数(クラス)の名前

```
print( "¥n [ クラス名 ]" )
```

```
print( class_name )
```

目的変数の値

```
label = breast_cancer.target
```

target

目的変数の値(0,1)

個数

569

クラスごとのデータ数

```
label_count = []
```

各クラスのデータ数

```
for i in range( np.min(label) , np.max(label)+1 ):
```

```
    label_count.append( len( label[label==i] ) )
```

```
print( "¥n [ クラスごとのデータ数 ]" )
```

```
print( label_count )
```

特徴量, クラス番号の表示

```
print( "¥n 特徴量      : クラス" )
```

データ(特徴量, クラス番号)の表示

```
for i in range(len(label)):
```

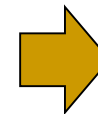
```
    print( data[i] , ":" , label[i] )
```

散布図の表示(データのソート)

```
for i in range(len(label)-1):  
    for j in range(len(label)-1,i,-1):  
        if label[j] < label[j-1]:  
            work = label[j]  
            label[j] = label[j-1]  
            label[j-1] = work  
  
            work1 = data[j].copy()  
            data[j] = data[j-1].copy()  
            data[j-1] = work1.copy()
```

ソート前のデータ

data	label
	0
	1
	.
	.
	.
	1
	0
	0
	.
	.
	.
	1



ソート後のデータ

data	label
	0
	0
	.
	.
	.
	0
	1
	1
	.
	.
	.
	1

散布図の表示

```
fig = plt.figure(figsize=(15,15))
```

グラフの描画領域の大きさ

```
plt.subplots_adjust(wspace=0.4, hspace=0.6)
```

(5×5)個の散布図の表示

```
color = [ 'r' , 'g' ]
```

```
count = 1
```

```
for i in range(5):
```

```
    for j in range(5):
```

```
        plt.subplot(5,5,count)
```

subplot(5,5,番号)

番号の欄にグラフを表示

```
total = 0
```

```
for k in range(5):
```

```
    x0 = [d[i] for d in data[total:total+label_count[k]]]
```

x軸のデータ

```
    x1 = [d[j] for d in data[total:total+label_count[k]]]
```

y軸のデータ

```
    plt.scatter(x0, x1, c=color[k], label=class_name[k])
```

```
    total += label_count[k]
```

```
    plt.xlabel(feature_name[i])
```

```
    plt.ylabel(feature_name[j])
```

```
count += 1
```

```
plt.suptitle("breast_cancer")
```

```
plt.show()
```

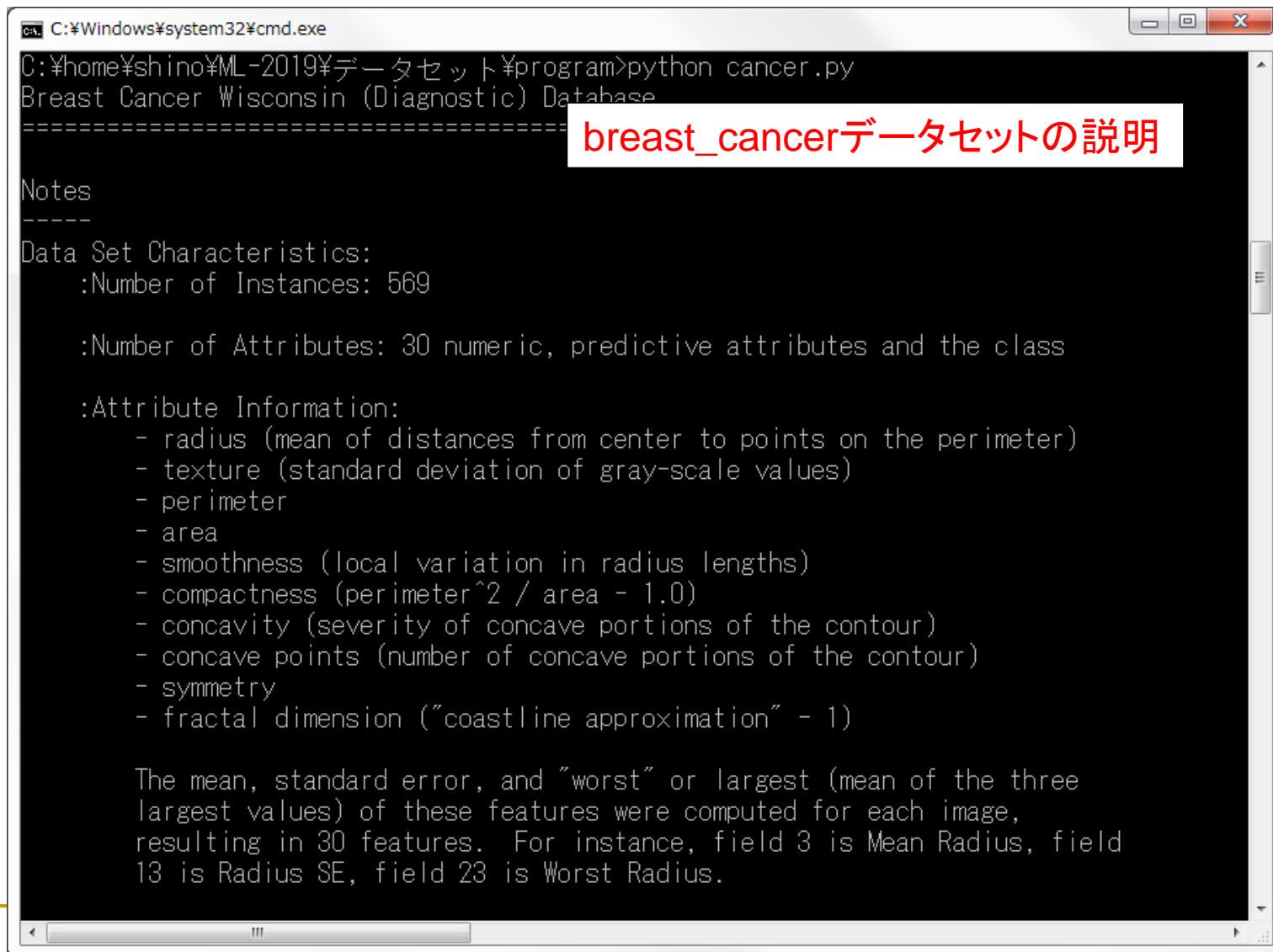
scatter(xの値, yの値, c=色, label=凡例名)
plt.legend()で凡例が表示(コメントしています)

xlabel(x軸の説明)
ylabel(y軸の説明)

suptitle(タイトル)
グラフのタイトル

show()
グラフの表示

実行結果



```
C:\Windows\system32\cmd.exe
C:\home\shino\ML-2019\データセット\program>python cancer.py
Breast Cancer Wisconsin (Diagnostic) Database
===== breast_cancerデータセットの説明 =====
Notes
-----
Data Set Characteristics:
  :Number of Instances: 569

  :Number of Attributes: 30 numeric, predictive attributes and the class

  :Attribute Information:
    - radius (mean of distances from center to points on the perimeter)
    - texture (standard deviation of gray-scale values)
    - perimeter
    - area
    - smoothness (local variation in radius lengths)
    - compactness (perimeter^2 / area - 1.0)
    - concavity (severity of concave portions of the contour)
    - concave points (number of concave portions of the contour)
    - symmetry
    - fractal dimension ("coastline approximation" - 1)

The mean, standard error, and "worst" or largest (mean of the three
largest values) of these features were computed for each image,
resulting in 30 features. For instance, field 3 is Mean Radius, field
13 is Radius SE, field 23 is Worst Radius.
```

```
C:\Windows\system32\cmd.exe

[ 特徴量 ]
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
'mean smoothness' 'mean compactness' 'mean concavity'
'mean concave points' 'mean symmetry' 'mean fractal dimension'
'radius error' 'texture error' 'perimeter error' 'area error'
'smoothness error' 'compactness error' 'concavity error'
'concave points error' 'symmetry error' 'fractal dimension error'
'worst radius' 'worst texture' 'worst perimeter' 'worst area'
'worst smoothness' 'worst compactness' 'worst concavity'
'worst concave points' 'worst symmetry' 'worst fractal dimension']

[ クラス名 ]
['malignant' 'benign']

[ クラスごとのデータ数 ]
[212, 357]

特徴量          : クラス
[1.799e+01 1.038e+01 1.228e+02 1.001e+03 1.184e-01 2.776e-01 3.001e-01
1.471e-01 2.419e-01 7.871e-02 1.095e+00 9.053e-01 8.589e+00 1.534e+02
6.399e-03 4.904e-02 5.373e-02 1.587e-02 3.003e-02 6.198e-03 2.538e+01
1.733e+01 1.846e+02 2.019e+03 1.622e-01 8.656e-01 7.119e-01 2.654e-01
4.601e-01 1.189e-01] : 0
[2.057e+01 1.777e+01 1.329e+02 1.326e+03 8.474e-02 7.864e-02 8.690e-02
7.017e-02 1.812e-01 5.667e-02 5.435e-01 7.339e-01 3.398e+00 7.408e+01
5.225e-03 1.308e-02 1.860e-02 1.340e-02 1.389e-02 3.532e-03 2.499e+01
2.341e+01 1.588e+02 1.956e+03 1.238e-01 1.866e-01 2.416e-01 1.860e-01
2.750e-01 8.902e-02] : 0
```

特徴量の名前

目的変数(クラス)の名前

特徴量の値
大きさ: (569,30)

クラスごとのデータ数

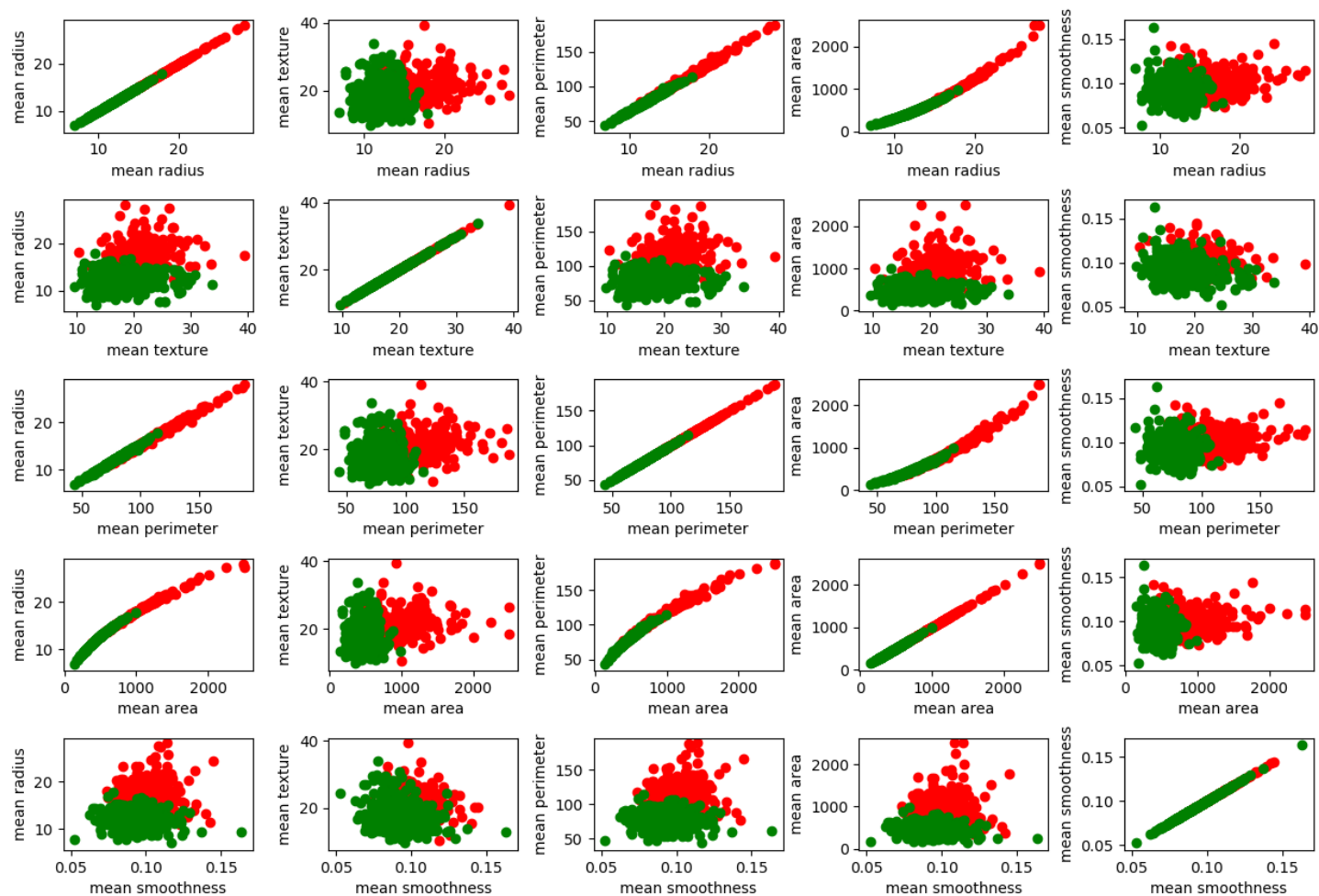
クラス
0 → malignant
1 → benign

個数: 569

散布図

特徴量は30次元
その中の5次元について表示

● malignant
● benign



Boston dataset

Boston dataset

■ ボストンの住宅価格の回帰問題

用途	回帰
データ数	506
特徴量	13
目的変数	1



Boston.py

```
import numpy as np
```

```
from sklearn import datasets
```

```
from matplotlib import pyplot as plt
```

データセットを用いるために必要

散布図の描画のために必要

```
# データのロード
```

```
boston = datasets.load_boston()
```

datasets.load_boston()

bostonデータセットのロード

```
# データの説明
```

```
print(boston.DESCR)
```

DESCR

bostonデータセットの説明

```
# 特徴量(13次元)
```

```
feature_name = boston.feature_names
```

```
print( "¥n [ 特徴量 ]" )
```

```
print( feature_name )
```

feature_names

特徴量の名前

```
# 特徴量のデータ
```

```
data = boston.data
```

data

特徴量

大きさ

(506,13)

目的変数の値

label = boston.target

target

目的変数の値

個数

509

特徴量, 目的値の表示

print("¥n 特徴量 : 目的値")

for i in range(len(label)):

print(data[i] , ":" , label[i])

データ(特徴量, 目的値)の表示

散布図の表示

グラフの描画領域の大きさ

fig = plt.figure(figsize=(10,8))

plt.subplots_adjust(wspace=0.4, hspace=0.6)

count = 1

for i in range(4):

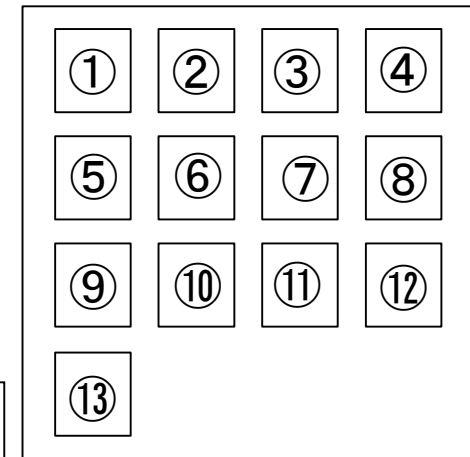
for j in range(4):

plt.subplot(4,4,count)

13個の散布図の表示

subplot(4,4,番号)

番号の欄にグラフを表示



```
x0 = data[:,i*4+j]
x1 = label
plt.scatter(x0, x1, color='r')
plt.xlabel(feature_name[i*4+j])
plt.ylabel("Price")

if count >= 13:
    break
count+=1

plt.suptitle('Boston')
plt.show()
```

x軸のデータ

y軸のデータ

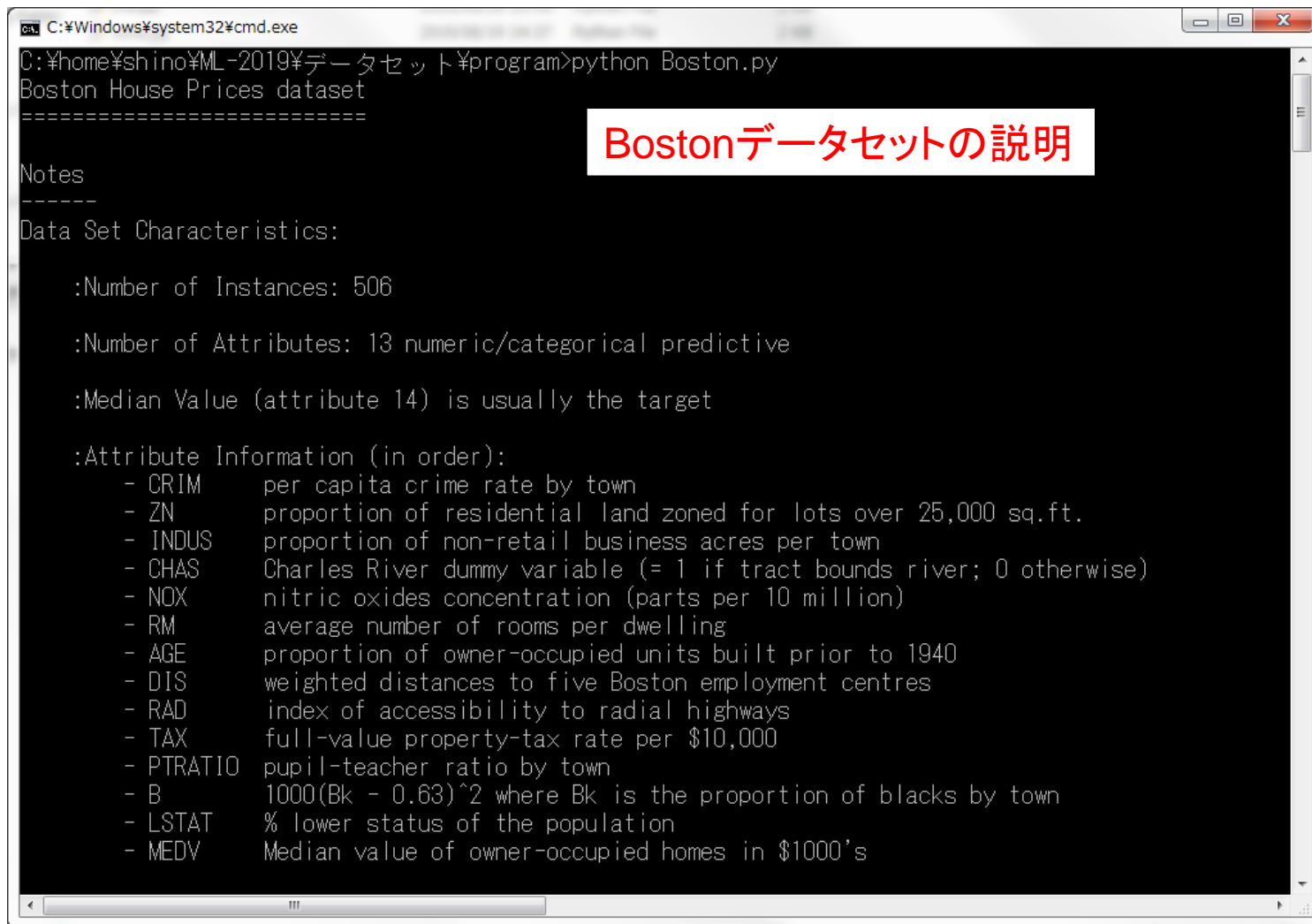
scatter(xの値, yの値, c=色)

xlabel(x軸の説明)
ylabel(y軸の説明)

suptitle(タイトル)
グラフのタイトル

show()
グラフの表示

実行結果



```
C:\Windows\system32\cmd.exe
C:\home\shino\ML-2019\データセット\program>python Boston.py
Boston House Prices dataset
=====
Notes
-----
Data Set Characteristics:

: Number of Instances: 506

: Number of Attributes: 13 numeric/categorical predictive

: Median Value (attribute 14) is usually the target

: Attribute Information (in order):
  - 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
```

Bostonデータセットの説明

特徴量の名前

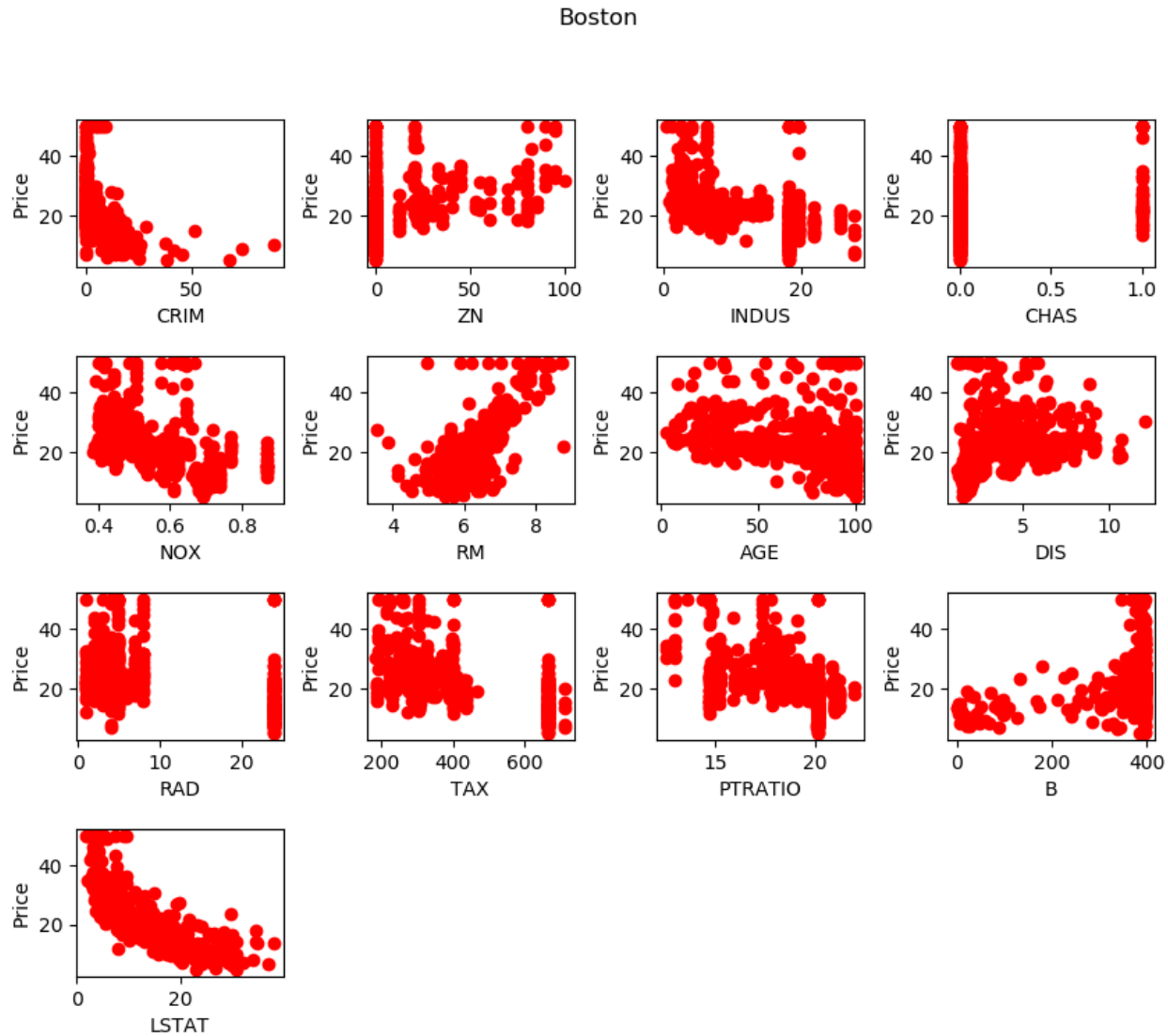
```
C:\Windows\system32\cmd.exe  
[ 特徴量 ]  
['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX', 'PTRATIO',  
'B', 'LSTAT']
```

```
特徴量      : 目的値  
[6.320e-03 1.800e+01 2.310e+00 0.000e+00 5.380e-01 6.575e+00 6.520e+01  
4.090e+00 1.000e+00 2.960e+02 1.530e+01 3.969e+02 4.980e+00] 24.0  
[2.7310e-02 0.0000e+00 7.0700e+00 0.0000e+00 4.6900e-01 6.4210e+00  
7.8900e+01 4.9671e+00 2.0000e+00 2.4200e+02 1.7800e+01 3.9690e+02  
9.1400e+00] 21.6  
[2.7290e-02 0.0000e+00 7.0700e+00 0.0000e+00 4.6900e-01 7.1850e+00  
6.1100e+01 4.9671e+00 2.0000e+00 2.4200e+02 1.7800e+01 3.9283e+02  
4.0300e+00] 34.7  
[3.2370e-02 0.0000e+00 2.1800e+00 0.0000e+00 4.5800e-01 6.9980e+00  
4.5800e+01 6.0622e+00 3.0000e+00 2.2200e+02 1.8700e+01 3.9463e+02  
2.9400e+00] 33.4  
[6.9050e-02 0.0000e+00 2.1800e+00 0.0000e+00 4.5800e-01 7.1470e+00  
5.4200e+01 6.0622e+00 3.0000e+00 2.2200e+02 1.8700e+01 3.9690e+02  
5.3300e+00] 36.2  
[2.9850e-02 0.0000e+00 2.1800e+00 0.0000e+00 4.5800e-01 6.4300e+00  
5.8700e+01 6.0622e+00 3.0000e+00 2.2200e+02 1.8700e+01 3.9412e+02  
5.2100e+00] 28.7  
[8.8290e-02 1.2500e+01 7.8700e+00 0.0000e+00 5.2400e-01 6.0120e+00  
6.6600e+01 5.5605e+00 5.0000e+00 3.1100e+02 1.5200e+01 3.9560e+02  
1.2430e+01] 22.9  
[1.4455e-01 1.2500e+01 7.8700e+00 0.0000e+00 5.2400e-01 6.1720e+00  
9.6100e+01 5.9505e+00 5.0000e+00 3.1100e+02 1.5200e+01 3.9690e+02  
1.9150e+01] 27.1  
[2.1124e-01 1.2500e+01 7.8700e+00 0.0000e+00 5.2400e-01 5.6310e+00  
1.0000e+02 6.0821e+00 5.0000e+00 3.1100e+02 1.5200e+01 3.8663e+02
```

特徴量の値
大きさ: (506,13)

目的値
個数: 506

散布図



Linnerud dataset

Linnerud dataset

■ 生理学的特徴と運動能力の関係

	回帰
データ数	20
特徴量	3
目的変数	3

linnerud.py

```
import numpy as np
```

```
from sklearn import datasets
```

```
from matplotlib import pyplot as plt
```

データセットを用いるために必要

散布図の描画のために必要

```
# データのロード
```

```
linnerud = datasets.load_linnerud()
```

datasets.load_linnerud()

linnerudデータセットのロード

```
# データの説明
```

```
print(linnerud.DESCR)
```

DESCR

linnerudデータセットの説明

```
# 特徴量(3次元)
```

```
feature_name = linnerud.feature_names
```

```
print( "¥n [ 特徴量 ]" )
```

```
print( feature_name )
```

feature_names

特徴量の名前

```
# 特徴量のデータ
```

```
data = linnerud.data
```

data

特徴量

大きさ

(20,3)

目的変数

```
label_name = linnerud.target_names
```

target_names

目的変数(クラス)の名前

```
print( "¥n [ クラス名 ]" )
```

```
print( label_name )
```

目的変数の値

```
label = linnerud.target
```

target

目的変数の値

個数

20

特徴量, 目的値の表示

```
print( "¥n 特徴量      : 目的値" )
```

データ(特徴量, 目的値)の表示

```
for i in range(len(label)):
```

```
    print( data[i] , ":" , label[i] )
```

散布図の表示

```
fig = plt.figure(figsize=(10,8))
```

グラフの描画領域の大きさ

```
plt.subplots_adjust(wspace=0.4, hspace=0.6)
```

```
color = [ 'r' , 'g' , 'b' ]
```

```
count = 1
```

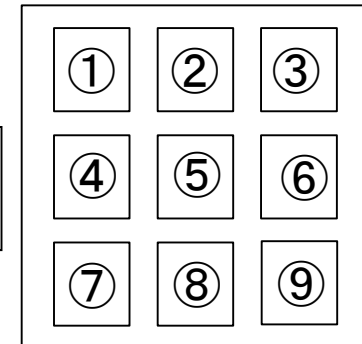
```
for i in range(3):
```

```
    for j in range(3):
```

```
        plt.subplot(3,3,count)
```

9個の散布図の表示

subplot(3,3,番号)
番号の欄にグラフを表示



```
        x0 = data[:,i]
```

x軸のデータ

```
        x1 = label[:,j]
```

y軸のデータ

```
        plt.scatter(x0, x1, color=color[i])
```

scatter(xの値, yの値, c=色)

```
        plt.xlabel(feature_name[i])
```

xlabel(x軸の説明)
ylabel(y軸の説明)

```
        plt.ylabel(label_name[j])
```

```
        plt.title(feature_name[i]+"-"+label_name[j])
```

title(タイトル)
グラフごとのタイトル

```
count+=1
```

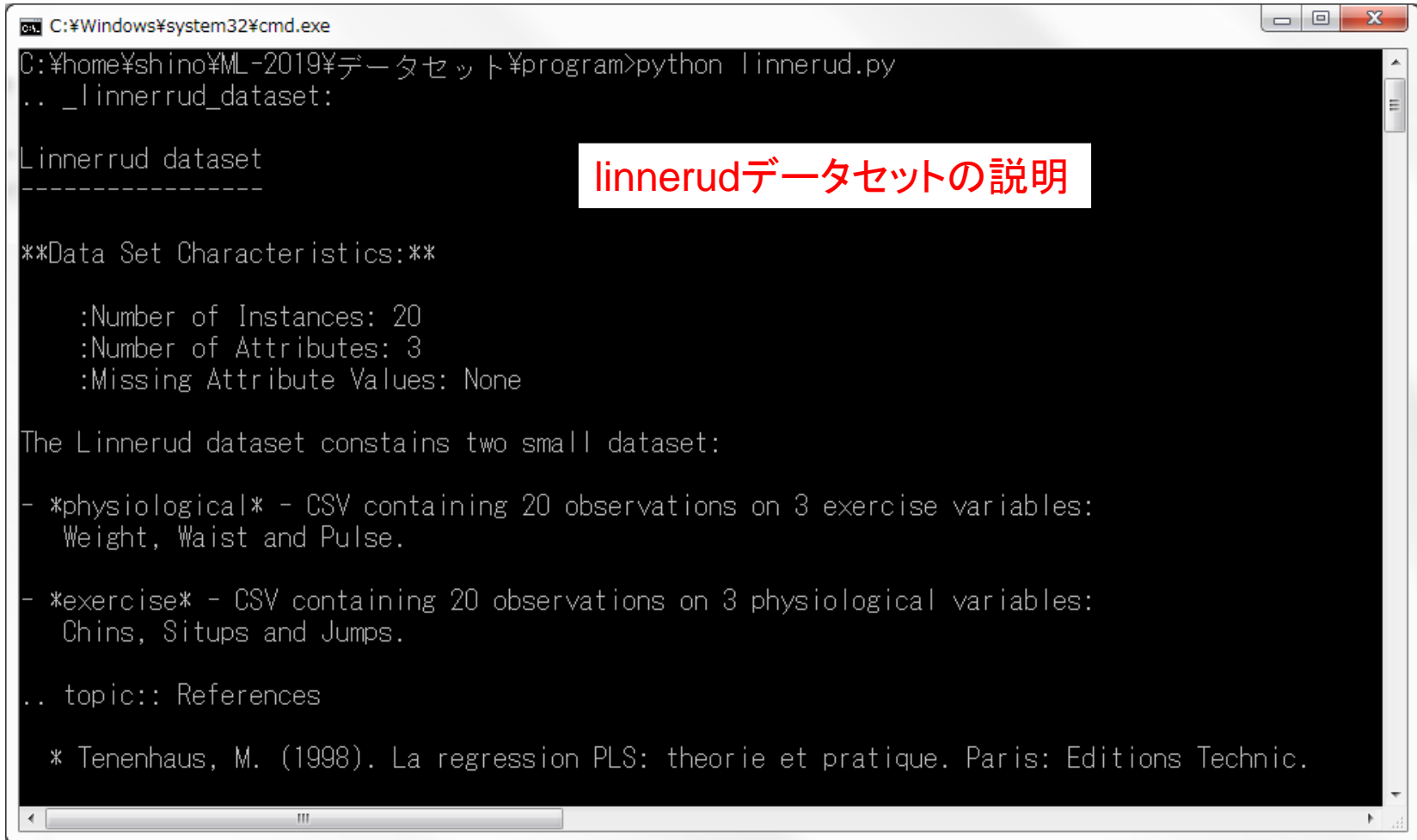
```
plt.suptitle('linnerud')
```

```
plt.show()
```

suptitle(タイトル)
グラフのタイトル

show()
グラフの表示

実行結果



```
C:\Windows\system32\cmd.exe
C:\home\shino\ML-2019\データセット\program>python linnerud.py
.. _linerrud_dataset:

Linnerrud dataset
-----

**Data Set Characteristics:**

: Number of Instances: 20
: Number of Attributes: 3
: Missing Attribute Values: None

The Linnerud dataset contains two small datasets:

- *physiological* - CSV containing 20 observations on 3 exercise variables:
  Weight, Waist and Pulse.

- *exercise* - CSV containing 20 observations on 3 physiological variables:
  Chins, Situps and Jumps.

.. topic:: References

* Tenenhaus, M. (1998). La regression PLS: theorie et pratique. Paris: Editions Technic.
```

linnerudデータセットの説明

```
C:\Windows\system32\cmd.exe

[ 特徴量 ]
['Chins', 'Situps', 'Jumps']

[ クラス名 ]
['Weight', 'Waist', 'Pulse']

特徴量          : 目的値
[ 5. 162. 60.] [191. 36. 50.]
[ 2. 110. 60.] [189. 37. 52.]
[12. 101. 101.] [193. 38. 58.]
[12. 105. 37.] [162. 35. 62.]
[13. 155. 58.] [189. 35. 46.]
[ 4. 101. 42.] [182. 36. 56.]
[ 8. 101. 38.] [211. 38. 56.]
[ 6. 125. 40.] [167. 34. 60.]
[15. 200. 40.] [176. 31. 74.]
[17. 251. 250.] [154. 33. 56.]
[17. 120. 38.] [169. 34. 50.]
[13. 210. 115.] [166. 33. 52.]
[14. 215. 105.] [154. 34. 64.]
[ 1. 50. 50.] [247. 46. 50.]
[ 6. 70. 31.] [193. 36. 46.]
[12. 210. 120.] [202. 37. 62.]
[ 4. 60. 25.] [176. 37. 54.]
[11. 230. 80.] [157. 32. 52.]
[15. 225. 73.] [156. 33. 54.]
[ 2. 110. 43.] [138. 33. 68.]
```

特徴量の名前

目的値の名前

特徴量の値
大きさ: (20,3)

目的値
大きさ: (20,3)

散布図

