In [68]:

```
# lod  the data sets

import pandas as pd
pd.read_csv('C://Users/Dell/Downloads/creditcard/creditcard.csv')
```

Out[68]:

	Time	V1	V2	V3	V4	V5	V6	V7
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941
...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006

284807 rows × 31 columns

In [75]:

```
url = 'https://raw.githubusercontent.com/datasets/covid-19/main/data/countries-aggregated.c
df = pd.read_csv(url)
```

In [12]:

```
url = 'https://raw.githubusercontent.com/datasets/covid-19/main/data/us_deaths.csv'
df = pd.read_csv(url)
```

In [13]:

```
df
```

Out[13]:

	Admin2	Date	Case	Country/Region	Province/State
0	Autauga	2020-01-22	0	US	Alabama
1	Autauga	2020-01-23	0	US	Alabama
2	Autauga	2020-01-24	0	US	Alabama
3	Autauga	2020-01-25	0	US	Alabama
4	Autauga	2020-01-26	0	US	Alabama
...	...	...	...	...	...
2727067	Weston	2022-04-12	18	US	Wyoming
2727068	Weston	2022-04-13	18	US	Wyoming
2727069	Weston	2022-04-14	18	US	Wyoming
2727070	Weston	2022-04-15	18	US	Wyoming
2727071	Weston	2022-04-16	18	US	Wyoming

2727072 rows × 5 columns

In [15]:

```
import sklearn.datasets
```

In [17]:

```
from sklearn.datasets import load_iris

iris = load_iris()
```

In [18]:

```
iris
```

```
this day.  (See Duda & Hart, for example.)  The\ndata set contains 3 class
es of 50 instances each, where each class refers to a\ntype of iris plant.
One class is linearly separable from the other 2; the\nlatter are NOT line
arly separable from each other.\n\n.. topic:: References\n\n   - Fisher,
R.A. "The use of multiple measurements in taxonomic problems"\n     Annual
Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to\n     Math
ematical Statistics" (John Wiley, NY, 1950).\n   - Duda, R.O., & Hart, P.
E. (1973) Pattern Classification and Scene Analysis.\n     (Q327.D83) John
Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.\n   - Dasarathy, B.V. (1
980) "Nosing Around the Neighborhood: A New System\n     Structure and Cla
ssification Rule for Recognition in Partially Exposed\n     Environments".
IEEE Transactions on Pattern Analysis and Machine\n     Intelligence, Vol.
PAMI-2, No. 1, 67-71.\n   - Gates, G.W. (1972) "The Reduced Nearest Neighb
or Rule".  IEEE Transactions\n     on Information Theory, May 1972, 431-43
3.\n   - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLA
SS II\n     conceptual clustering system finds 3 classes in the data.\n
- Many, many more ...',
 'feature_names': ['sepal length (cm)',
  'sepal width (cm)',
  'petal length (cm)',
```

In [19]:

```
iris = load_iris(return_X_y = True)   # with out featuer and target
```

In [20]:

```
iris
```

```
        [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],
        [5.1, 3.8, 1.5, 0.3],
        [5.4, 3.4, 1.7, 0.2],
        [5.1, 3.7, 1.5, 0.4],
        [4.6, 3.6, 1. , 0.2],
```

In [21]:

```
X,y = load_iris(return_X_y = True) # with feather and targeat X,y
```

In [22]:

```
X
```

```
       [7.9, 3.8, 6.4, 2. ],
       [6.4, 2.8, 5.6, 2.2],
       [6.3, 2.8, 5.1, 1.5],
       [6.1, 2.6, 5.6, 1.4],
       [7.7, 3. , 6.1, 2.3],
       [6.3, 3.4, 5.6, 2.4],
       [6.4, 3.1, 5.5, 1.8],
       [6. , 3. , 4.8, 1.8],
       [6.9, 3.1, 5.4, 2.1],
       [6.7, 3.1, 5.6, 2.4],
       [6.9, 3.1, 5.1, 2.3],
       [5.8, 2.7, 5.1, 1.9],
       [6.8, 3.2, 5.9, 2.3],
       [6.7, 3.3, 5.7, 2.5],
       [6.7, 3. , 5.2, 2.3],
       [6.3, 2.5, 5. , 1.9],
       [6.5, 3. , 5.2, 2. ],
       [6.2, 3.4, 5.4, 2.3],
       [5.9, 3. , 5.1, 1.8]])
```

In [23]:

```
y
```

Out[23]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 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, 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])
```

In [24]:

```
:urn_X_y = True, as_frame = True) # use the as_frame = True for the dataset show the data fr
```

In [25]:

```
X
```

Out[25]:

	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 × 4 columns

In [26]:

```
y
```

Out[26]:

```
0      0
1      0
2      0
3      0
4      0
      ..
145    2
146    2
147    2
148    2
149    2
Name: target, Length: 150, dtype: int32
```

In [27]:

```
from sklearn.datasets import load_digits

digits = load_digits()
```

In [28]:

```
digits
```

```
        ...,
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  2., ..., 12.,  0.,  0.],
        [ 0.,  0., 10., ..., 12.,  1.,  0.]]),
 'target': array([0, 1, 2, ..., 8, 9, 8]),
 'frame': None,
 'feature_names': ['pixel_0_0',
  'pixel_0_1',
  'pixel_0_2',
  'pixel_0_3',
  'pixel_0_4',
  'pixel_0_5',
  'pixel_0_6',
  'pixel_0_7',
  'pixel_1_0',
  'pixel_1_1',
  'pixel_1_2',
  'pixel_1_3',
  'pixel_1_4',
  'pixel_1_5',
```

In [29]:

```
digits = load_digits(return_X_y = True)   # with out featuer and target
```

In [30]:

```
digits
```

Out[30]:

```
(array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
        [ 0.,  0.,  0., ..., 10.,  0.,  0.],
        [ 0.,  0.,  0., ..., 16.,  9.,  0.],
        ...,
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  2., ..., 12.,  0.,  0.],
        [ 0.,  0., 10., ..., 12.,  1.,  0.]]),
 array([0, 1, 2, ..., 8, 9, 8]))
```

In [31]:

```
from sklearn.datasets import load_diabetes

diabetes = load_diabetes()
```

In [32]:

```
diabetes
```

Out[32]:

```
{'data': array([[ 0.03807591,  0.05068012,  0.06169621, ..., -0.00259226,
          0.01990842, -0.01764613],
        [-0.00188202, -0.04464164, -0.05147406, ..., -0.03949338,
         -0.06832974, -0.09220405],
        [ 0.08529891,  0.05068012,  0.04445121, ..., -0.00259226,
          0.00286377, -0.02593034],
        ...,
        [ 0.04170844,  0.05068012, -0.01590626, ..., -0.01107952,
         -0.04687948,  0.01549073],
        [-0.04547248, -0.04464164,  0.03906215, ...,  0.02655962,
          0.04452837, -0.02593034],
        [-0.04547248, -0.04464164, -0.0730303 , ..., -0.03949338,
         -0.00421986,  0.00306441]]),
 'target': array([151.,  75., 141., 206., 135.,  97., 138.,  63., 110., 31
0., 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\nDiabetes dataset\n----------------\n\nTe
n baseline variables, age, sex, body mass index, average blood\npressure, an
d six blood serum measurements were obtained for each of n =\n442 diabetes p
atients, as well as the response of interest, a\nquantitative measure of dis
ease progression one year after baseline.\n\n**Data Set Characteristics:**\n
\n  :Number of Instances: 442\n\n  :Number of Attributes: First 10 columns a
re numeric predictive values\n\n  :Target: Column 11 is a quantitative measu
re of disease progression one year after baseline\n\n  :Attribute Informatio
n:\n      - age     age in years\n      - sex\n      - bmi     body mass ind
ex\n      - bp      average blood pressure\n      - s1      tc, total serum 
cholesterol\n      - s2      ldl, low-density lipoproteins\n      - s3      
hdl, high-density lipoproteins\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 centered and scaled by the standard deviation times `n_samples` (i.e. t
he sum of squares of each column totals 1).\n\nSource URL:\nhttps://www4.sta
t.ncsu.edu/~boos/var.select/diabetes.html\n\nFor more information see:\nBrad
ley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least
Angle Regression," Annals of Statistics (with discussion), 407-499.\n(http
s://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)',
 'feature_names': ['age',
  'sex',
  'bmi',
  'bp',
  's1',
  's2',
  's3',
  's4',
  's5',
  's6'],
 'data_filename': 'C:\\Users\\Dell\\anaconda3\\lib\\site-packages\\sklearn
\\datasets\\data\\diabetes_data.csv.gz',
 'target_filename': 'C:\\Users\\Dell\\anaconda3\\lib\\site-packages\\sklearn
\\datasets\\data\\diabetes_target.csv.gz'}
```

In [34]:

```
from sklearn.datasets import load_linnerud

linnerud = load_linnerud()
```

In [35]:

```
linnerud
```

Out[35]:

```
{'data': array([[  5., 162.,  60.],
        [  2., 110.,  60.],
        [ 12., 101., 101.],
        [ 12., 105.,  37.],
        [ 13., 155.,  58.],
        [  4., 101.,  42.],
        [  8., 101.,  38.],
        [  6., 125.,  40.],
        [ 15., 200.,  40.],
        [ 17., 251., 250.],
        [ 17., 120.,  38.],
        [ 13., 210., 115.],
        [ 14., 215., 105.],
        [  1.,  50.,  50.],
        [  6.,  70.,  31.],
        [ 12., 210., 120.],
        [  4.,  60.,  25.],
        [ 11., 230.,  80.],
        [ 15., 225.,  73.],
        [  2., 110.,  43.]]),
 'feature_names': ['Chins', 'Situps', 'Jumps'],
 'target': array([[191.,  36.,  50.],
        [189.,  37.,  52.],
        [193.,  38.,  58.],
        [162.,  35.,  62.],
        [189.,  35.,  46.],
        [182.,  36.,  56.],
        [211.,  38.,  56.],
        [167.,  34.,  60.],
        [176.,  31.,  74.],
        [154.,  33.,  56.],
        [169.,  34.,  50.],
        [166.,  33.,  52.],
        [154.,  34.,  64.],
        [247.,  46.,  50.],
        [193.,  36.,  46.],
        [202.,  37.,  62.],
        [176.,  37.,  54.],
        [157.,  32.,  52.],
        [156.,  33.,  54.],
        [138.,  33.,  68.]]),
 'target_names': ['Weight', 'Waist', 'Pulse'],
 'frame': None,
 'DESCR': '.. _linnerud_dataset:\n\nLinnerud dataset\n----------------\n
\n**Data Set Characteristics:**\n\n    :Number of Instances: 20\n    :Number
of Attributes: 3\n    :Missing Attribute Values: None\n\nThe Linnerud datase
t is a multi-output regression dataset. It consists of three\nexercise (dat
a) and three physiological (target) variables collected from\ntwenty middle-
aged men in a fitness club:\n\n- *physiological* - CSV containing 20 observa
tions on 3 physiological variables:\n   Weight, Waist and Pulse.\n- *exercis
e* - CSV containing 20 observations on 3 exercise variables:\n   Chins, Situ
ps and Jumps.\n\n.. topic:: References\n\n  * Tenenhaus, M. (1998). La regre
ssion PLS: theorie et pratique. Paris:\n    Editions Technic.\n',
 'data_filename': 'C:\\Users\\Dell\\anaconda3\\lib\\site-packages\\sklearn
\\datasets\\data/linnerud_exercise.csv',
```

```
 'target_filename': 'C:\\Users\\Dell\\anaconda3\\lib\\site-packages\\sklearn
\\datasets\\data/linnerud_physiological.csv'}
```

In [36]:

```
from sklearn.datasets import load_wine

wine = load_wine()
```

In [37]:

```
wine
```

Out[37]:

```
{'data': array([[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]]),
 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 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, 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]),
 'frame': None,
 'target_names': array(['class_0', 'class_1', 'class_2'], dtype='<U7'),
 'DESCR': '.. _wine_dataset:\n\nWine recognition dataset\n------------------
------\n\n**Data Set Characteristics:**\n\n    :Number of Instances: 178 (50
in each of three classes)\n    :Number of Attributes: 13 numeric, predictive
attributes and the class\n    :Attribute Information:\n \t\t- Alcohol\n \t\t
- Malic acid\n \t\t- Ash\n\t\t- Alcalinity of ash  \n \t\t- Magnesium\n\t\t-
Total phenols\n \t\t- Flavanoids\n \t\t- Nonflavanoid phenols\n \t\t- Proant
hocyanins\n\t\t- Color intensity\n \t\t- Hue\n \t\t- OD280/OD315 of diluted
wines\n \t\t- Proline\n\n    - class:\n            - class_0\n            -
class_1\n            - class_2\n\t\t\n    :Summary Statistics:\n    \n    ==
========================== ==== ===== ======= =====\n
Min   Max   Mean     SD\n    ============================ ==== ===== ======
= =====\n    Alcohol:                          11.0  14.8    13.0   0.8\n    Mal
ic Acid:                    0.74  5.80    2.34  1.12\n    Ash:
1.36  3.23    2.36  0.27\n    Alcalinity of Ash:            10.6  30.0    1
9.5   3.3\n    Magnesium:                       70.0 162.0    99.7  14.3\n    T
otal Phenols:                0.98  3.88    2.29  0.63\n    Flavanoids:
0.34  5.08    2.03  1.00\n    Nonflavanoid Phenols:         0.13  0.66    0.
36  0.12\n    Proanthocyanins:              0.41  3.58    1.59  0.57\n    Co
lour Intensity:             1.3  13.0     5.1   2.3\n    Hue:
0.48  1.71    0.96  0.23\n    OD280/OD315 of diluted wines: 1.27  4.00    2.
61  0.71\n    Proline:                         278  1680    746   315\n    ==
========================== ==== ===== ======= =====\n\n    :Missing Attribu
te Values: None\n    :Class Distribution: class_0 (59), class_1 (71), class_
2 (48)\n    :Creator: R.A. Fisher\n    :Donor: Michael Marshall (MARSHALL%PL
U@io.arc.nasa.gov)\n    :Date: July, 1988\n\nThis is a copy of UCI ML Wine r
ecognition datasets.\nhttps://archive.ics.uci.edu/ml/machine-learning-databa
ses/wine/wine.data\n\nThe data is the results of a chemical analysis of wine
s grown in the same\nregion in Italy by three different cultivators. There a
re thirteen different\nmeasurements taken for different constituents found i
n the three types of\nwine.\n\nOriginal Owners: \n\nForina, M. et al, PARVUS
```

```
- \nAn Extendible Package for Data Exploration, Classification and Correlati
on. \nInstitute of Pharmaceutical and Food Analysis and Technologies,\nVia B
rigata Salerno, 16147 Genoa, Italy.\n\nCitation:\n\nLichman, M. (2013). UCI
Machine Learning Repository\n[https://archive.ics.uci.edu/ml]. Irvine, CA: U
niversity of California,\nSchool of Information and Computer Science. \n\n..
topic:: References\n\n  (1) S. Aeberhard, D. Coomans and O. de Vel, \n  Comp
arison of Classifiers in High Dimensional Settings, \n  Tech. Rep. no. 92-0
2, (1992), Dept. of Computer Science and Dept. of  \n  Mathematics and Stati
stics, James Cook University of North Queensland. \n  (Also submitted to Tec
hnometrics). \n\n  The data was used with many others for comparing various
\n  classifiers. The classes are separable, though only RDA \n  has achieved
100% correct classification. \n  (RDA : 100%, QDA 99.4%, LDA 98.9%, 1NN 96.
1% (z-transformed data)) \n  (All results using the leave-one-out technique)
\n\n  (2) S. Aeberhard, D. Coomans and O. de Vel, \n  "THE CLASSIFICATION PE
RFORMANCE OF RDA" \n  Tech. Rep. no. 92-01, (1992), Dept. of Computer Scienc
e and Dept. of \n  Mathematics and Statistics, James Cook University of Nort
h Queensland. \n  (Also submitted to Journal of Chemometrics).\n',
 'feature_names': ['alcohol',
  'malic_acid',
  'ash',
  'alcalinity_of_ash',
  'magnesium',
  'total_phenols',
  'flavanoids',
  'nonflavanoid_phenols',
  'proanthocyanins',
  'color_intensity',
  'hue',
  'od280/od315_of_diluted_wines',
  'proline']}
```

In [38]:

```
from sklearn.datasets import load_breast_cancer

breast_cancer = load_breast_cancer()
```

In [39]:

```
breast_cancer
```

Out[39]:

```
{'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
         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, 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'),
 'DESCR': '.. _breast_cancer_dataset:\n\nBreast cancer wisconsin (diagnosti
c) dataset\n--------------------------------------------\n\n**Data Set Chara
cteristics:**\n\n    :Number of Instances: 569\n\n    :Number of Attributes:
30 numeric, predictive attributes and the class\n\n    :Attribute Informatio
n:\n        - radius (mean of distances from center to points on the perimet
er)\n        - texture (standard deviation of gray-scale values)\n        -
perimeter\n        - area\n        - smoothness (local variation in radius l
engths)\n        - compactness (perimeter^2 / area - 1.0)\n        - concavi
ty (severity of concave portions of the contour)\n        - concave points
(number of concave portions of the contour)\n        - symmetry\n        - f
ractal dimension ("coastline approximation" - 1)\n\n        The mean, standa
rd error, and "worst" or largest (mean of the three\n        worst/largest v
alues) of these features were computed for each image,\n        resulting in
```

```
30 features.  For instance, field 0 is Mean Radius, field\n        10 is Rad
ius SE, field 20 is Worst Radius.\n\n        - class:\n                - WDB
C-Malignant\n                - WDBC-Benign\n\n    :Summary Statistics:\n\n
===================================== ====== ======\n                                       
Min    Max\n    ===================================== ====== ======\n    rad
ius (mean):                        6.981  28.11\n    texture (mean):
9.71   39.28\n    perimeter (mean):                     43.79  188.5\n    ar
ea (mean):                          143.5  2501.0\n    smoothness (mean):
0.053  0.163\n    compactness (mean):                   0.019  0.345\n    co
ncavity (mean):                     0.0    0.427\n    concave points (mean):
0.0    0.201\n    symmetry (mean):                      0.106  0.304\n    fr
actal dimension (mean):             0.05   0.097\n    radius (standard erro
r):             0.112  2.873\n    texture (standard error):            0.3
6   4.885\n    perimeter (standard error):          0.757  21.98\n    area
(standard error):                6.802  542.2\n    smoothness (standard erro
r):         0.002  0.031\n    compactness (standard error):        0.002
0.135\n    concavity (standard error):          0.0    0.396\n    concave p
oints (standard error):      0.0    0.053\n    symmetry (standard error):
0.008  0.079\n    fractal dimension (standard error):   0.001  0.03\n    rad
ius (worst):                        7.93   36.04\n    texture (worst):
12.02  49.54\n    perimeter (worst):                    50.41  251.2\n    ar
ea (worst):                         185.2  4254.0\n    smoothness (worst):
0.071  0.223\n    compactness (worst):                  0.027  1.058\n    co
ncavity (worst):                    0.0    1.252\n    concave points (wors
t):              0.0    0.291\n    symmetry (worst):                    0.
156  0.664\n    fractal dimension (worst):           0.055  0.208\n    ====
================================= ====== ======\n\n    :Missing Attribute Va
lues: None\n\n    :Class Distribution: 212 - Malignant, 357 - Benign\n\n
:Creator:  Dr. William H. Wolberg, 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\nFea
tures are computed from a digitized image of a fine needle\naspirate (FNA) o
f a breast mass.  They describe\ncharacteristics of the cell nuclei present
in the image.\n\nSeparating plane described above was obtained using\nMultis
urface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree\nConstruction Via
Linear Programming." Proceedings of the 4th\nMidwest Artificial Intelligence
and Cognitive Science Society,\npp. 97-101, 1992], a classification method w
hich uses linear\nprogramming to construct a decision tree.  Relevant featur
es\nwere selected using an exhaustive search in the space of 1-4\nfeatures a
nd 1-3 separating planes.\n\nThe actual linear program used to obtain the se
parating plane\nin the 3-dimensional space is that described in:\n[K. P. Ben
nett and O. L. Mangasarian: "Robust Linear\nProgramming Discrimination of Tw
o Linearly Inseparable Sets",\nOptimization Methods and Software 1, 1992, 23
-34].\n\nThis database is also available through the UW CS ftp server:\n\nft
p ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-learn/WDBC/\n\n.. topi
c:: References\n\n   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nucle
ar feature extraction \n     for breast tumor diagnosis. IS&T/SPIE 1993 Inte
rnational Symposium on \n     Electronic Imaging: Science and Technology, vo
lume 1905, pages 861-870,\n     San Jose, CA, 1993.\n   - O.L. Mangasarian,
W.N. Street and W.H. Wolberg. Breast cancer diagnosis and \n     prognosis v
ia linear programming. Operations Research, 43(4), pages 570-577, \n     Jul
y-August 1995.\n   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machin
e learning techniques\n     to diagnose breast cancer from fine-needle aspir
ates. Cancer Letters 77 (1994) \n     163-171.',
 'feature_names': array(['mean radius', 'mean texture', 'mean perimeter', 'm
ean 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'], dtype='<U23'),
 'filename': 'C:\\Users\\Dell\\anaconda3\\lib\\site-packages\\sklearn\\datas
ets\\data\\breast_cancer.csv'}
```

Explor the data

In [40]:

```
url = 'https://raw.githubusercontent.com/datasets/covid-19/main/data/us_deaths.csv'
df = pd.read_csv(url)
```

In [41]:

```
df
```

Out[41]:

	Admin2	Date	Case	Country/Region	Province/State
0	Autauga	2020-01-22	0	US	Alabama
1	Autauga	2020-01-23	0	US	Alabama
2	Autauga	2020-01-24	0	US	Alabama
3	Autauga	2020-01-25	0	US	Alabama
4	Autauga	2020-01-26	0	US	Alabama
...	...	...	...	...	...
2727067	Weston	2022-04-12	18	US	Wyoming
2727068	Weston	2022-04-13	18	US	Wyoming
2727069	Weston	2022-04-14	18	US	Wyoming
2727070	Weston	2022-04-15	18	US	Wyoming
2727071	Weston	2022-04-16	18	US	Wyoming

2727072 rows × 5 columns

In [42]:

```
df.shape
```

Out[42]:

```
(2727072, 5)
```

In [44]:

```
f.info()  # show the info matab us ki information dakh rahy hain ak kesam ki us ki type dakh
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2727072 entries, 0 to 2727071
Data columns (total 5 columns):
 #   Column          Dtype 
---  ------          ----- 
 0   Admin2          object
 1   Date            object
 2   Case            int64 
 3   Country/Region  object
 4   Province/State  object
dtypes: int64(1), object(4)
memory usage: 104.0+ MB
```

In [45]:

```
df.Admin2.unique() # show the unique values
```

Out[45]:

```
array(['Autauga', 'Baldwin', 'Barbour', ..., 'Uinta', 'Washakie',
       'Weston'], dtype=object)
```

In [58]:

```
len(df.Admin2.unique()) # show the unique values
```

Out[58]:

```
1981
```

In [47]:

```
df.Date.unique() # show the unique values
```

```
       '2021-01-16', '2021-01-17', '2021-01-18', '2021-01-19',
       '2021-01-20', '2021-01-21', '2021-01-22', '2021-01-23',
       '2021-01-24', '2021-01-25', '2021-01-26', '2021-01-27',
       '2021-01-28', '2021-01-29', '2021-01-30', '2021-01-31',
       '2021-02-01', '2021-02-02', '2021-02-03', '2021-02-04',
       '2021-02-05', '2021-02-06', '2021-02-07', '2021-02-08',
       '2021-02-09', '2021-02-10', '2021-02-11', '2021-02-12',
       '2021-02-13', '2021-02-14', '2021-02-15', '2021-02-16',
       '2021-02-17', '2021-02-18', '2021-02-19', '2021-02-20',
       '2021-02-21', '2021-02-22', '2021-02-23', '2021-02-24',
       '2021-02-25', '2021-02-26', '2021-02-27', '2021-02-28',
       '2021-03-01', '2021-03-02', '2021-03-03', '2021-03-04',
       '2021-03-05', '2021-03-06', '2021-03-07', '2021-03-08',
       '2021-03-09', '2021-03-10', '2021-03-11', '2021-03-12',
       '2021-03-13', '2021-03-14', '2021-03-15', '2021-03-16',
       '2021-03-17', '2021-03-18', '2021-03-19', '2021-03-20',
       '2021-03-21', '2021-03-22', '2021-03-23', '2021-03-24',
       '2021-03-25', '2021-03-26', '2021-03-27', '2021-03-28',
       '2021-03-29', '2021-03-30', '2021-03-31', '2021-04-01',
       '2021-04-02', '2021-04-03', '2021-04-04', '2021-04-05',
```

In [57]:

```
len(df.Date.unique()) # show the unique values
```

Out[57]:

```
816
```

In [49]:

```
df.Case.unique() # show the unique values
```

Out[49]:

```
array([   0,    1,    2, ..., 5798, 5822, 5905], dtype=int64)
```

In [56]:

```
len(df.Case.unique()) # show the unique values
```

Out[56]:

```
8442
```

In [54]:

```
len(df['Country/Region'].unique()) # show the unique values
```

Out[54]:

```
1
```

In [55]:

```
len(df['Province/State'].unique()) # show the unique values
```

Out[55]:

```
58
```

In [53]:

```
df['Province/State'].unique() # show the unique values
```

Out[53]:

```
array(['Alabama', 'Alaska', 'American Samoa', 'Arizona', 'Arkansas',
       'California', 'Colorado', 'Connecticut', 'Delaware',
       'Diamond Princess', 'District of Columbia', 'Florida', 'Georgia',
       'Grand Princess', 'Guam', 'Hawaii', 'Idaho', 'Illinois', 'Indiana',
       'Iowa', 'Kansas', 'Kentucky', 'Louisiana', 'Maine', 'Maryland',
       'Massachusetts', 'Michigan', 'Minnesota', 'Mississippi',
       'Missouri', 'Montana', 'Nebraska', 'Nevada', 'New Hampshire',
       'New Jersey', 'New Mexico', 'New York', 'North Carolina',
       'North Dakota', 'Northern Mariana Islands', 'Ohio', 'Oklahoma',
       'Oregon', 'Pennsylvania', 'Puerto Rico', 'Rhode Island',
       'South Carolina', 'South Dakota', 'Tennessee', 'Texas', 'Utah',
       'Vermont', 'Virgin Islands', 'Virginia', 'Washington',
       'West Virginia', 'Wisconsin', 'Wyoming'], dtype=object)
```

In [50]:

```
df.isna().sum()  # show the null values
```

Out[50]:

```
Admin2            4896
Date                 0
Case                 0
Country/Region       0
Province/State       0
dtype: int64
```

In [ ]:

```

```