Numpy

In [56]:

```
import numpy as np
```

arange

In [57]:

```
linear_array = np.arange(2,100,7)
linear_array
```

Out[57]:

```
array([ 2,  9, 16, 23, 30, 37, 44, 51, 58, 65, 72, 79, 86, 93])
```

rand

In [58]:

```
rand_matrix = np.random.rand(3, 5)
rand_matrix
```

Out[58]:

```
array([[0.09645495, 0.54832516, 0.37579172, 0.24206672, 0.7980458 ],
       [0.11548048, 0.05029228, 0.60230654, 0.47101472, 0.37439272],
       [0.62830997, 0.77810449, 0.02028247, 0.96240307, 0.16066107]])
```

Базові операції

In [59]:

```
a = np.arange(6)
a
```

Out[59]:

```
array([0, 1, 2, 3, 4, 5])
```

In [60]:

```
a+11
```

Out[60]:

```
array([11, 12, 13, 14, 15, 16])
```

In [61]:

```
b = np.arange(3,9)
b
```

Out[61]:

```
array([3, 4, 5, 6, 7, 8])
```

In [62]:

```
a+b
```

Out[62]:

```
array([ 3,  5,  7,  9, 11, 13])
```

In [63]:

```
a-b
```

Out[63]:

```
array([-3, -3, -3, -3, -3, -3])
```

In [64]:

```
a*b
```

Out[64]:

```
array([ 0,  4, 10, 18, 28, 40])
```

In [65]:

```
A = np.arange(0, 12).reshape(3, 4)
A
```

Out[65]:

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [66]:

```
B = np.ones((4, 3))
B
```

Out[66]:

```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

In [67]:

```
C=A.dot(B)
C
```

Out[67]:

```
array([[ 6.,  6.,  6.],
       [22., 22., 22.],
       [38., 38., 38.]])
```

In [68]:

```
D=B.dot(A)
D
```

Out[68]:

```
array([[12., 15., 18., 21.],
       [12., 15., 18., 21.],
       [12., 15., 18., 21.],
       [12., 15., 18., 21.]])
```

Pandas

In [69]:

```
import pandas as pd
```

In [70]:

```
df1 = pd.DataFrame({'x': [1, 2, 3], 'y': [5, 5, 11]})
df1
```

Out[70]:

	x	y
0	1	5
1	2	5
2	3	11

In [71]:

```
df2 = pd.DataFrame([[1, 5],[2, 5], [3, 11]],columns=['x', 'y'])
df2
```

Out[71]:

	x	y
0	1	5
1	2	5
2	3	11

In [72]:

```
df1.head(2)
```

Out[72]:

	x	y
0	1	5
1	2	5

In [73]:

```
df2.describe()
```

Out[73]:

	x	y
count	3.0	3.000000
mean	2.0	7.000000
std	1.0	3.464102
min	1.0	5.000000
25%	1.5	5.000000
50%	2.0	5.000000
75%	2.5	8.000000
max	3.0	11.000000

In [74]:

```
df1.iloc[:,0]
```

Out[74]:

```
0    1
1    2
2    3
Name: x, dtype: int64
```

In [75]:

```
df2.loc[:,'x']
```

Out[75]:

```
0    1
1    2
2    3
Name: x, dtype: int64
```

In [76]:

```
df1.sort_values(by=['y'])
```

Out[76]:

	x	y
0	1	5
1	2	5
2	3	11

Matplotlib

In [77]:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

In [78]:

```
x = np.random.uniform(-10, 10, size=50)
```

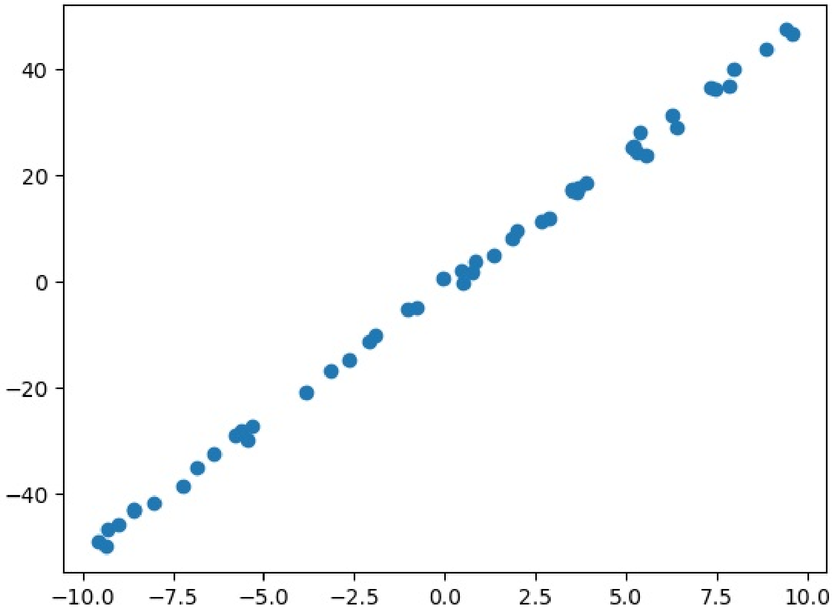
In [79]:

```
noise = np.random.normal(0, 1, size=50)
```

```
In [80]: y = 5 * x - 1 + noise
```

```
In [81]: plt.scatter(x, y)
```

```
Out[81]: <matplotlib.collections.PathCollection at 0x1a312ce53d0>
```



```
In [82]: mae = mean_absolute_error(x, y)
```

```
In [83]: mse = mean_squared_error(x, y)
```

```
In [ ]: import csv
```

```
In [85]: with open('info.csv', "w", newline="") as file:
           writer = csv.writer(file)
           writer.writerow(["Name", "Value"])
           writer.writerows([
                             ["x", x],
                             ["y", y],
                             ["mae", mae],
                             ["mse", mse]
           ])
```

```
In [ ]: 
```

Sklearn

Лінійна регресія

```
In [86]: from sklearn.linear_model import LinearRegression
```

```
In [87]: reg = LinearRegression().fit(x.reshape(-1, 1), y.reshape(-1, 1))
```

```
In [88]: reg.score(x.reshape(-1, 1), y.reshape(-1, 1))
```

```
Out[88]: 0.9986597661207988
```

```
In [89]: from sklearn.tree import DecisionTreeClassifier
```

```
In [90]: from sklearn.datasets import load_iris
```

```
In [91]: iris=load_iris()
```

```
In [92]: x=iris.data
         y=iris.target
```

```
In [93]: dt = DecisionTreeClassifier()
```

```
In [94]: from sklearn.model_selection import train_test_split
         x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.5)
```

```
In [95]: dt.fit(x_train,y_train)
```

```
Out[95]: DecisionTreeClassifier()
```

```
In [96]: result = dt.predict(x_test)
```

```
In [97]: result
```

```
Out[97]: array([1, 1, 2, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 1, 0, 2, 1, 2, 1, 2, 2,
                2, 2, 1, 0, 1, 1, 2, 0, 1, 0, 1, 2, 0, 2, 1, 1, 2, 2, 2, 1, 0, 2,
                1, 0, 1, 2, 1, 0, 1, 2, 0, 0, 1, 2, 0, 1, 2, 2, 0, 1, 2, 2, 1, 1,
                0, 0, 1, 1, 0, 1, 1, 1, 2])
```

```
In [98]: from sklearn.metrics import accuracy_score
```

```
In [99]: accuracy_score(y_test, result)
```

```
Out[99]: 0.96
```

```
In [ ]:
```

Tensorflow

```
In [100… import tensorflow as tf
```

```
In [101… data = tf.Variable([[4], [9], [16], [25]], tf.int32)
```

```
In [102… data.shape[1]
```

```
Out[102]: 1
```

```
In [103… data
```

```
Out[103]: <tf.Variable 'Variable:0' shape=(4, 1) dtype=int32, numpy=
          array([[ 4],
                 [ 9],
                 [16],
                 [25]])>
```

```
In [104… tf.reshape(data, [1,4])
```

```
Out[104]: <tf.Tensor: shape=(1, 4), dtype=int32, numpy=array([[ 4,  9, 16, 25]])>
```

```
In [105… np_data = np.array(data)
         np_data
```

```
Out[105]: array([[ 4],
                 [ 9],
                 [16],
                 [25]])
```

```
In [106… data[0]
```

```
Out[106]: <tf.Tensor: shape=(1,), dtype=int32, numpy=array([4])>
```

```
In [107… import pandas as pd
         df1 = pd.DataFrame({'x': [1, 2, 3], 'y': [5, 5, 11]})
```

```
In [108… ds = tf.data.Dataset.from_tensor_slices(dict(df1))
```

```
In [109… ds
```

```
Out[109]: <_TensorSliceDataset element_spec={'x': TensorSpec(shape=(), dtype=tf.int64, name=None), 'y': TensorSpec(shape
          =(), dtype=tf.int64, name=None)}>
```

```
In [ ]:
```

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js