Type *Markdown* and LaTeX: *𝛼*2

In [15]:

**import** pandas **as** pd

**import** numpy **as** np

df**=**pd.DataFrame(np.random.randn(5,3),index**=**['a','c','e','f','h'],columns**=**[' print(df)

one two three a -1.417411 -0.366846 -1.947782

c -1.688598 0.795150 -1.598514

e -0.057151 0.188724 -0.549276

f -0.778126 -0.391546 0.237263

h 1.493284 -0.469932 -0.742576

In [16]:

df**=**df.reindex(['a','b','c','d','e','f','g','h']) print(df)

|  |  |  |
| --- | --- | --- |
| one | two | three |
| a -1.417411 | -0.366846 | -1.947782 |
| b NaN | NaN | NaN |
| c -1.688598 | 0.795150 | -1.598514 |
| d NaN | NaN | NaN |
| e -0.057151 | 0.188724 | -0.549276 |
| f -0.778126 | -0.391546 | 0.237263 |
| g NaN | NaN | NaN |
| h 1.493284 | -0.469932 | -0.742576 |

In [17]:

print(df.dropna())

one two three a -1.417411 -0.366846 -1.947782

c -1.688598 0.795150 -1.598514

e -0.057151 0.188724 -0.549276

f -0.778126 -0.391546 0.237263

h 1.493284 -0.469932 -0.742576

In [18]:

b**=**df

print(df)

|  |  |  |
| --- | --- | --- |
| one | two | three |
| a -1.417411 | -0.366846 | -1.947782 |
| b NaN | NaN | NaN |
| c -1.688598 | 0.795150 | -1.598514 |
| d NaN | NaN | NaN |
| e -0.057151 | 0.188724 | -0.549276 |
| f -0.778126 | -0.391546 | 0.237263 |
| g NaN | NaN | NaN |
| h 1.493284 | -0.469932 | -0.742576 |

In [19]:

df2**=**b

print(b)

|  |  |  |
| --- | --- | --- |
| one | two | three |
| a -1.417411 | -0.366846 | -1.947782 |
| b NaN | NaN | NaN |
| c -1.688598 | 0.795150 | -1.598514 |
| d NaN | NaN | NaN |
| e -0.057151 | 0.188724 | -0.549276 |
| f -0.778126 | -0.391546 | 0.237263 |
| g NaN | NaN | NaN |
| h 1.493284 | -0.469932 | -0.742576 |

In [20]:

print(df2.fillna(method**=**'pad'))

|  |  |  |
| --- | --- | --- |
| one | two | three |
| a -1.417411 | -0.366846 | -1.947782 |
| b -1.417411 | -0.366846 | -1.947782 |
| c -1.688598 | 0.795150 | -1.598514 |
| d -1.688598 | 0.795150 | -1.598514 |
| e -0.057151 | 0.188724 | -0.549276 |
| f -0.778126 | -0.391546 | 0.237263 |
| g -0.778126 | -0.391546 | 0.237263 |
| h 1.493284 | -0.469932 | -0.742576 |

In [22]:

df4**=**df2

print(df4.fillna(method**=**'bfill'))

|  |  |  |
| --- | --- | --- |
| one | two | three |
| a -1.417411 | -0.366846 | -1.947782 |
| b -1.688598 | 0.795150 | -1.598514 |
| c -1.688598 | 0.795150 | -1.598514 |
| d -0.057151 | 0.188724 | -0.549276 |
| e -0.057151 | 0.188724 | -0.549276 |
| f -0.778126 | -0.391546 | 0.237263 |
| g 1.493284 | -0.469932 | -0.742576 |
| h 1.493284 | -0.469932 | -0.742576 |

In [23]:

print(df['one'].notnull())

1. True
2. False
3. True
4. False
5. True
6. True
7. False
8. True

Name: one, dtype: bool

In [24]:

print(df['one'].isnull())

1. False
2. True
3. False
4. True
5. False
6. False
7. True
8. False

Name: one, dtype: bool

In [25]:

a1**=**pd.DataFrame([['ajay',18],['arun',19],['ashwin',21]],columns**=**['name','ag print(a1)

name age

1. ajay 18
2. arun 19
3. ashwin 21

In [26]:

print(a1.replace({18:15,19:17,21:20}))

name age

1. ajay 15
2. arun 17
3. ashwin 20

In [6]:

**import** pandas **as** pd

**import** numpy **as** np

a1**=**pd.DataFrame([['lion',300],['leapord',190],['tiger',211],['lion',298],[' print(a1)

animal speed

1. lion 300
2. leapord 190
3. tiger 211
4. lion 298
5. tiger 255

In [7]:

a2**=**a1.groupby(['animal']).mean() print(a2)

|  |  |
| --- | --- |
| animal | speed |
| leapord | 190.0 |
| lion | 299.0 |
| tiger | 233.0 |

In [8]:

a2**=**a1.groupby(['animal']).sum() print(a2)

|  |  |
| --- | --- |
|  | speed |
| animal |  |
| leapord | 190 |
| lion | 598 |
| tiger | 466 |

In [9]:

a2**=**a1.groupby(['animal']).count() print(a2)

speed

animal

leapord 1

lion 2

tiger 2

In [10]:

a2**=**a1.groupby(['animal']).first() print(a2)

|  |  |
| --- | --- |
|  | speed |
| animal |  |
| leapord | 190 |
| lion | 300 |
| tiger | 211 |

In [11]:

a2**=**a1.groupby(['animal']).last() print(a2)

|  |  |
| --- | --- |
|  | speed |
| animal |  |
| leapord | 190 |
| lion | 298 |
| tiger | 255 |

In [2]:

**import** datetime **as** d r**=**d.datetime.now()

print(r)

2024-08-21 10:42:02.877264

In [3]:

**import** datetime **as** d r**=**d.datetime.today() print(r)

2024-08-21 10:43:01.811434

In [5]:

**import** datetime **as** d

r1**=**d.datetime(2020,6,8,23,10,25,404040)

print(r1)

2020-06-08 23:10:25.404040

In [6]:

print(r1.replace(day**=**10))

2020-06-10 23:10:25.404040

In [7]:

print(r1.replace(month**=**11))

2020-11-08 23:10:25.404040

In [8]:

print(r1.replace(year**=**2004))

2004-06-08 23:10:25.404040

In [9]:

print(r1.replace(day**=**10,month**=**11,year**=**2004))

2004-11-10 23:10:25.404040

In [10]:

**from** datetime **import** date print(date(2004,11,10))

2004-11-10

In [11]:

**from** datetime **import** date

print(date(2004,11,10).ctime())

Wed Nov 10 00:00:00 2004

In [16]:

print(r.strftime("%Y"))

2024

In [15]:

print(r.strftime("%y"))

24

In [17]:

print(r.strftime("%m"))

08

In [18]:

print(r.strftime("%b"))

Aug

In [19]:

print(r.strftime("%B"))

August

In [20]:

print(r.strftime("%j"))

234

In [21]:

print(r.strftime("%D"))

08/21/24

In [22]:

print(r.strftime("%d"))

21

In [23]:

print(r.strftime("%a"))

Wed

In [24]:

print(r.strftime("%A"))

Wednesday

In [25]:

print(r.strftime("%H"))

10

In [26]:

print(r.strftime("%S"))

01

In [27]:

print(r.strftime("%C"))

20

In [28]:

print(r.strftime("%c"))

Wed Aug 21 10:43:01 2024

In [29]:

print(r.strftime("%F"))

2024-08-21

In [30]:

print(r.strftime("%f"))

811434

In [31]:

pri nt(r.strftime("%p"))

AM

In [32]:

print(r.strftime("%x"))

08/21/24

In [33]:

print(r.strftime("%X"))

10:43:01

In [34]:

print(r.strftime("%r"))

10:43:01 AM

In [36]:

print(r.strftime("%T"))

10:43:01

In [2]:

{'data': array([[5.1, 3.5, 1.4, 0.2],

**from** sklearn **import** datasets

**import** pandas **as** pd

iris**=**datasets.load\_iris() print(iris)

|  |  |  |  |
| --- | --- | --- | --- |
| [4.9, | 3. , | 1.4, | 0.2], |
| [4.7, | 3.2, | 1.3, | 0.2], |
| [4.6, | 3.1, | 1.5, | 0.2], |
| [5. , | 3.6, | 1.4, | 0.2], |
| [5.4, | 3.9, | 1.7, | 0.4], |
| [4.6, | 3.4, | 1.4, | 0.3], |
| [5. , | 3.4, | 1.5, | 0.2], |
| [4.4, | 2.9, | 1.4, | 0.2], |
| [4.9, | 3.1, | 1.5, | 0.1], |
| [5.4, | 3.7, | 1.5, | 0.2], |
| [4.8, | 3.4, | 1.6, | 0.2], |
| [4.8, | 3. , | 1.4, | 0.1], |
| [4.3, | 3. , | 1.1, | 0.1], |
| [5.8, | 4. , | 1.2, | 0.2], |
| [5.7, | 4.4, | 1.5, | 0.4], |
| [5.4, | 3.9, | 1.3, | 0.4], |
| [5.1, | 3.5, | 1.4, | 0.3], |
| [5.7, | 3.8, | 1.7, | 0.3], |

In [3]:

print(type(iris))

<class 'sklearn.utils.\_bunch.Bunch'>

In [4]:

print(iris.keys())

dict\_keys(['data', 'target', 'frame', 'target\_names', 'DESCR', 'feature\_na mes', 'filename', 'data\_module'])

In [5]:

print(type(object))

<class 'type'>

In [6]:

print(type(iris.data))

<class 'numpy.ndarray'>

In [7]:

print(type(iris.target))

<class 'numpy.ndarray'>

In [8]:

print(iris.data.shape)

(150, 4)

In [9]:

print(iris.target\_names)

['setosa' 'versicolor' 'virginica']

In [10]:

x**=**iris.data y**=**iris.target print(x)

print(y)

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

In [19]:

df**=**pd.DataFrame(x,columns**=**iris.feature\_names) print(df)

sepal length (cm) sepal width (cm) petal length (cm) petal width

(cm)

0 5.1 3.5 1.4

0.2

1 4.9 3.0 1.4

0.2

2 4.7 3.2 1.3

0.2

3 4.6 3.1 1.5

0.2

4 5.0 3.6 1.4

0.2

.. ... ... ...

...

145 6.7 3.0 5.2

2.3

146 6.3 2.5 5.0

1.9

147 6.5 3.0 5.2

2.0

148 6.2 3.4 5.4

2.3

149 5.9 3.0 5.1

1.8

[150 rows x 4 columns]

In [20]:

print(df.head())

sepal length (cm) sepal width (cm) petal length (cm) petal width (c

m)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | 0 | 5.1 | 3.5 | 1.4 | 0. |
| 2 |  |  |  |  |
| 1 | 4.9 | 3.0 | 1.4 | 0. |
| 2 |  |  |  |  |
| 2 | 4.7 | 3.2 | 1.3 | 0. |
| 2 |  |  |  |  |
| 3 | 4.6 | 3.1 | 1.5 | 0. |
| 2 |  |  |  |  |
| 4 | 5.0 | 3.6 | 1.4 | 0. |
| 2 |  |  |  |  |
| In | [21]: | print(df.tail()) |  |  |  |  |

sepal length (cm) sepal width (cm) petal length (cm) petal width

(cm)

145 6.7 3.0 5.2

2.3

146 6.3 2.5 5.0

1.9

147 6.5 3.0 5.2

2.0

148 6.2 3.4 5.4

2.3

149 5.9 3.0 5.1

1.8

In [22]:

print(df.describe())

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | sepal | length (cm) | sepal | width (cm) | petal | length (cm) | \ |
| count |  | 150.000000 |  | 150.000000 |  | 150.000000 |  |
| mean |  | 5.843333 |  | 3.057333 |  | 3.758000 |  |
| std |  | 0.828066 |  | 0.435866 |  | 1.765298 |  |
| min |  | 4.300000 |  | 2.000000 |  | 1.000000 |  |
| 25% |  | 5.100000 |  | 2.800000 |  | 1.600000 |  |
| 50% |  | 5.800000 |  | 3.000000 |  | 4.350000 |  |
| 75% |  | 6.400000 |  | 3.300000 |  | 5.100000 |  |
| max |  | 7.900000 |  | 4.400000 |  | 6.900000 |  |
| count | petal | width (cm) 150.000000 | | | | | |
| mean |  | 1.199333 | | | | | |
| std |  | 0.762238 | | | | | |
| min |  | 0.100000 | | | | | |
| 25% |  | 0.300000 | | | | | |
| 50% |  | 1.300000 | | | | | |
| 75% |  | 1.800000 | | | | | |
| max |  | 2.500000 | | | | | |

In [24]:

print(df.min())

|  |  |  |
| --- | --- | --- |
| sepal | length (cm) | 4.3 |
| sepal | width (cm) | 2.0 |
| petal | length (cm) | 1.0 |
| petal | width (cm) | 0.1 |

dtype: float64

In [25]:

print(df.max())

|  |  |  |
| --- | --- | --- |
| sepal | length (cm) | 7.9 |
| sepal | width (cm) | 4.4 |
| petal | length (cm) | 6.9 |
| petal | width (cm) | 2.5 |

dtype: float64

In [26]:

**from** sklearn **import** datasets

**import** pandas **as** pd

dia**=**datasets.load\_diabetes() print(dia)

{'data': array([[ 0.03807591, 0.05068012, 0.06169621, ..., -0.00259226,

|  |  |  |  |
| --- | --- | --- | --- |
| 0.01990749, | | -0.01764613], |  |
| [-0.00188202, | | -0.04464164, -0.05147406, | ..., -0.03949338, |
| -0.06833155, | | -0.09220405], |  |
| [ 0.08529891, | | 0.05068012, 0.04445121, | ..., -0.00259226, |
| 0.00286131, | | -0.02593034], |  |
| ..., | |  |  |
| [ 0.04170844, | | 0.05068012, -0.01590626, | ..., -0.01107952, |
| -0.04688253, | | 0.01549073], |  |
| [-0.04547248, | | -0.04464164, 0.03906215, | ..., 0.02655962, |
| 0.04452873, | | -0.02593034], |  |
| [-0.04547248, | | -0.04464164, -0.0730303 , | ..., -0.03949338, |
| -0.00422151, | | 0.00306441]]), 'target': | array([151., 75., 141., 20 |
| 6., | 135., 97., 138., 63., 110., 310., 101., | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 69., | 179., | 185., | 118., | 171., | 166., | 144., | 97., | 168., | 68., | 49., |
| 68., | 245., | 184., | 202., | 137., | 85., | 131., | 283., | 129., | 59., | 341., |
| 87., | 65., | 102., | 265., | 276., | 252., | 90., | 100., | 55., | 61., | 92., |
| 259., | 53., | 190., | 142., | 75., | 142., | 155., | 225., | 59., | 104., | 182., |
| 128., | 52., | 37., | 170., | 170., | 61., | 144., | 52., | 128., | 71., | 163., |
| 150., | 97., | 160., | 178., | 48., | 270., | 202., | 111., | 85., | 42., | 170., |
| 200., | 252., | 113., | 143., | 51., | 52., | 210., | 65., | 141., | 55., | 134., |
| 42., | 111., | 98., | 164., | 48., | 96., | 90., | 162., | 150., | 279., | 92., |
| 83., | 128., | 102., | 302., | 198., | 95., | 53., | 134., | 144., | 232., | 81., |
| 104., | 59., | 246., | 297., | 258., | 229., | 275., | 281., | 179., | 200., | 200., |
| 173., | 180., | 84., | 121., | 161., | 99., | 109., | 115., | 268., | 274., | 158., |
| 107., | 83., | 103., | 272., | 85., | 280., | 336., | 281., | 118., | 317., | 235., |
| 60., | 174., | 259., | 178., | 128., | 96., | 126., | 288., | 88., | 292., | 71., |
| 197., | 186., | 25., | 84., | 96., | 195., | 53., | 217., | 172., | 131., | 214., |
| 59., | 70., | 220., | 268., | 152., | 47., | 74., | 295., | 101., | 151., | 127., |
| 237., | 225., | 81., | 151., | 107., | 64., | 138., | 185., | 265., | 101., | 137., |
| 143., | 141., | 79., | 292., | 178., | 91., | 116., | 86., | 122., | 72., | 129., |
| 142., | 90., | 158., | 39., | 196., | 222., | 277., | 99., | 196., | 202., | 155., |
| 77., | 191., | 70., | 73., | 49., | 65., | 263., | 248., | 296., | 214., | 185., |
| 78., | 93., | 252., | 150., | 77., | 208., | 77., | 108., | 160., | 53., | 220., |
| 154., | 259., | 90., | 246., | 124., | 67., | 72., | 257., | 262., | 275., | 177., |
| 71., | 47., | 187., | 125., | 78., | 51., | 258., | 215., | 303., | 243., | 91., |
| 150., | 310., | 153., | 346., | 63., | 89., | 50., | 39., | 103., | 308., | 116., |
| 145., | 74., | 45., | 115., | 264., | 87., | 202., | 127., | 182., | 241., | 66., |
| 94., | 283., | 64., | 102., | 200., | 265., | 94., | 230., | 181., | 156., | 233., |
| 60., | 219., | 80., | 68., | 332., | 248., | 84., | 200., | 55., | 85., | 89., |
| 31., | 129., | 83., | 275., | 65., | 198., | 236., | 253., | 124., | 44., | 172., |
| 114., | 142., | 109., | 180., | 144., | 163., | 147., | 97., | 220., | 190., | 109., |
| 191., | 122., | 230., | 242., | 248., | 249., | 192., | 131., | 237., | 78., | 135., |
| 244., | 199., | 270., | 164., | 72., | 96., | 306., | 91., | 214., | 95., | 216., |
| 263., | 178., | 113., | 200., | 139., | 139., | 88., | 148., | 88., | 243., | 71., |
| 77., | 109., | 272., | 60., | 54., | 221., | 90., | 311., | 281., | 182., | 321., |
| 58., | 262., | 206., | 233., | 242., | 123., | 167., | 63., | 197., | 71., | 168., |
| 140., | 217., | 121., | 235., | 245., | 40., | 52., | 104., | 132., | 88., | 69., |
| 219., | 72., | 201., | 110., | 51., | 277., | 63., | 118., | 69., | 273., | 258., |
| 43., | 198., | 242., | 232., | 175., | 93., | 168., | 275., | 293., | 281., | 72., |
| 140., | 189., | 181., | 209., | 136., | 261., | 113., | 131., | 174., | 257., | 55., |
| 84., | 42., | 146., | 212., | 233., | 91., | 111., | 152., | 120., | 67., | 310., |
| 94., | 183., | 66., | 173., | 72., | 49., | 64., | 48., | 178., | 104., | 132., |

220., 57.]), 'frame': None, 'DESCR': '.. \_diabetes\_dataset:\n\nDia betes dataset\n----------------\n\nTen baseline variables, age, sex, body mass index, average blood\npressure, and six blood serum measurements were obtained for each of n =\n442 diabetes patients, as well as the response o f interest, a\nquantitative measure of disease progression one year after baseline.\n\n\*\*Data Set Characteristics:\*\*\n\n :Number of Instances: 442

\n\n :Number of Attributes: First 10 columns are numeric predictive value s\n\n :Target: Column 11 is a quantitative measure of disease progression

one year after baseline\n\n :Attribute Information:\n - age age in years\n - sex\n - bmi body mass index\n - bp av erage blood pressure\n - s1 tc, total serum cholesterol\n - s2 ldl, low-density lipoproteins\n - s3 hdl, high-density l ipoproteins\n - s4 tch, total cholesterol / HDL\n - s5

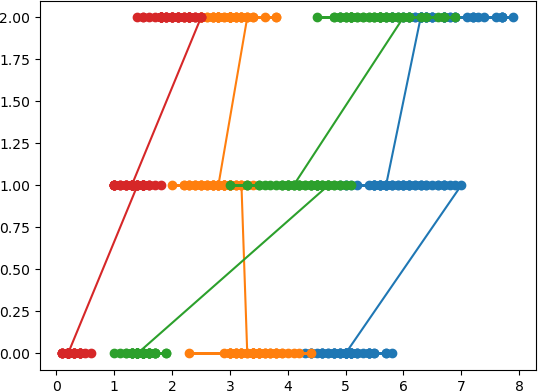
ltg, possibly log of serum triglycerides level\n - s6 glu, blood sugar level\n\nNote: Each of these 10 feature variables have been mean cen tered and scaled by the standard deviation times the square root of `n\_sam ples` (i.e. the sum of squares of each column totals 1).\n\nSource URL:\nh ttps://www4.stat.ncsu.edu/~boos/var.select/diabetes.html\n\nFor more infor mation see:\nBradley Efron, Trevor Hastie, Iain Johnstone and Robert Tibsh irani (2004) "Least Angle Regression," Annals of Statistics (with discussi on), 407-499.\n(https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle\_20 02.pdf)\n', 'feature\_names': ['age', 'sex', 'bmi', 'bp', 's1', 's2', 's3', 's4', 's5', 's6'], 'data\_filename': 'diabetes\_data\_raw.csv.gz', 'target\_fi lename': 'diabetes\_target.csv.gz', 'data\_module': 'sklearn.datasets.data'}

In [28]:

**import** matplotlib.pyplot

**import** matplotlib.pyplot **as** pl pl.plot(x,y,marker**=**'o')

pl.show()



In [29]:

**from** sklearn **import** datasets

**import** pandas **as** pd

bc**=**datasets.load\_breast\_cancer() print(bc)

{'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-0 1,

1.189e-01],

[2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01, 8.902e-02],

[1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01, 8.758e-02],

...,

[1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01, 7.820e-02],

[2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01, 1.240e-01],

[7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,

7.039e-02]]), 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 1, 1, 1,

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 1, | 0, | 0, | 0, | 0, | 0, | 0, |
| 0, | 0, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 0, | 0, | 1, | 0, | 0, | 1, | 1, | 1, | 1, | 0, | 1, | 0, | 0, |
| 1, | 1, | 1, | 1, | 0, | 1, | 0, | 0, | 1, | 0, | 1, | 0, | 0, | 1, | 1, | 1, | 0, | 0, | 1, | 0, | 0, | 0, |
| 1, | 1, | 1, | 0, | 1, | 1, | 0, | 0, | 1, | 1, | 1, | 0, | 0, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 0, | 1, |
| 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 0, | 0, | 1, | 0, | 0, | 1, | 1, | 1, | 0, | 0, | 1, | 0, | 1, | 0, |
| 0, | 1, | 0, | 0, | 1, | 1, | 0, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 1, |
| 1, | 1, | 0, | 1, | 1, | 1, | 1, | 0, | 0, | 1, | 0, | 1, | 1, | 0, | 0, | 1, | 1, | 0, | 0, | 1, | 1, | 1, |
| 1, | 0, | 1, | 1, | 0, | 0, | 0, | 1, | 0, | 1, | 0, | 1, | 1, | 1, | 0, | 1, | 1, | 0, | 0, | 1, | 0, | 0, |
| 0, | 0, | 1, | 0, | 0, | 0, | 1, | 0, | 1, | 0, | 1, | 1, | 0, | 1, | 0, | 0, | 0, | 0, | 1, | 1, | 0, | 0, |
| 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 0, | 0, | 1, | 1, | 0, | 1, | 1, | 0, | 0, | 1, | 0, | 1, | 1, |
| 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, | 0, |
| 0, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 0, | 1, | 1, | 0, | 1, | 1, | 0, | 1, | 0, | 0, | 1, | 1, |
| 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 0, | 1, | 0, | 1, | 1, | 1, | 1, | 1, |
| 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 0, | 1, | 0, | 1, | 1, | 1, | 1, | 0, | 0, |
| 0, | 1, | 1, | 1, | 1, | 0, | 1, | 0, | 1, | 0, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, |
| 0, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 0, | 1, | 0, | 0, | 0, | 1, | 0, | 0, |
| 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 0, | 1, | 1, | 0, | 0, | 1, | 1, |
| 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 0, |
| 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 0, | 0, | 1, | 0, | 1, | 1, | 1, | 1, |
| 1, | 0, | 1, | 1, | 0, | 1, | 0, | 1, | 1, | 0, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 0, |
| 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, |
| 1, | 1, | 1, | 0, | 1, | 0, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 0, | 0, | 1, | 0, | 1, | 0, | 1, | 1, |
| 1, | 1, | 1, | 0, | 1, | 1, | 0, | 1, | 0, | 1, | 0, | 0, | 1, | 1, | 1, | 0, | 1, | 1, | 1, | 1, | 1, | 1, |
| 1, | 1, | 1, | 1, | 1, | 0, | 1, | 0, | 0, | 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, 0, 0, 0, 0, 0, 0, 1]), 'frame':

None, 'target\_names': array(['malignant', 'benign'], dtype='<U9'), 'DESC

R': '.. \_breast\_cancer\_dataset:\n\nBreast cancer wisconsin (diagnostic) da taset\n--------------------------------------------\n\n\*\*Data Set Characte ristics:\*\*\n\n :Number of Instances: 569\n\n :Number of Attributes:

30 numeric, predictive attributes and the class\n\n :Attribute Informat ion:\n - radius (mean of distances from center to points on the per imeter)\n - texture (standard deviation of gray-scale values)\n

* perimeter\n - area\n - smoothness (local variation in radi us lengths)\n - compactness (perimeter^2 / area - 1.0)\n - c oncavity (severity of concave portions of the contour)\n - concave points (number of concave portions of the contour)\n - symmetry\n
* fractal dimension ("coastline approximation" - 1)\n\n The mean, s tandard error, and "worst" or largest (mean of the three\n worst/la rgest values) of these features were computed for each image,\n res ulting in 30 features. For instance, field 0 is Mean Radius, field\n

10 is Radius SE, field 20 is Worst Radius.\n\n - class:\n

* WDBC-Malignant\n - WDBC-Benign\n\n :Summary Statistic s:\n\n ===================================== ====== ======\n

Min Max\n ===================================== ====== ======\n r

adius (mean): 6.981 28.11\n texture (mean): 9.71 39.28\n perimeter (mean): 43.79 188.5\n

area (mean): 143.5 2501.0\n smoothness (mea n): 0.053 0.163\n compactness (mean):

0.019 0.345\n concavity (mean): 0.0 0.427\n

concave points (mean): 0.0 0.201\n symmetry (mean):

0.106 0.304\n fractal dimension (mean): 0.05 0.097\n

radius (standard error): 0.112 2.873\n texture (standard error): 0.36 4.885\n perimeter (standard error):

0.757 21.98\n area (standard error): 6.802 542.2\n

smoothness (standard error): 0.002 0.031\n compactness (stand ard error): 0.002 0.135\n concavity (standard error):

0.0 0.396\n concave points (standard error): 0.0 0.053\n

symmetry (standard error): 0.008 0.079\n fractal dimension (standard error): 0.001 0.03\n radius (worst):

7.93 36.04\n texture (worst): 12.02 49.54\n perimeter (worst): 50.41 251.2\n area (worst):

185.2 4254.0\n smoothness (worst): 0.071 0.223\n

compactness (worst): 0.027 1.058\n concavity (worst):

0.0 1.252\n concave points (worst): 0.0 0.291\n

symmetry (worst): 0.156 0.664\n fractal dimension (worst): 0.055 0.208\n ====================================

= ====== ======\n\n :Missing Attribute Values: None\n\n :Class Distr ibution: 212 - Malignant, 357 - Benign\n\n :Creator: Dr. William H. Wo lberg, W. Nick Street, Olvi L. Mangasarian\n\n :Donor: Nick Street\n\n

:Date: November, 1995\n\nThis is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.\nhttps://goo.gl/U2Uwz2\n\nFeatures are computed fro m a digitized image of a fine needle\naspirate (FNA) of a breast mass. Th ey describe\ncharacteristics of the cell nuclei present in the image.\n\nS eparating plane described above was obtained using\nMultisurface Method-Tr ee (MSM-T) [K. P. Bennett, "Decision Tree\nConstruction Via Linear Program ming." Proceedings of the 4th\nMidwest Artificial Intelligence and Cogniti ve Science Society,\npp. 97-101, 1992], a classification method which uses linear\nprogramming to construct a decision tree. Relevant features\nwere selected using an exhaustive search in the space of 1-4\nfeatures and 1-3 separating planes.\n\nThe actual linear program used to obtain the separat ing plane\nin the 3-dimensional space is that described in:\n[K. P. Bennet t and O. L. Mangasarian: "Robust Linear\nProgramming Discrimination of Two Linearly Inseparable Sets",\nOptimization Methods and Software 1, 1992, 23

-34].\n\nThis database is also available through the UW CS ftp server:\n\n ftp ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-learn/WDBC/\n\n.. to pic:: References\n\n - W.N. Street, W.H. Wolberg and O.L. Mangasarian. N uclear feature extraction \n for breast tumor diagnosis. IS&T/SPIE 199

3 International Symposium on \n Electronic Imaging: Science and Techno logy, volume 1905, pages 861-870,\n San Jose, CA, 1993.\n - O.L. Man gasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and \n

prognosis via linear programming. Operations Research, 43(4), pages 570-57 7, \n July-August 1995.\n - W.H. Wolberg, W.N. Street, and O.L. Mang asarian. Machine learning techniques\n to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994) \n 163-171.', 'feature

\_names': array(['mean radius', 'mean texture', 'mean perimeter', 'mean are a',

'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'], dtype='<U23'), 'filen ame': 'breast\_cancer.csv', 'data\_module': 'sklearn.datasets.data'}

In [ ]: