Analytics Vidhya

NumPy Basics

Prashant Sahu

Manager Data Science, Analytics Vidhya

NumPy: The Foundation of Python for Data Science

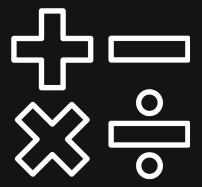
- High-performance array operations
- Fundamental package for scientific computing in Python
- Widely used in data science, machine learning, and numerical analysis

“NumPy stands for ‘Numerical Python,’ and it’s built around the ndarray (n-dimensional array) data structure.

What is NumPy & Why Do We Need It?

- ndarray object: efficient multi-dimensional arrays
- Vectorized operations for speed and simplicity
- Replaces slow Python loops with optimized C-based routines
- Backbone for libraries like Pandas, SciPy, and scikit-learn
- Core numerical engine for data manipulation and analysis

Arithmetic Universal Functions



Arithmetic Operations

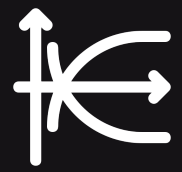
np.add, np.subtract, np.multiply, np.divide

Exponential

np.exp(x)

Exponentiation

np.power(x, y)

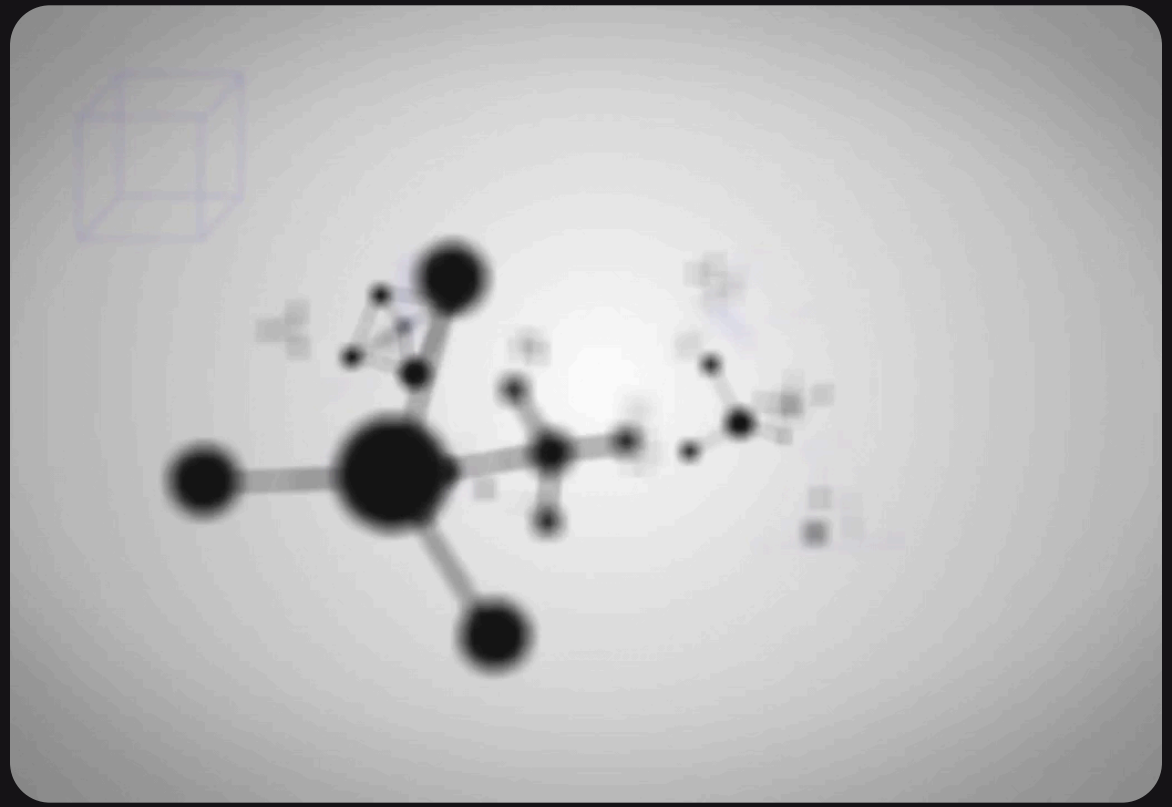


Logarithms

np.log(x), np.log2(x), np.log10(x)

Trigonometric & Hyperbolic UFuncs

- **Trigonometric**
  np.sin(x), np.cos(x), np.tan(x)
- **Inverse trig**
  np.arcsin(x), np.arccos(x), np.arctan(x)
- **Hyperbolic**
  np.sinh(x), np.cosh(x), np.tanh(x)
- **Inverse hyperbolic**
  np.arcsinh(x), np.arccosh(x), np.arctanh(x)



Rounding & Other Element-wise Numeric UFuncs

NumPy Function	What is does?	Examples	Output
np.round(x)	Round off to nearest integer	np.round(2.81) np.round(-5.67)	3 -6
np.floor(x)	Round off to lower integer	np.floor(2.81) np.floor(-5.67)	2 -6
np.ceil(x)	Round off to a higher integer	np.ceil(2.81) np.ceil(-5.67)	3 -5
np.abs(x)	Returns the absolute value of a number	np.abs(2.81) np.abs(-5.67)	2.81 5.67
np.sign(x)	Returns the indication of the sign of a number	np.sign(2.81) np.sign(-5.67)	1 -1

Rounding

np.round(x), np.floor(x), np.ceil(x), np.trunc(x)

Absolute & Sign

np.abs(x), np.sign(x)

Comparison UFuncs

- **Relational**

  np.less(), np.less_equal(), np.greater(), np.greater_equal(), np.equal(), np.not_equal()

- **Logical**

  np.logical_and(), np.logical_or(), np.logical_xor()

- **Bitwise**

  np.bitwise_and(), np.bitwise_or(), np.bitwise_xor()

Statistical & Aggregation Functions

- **Basic stats**

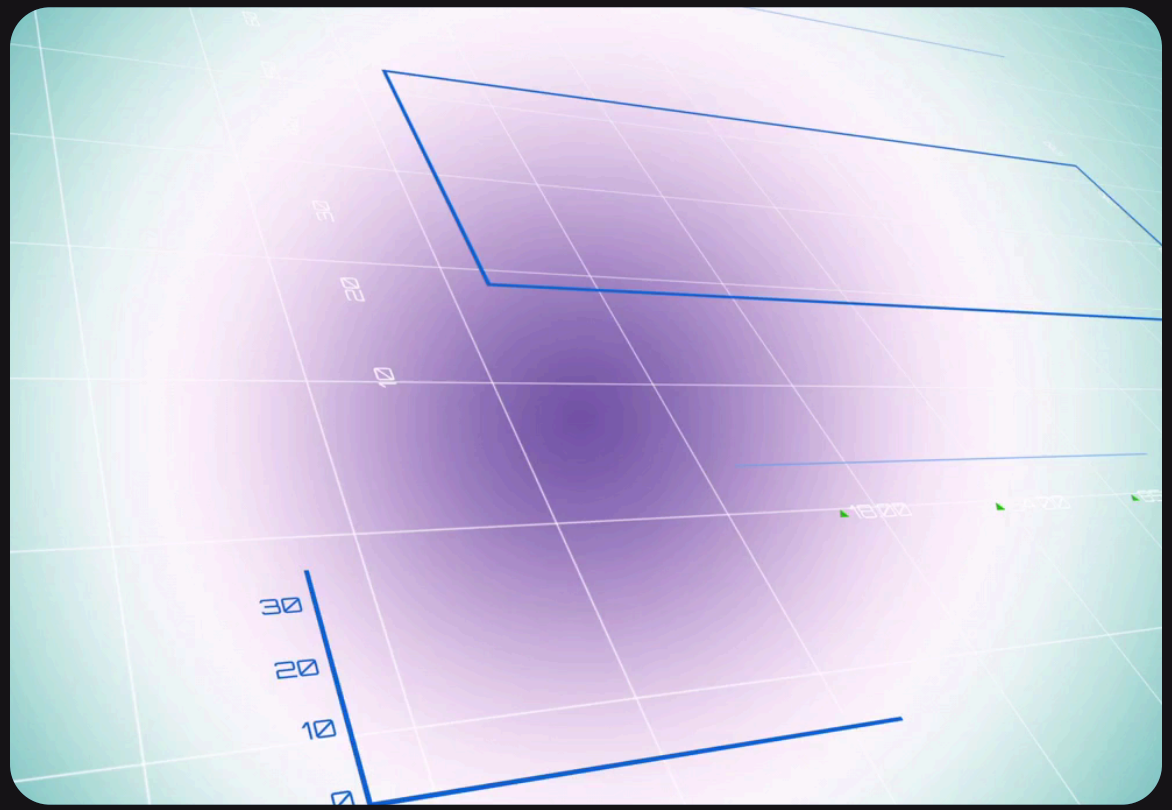
  np.sum(x), np.mean(x), np.std(x), np.var(x)

- **Min/Max**

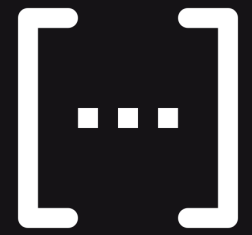
  np.min(x), np.max(x), np.argmin(x), np.argmax(x)

- **Others**

  np.median(x), np.quantile(x, q)

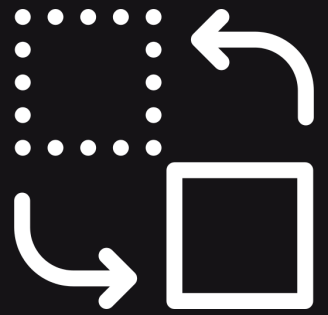


NumPy Core



Array creation

np.array(), np.zeros(), np.ones(),
np.arange(), np.linspace()



Shape manipulation

reshape(), ravel(), transpose()

Data types

dtype, .astype()

Broadcasting in Detail with Examples

- **Definition**
  Automatic expansion of array dimensions
- **Rules**
  Matching shapes from the trailing dimensions
- **Examples**
  - Adding a scalar to an array
  - Adding arrays with compatible shapes

```
import numpy as np


A = np.array([1, 2, 3])
B = np.array([[10], [20], [30]])


print(A + B)
```

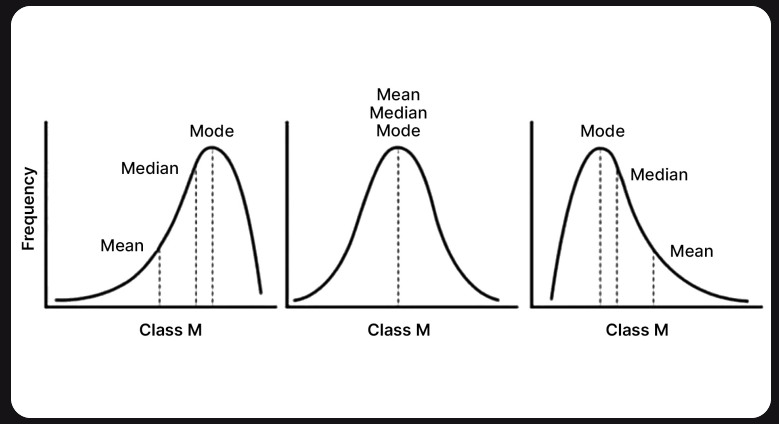
NumPy Linalg

- Matrix and vector products
  np.dot, np.matmul, @ operator
- Solving linear equations
  np.linalg.solve(A, b)
- Inversion & Determinant
  - np.linalg.inv(A), np.linalg.det(A)

NumPy Linalg Module

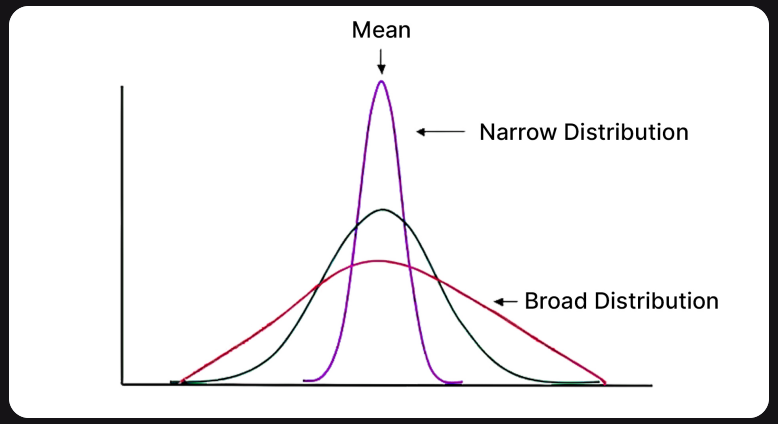
- Matrix and vector products
  np.dot, np.matmul, @ operator
- Solving linear equations
  np.linalg.solve(A, b)
- Inversion & Determinant
  - np.linalg.inv(A), np.linalg.det(A)
- Eigenvalues & Eigenvectors
  np.linalg.eig(A)
- Singular Value Decomposition
  np.linalg.svd(A)
- Norms
  - np.linalg.norm(A, ord)

Statistical Functions



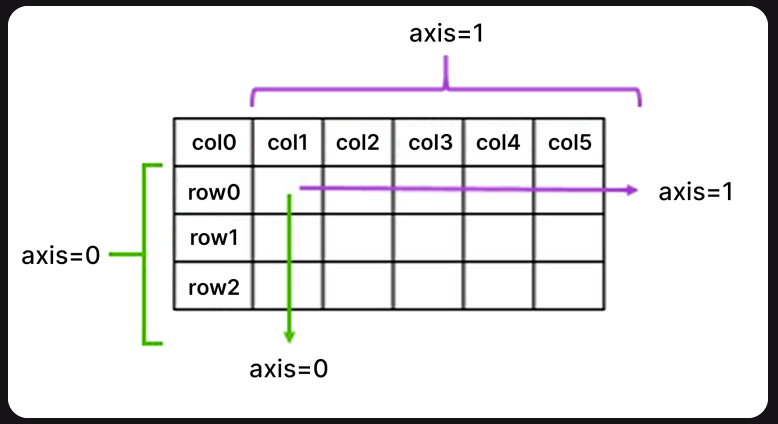
Measures of central tendency

np.mean(), np.median(), np.quantile()



Measures of spread

np.std(), np.var()

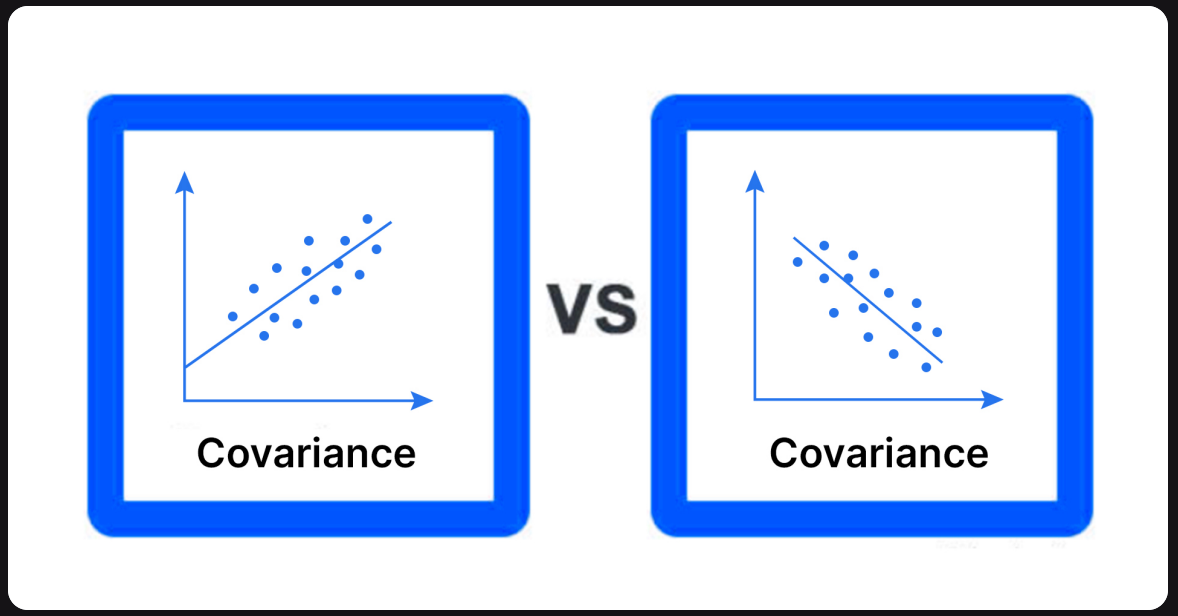


Axis parameter

Compute along rows or columns

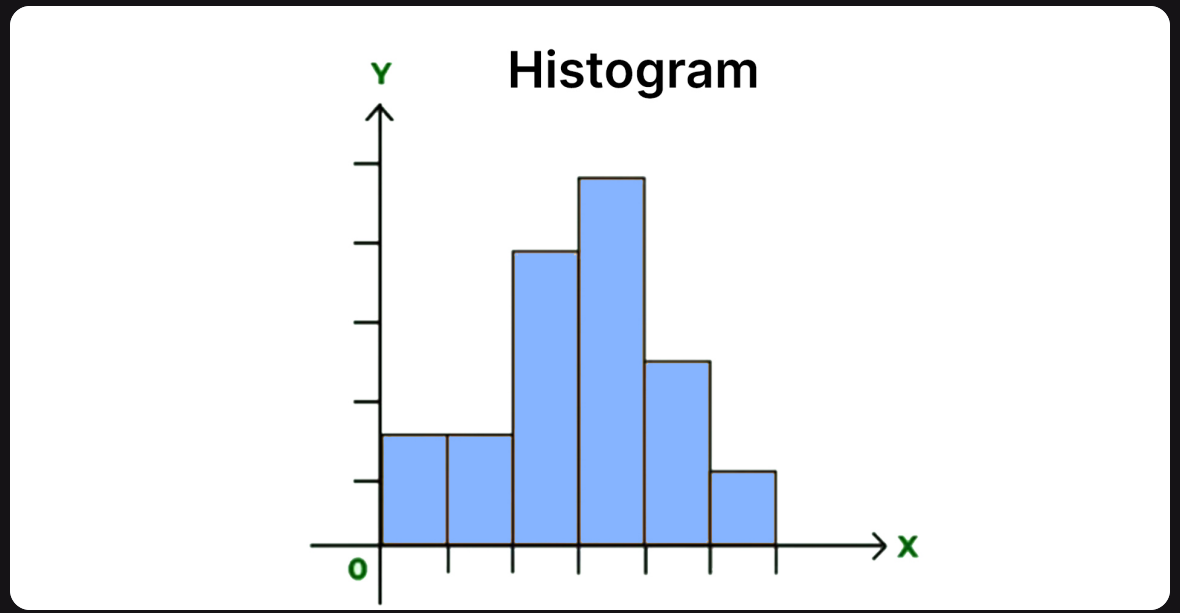
Analytics Vidhya

Statistical Functions



Correlation & Covariance

np.corrcoef(), np.cov()



Histogram

np.histogram()

Use cases: Quick data exploration & summarization

Analytics Vidhya

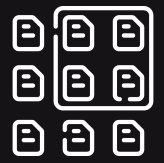
NumPy Random Module

Random number generation

np.random.default_rng(seed=...)

Distributions

normal, uniform, binomial, etc.



Sampling

rng.integers(), rng.random(), rng.normal()

Shuffling & permutations

rng.shuffle(), rng.permutation()

Use cases: Reproducible experiments, Data augmentation, Monte Carlo

Thanks