

NumPy Cheat Sheet

Topic	Cheat Codes / Notes
Setup	import numpy as np
Creating Arrays	# From list a = np.array([1, 2, 3]) # 1D A = np.array([[1, 2, 3], [4, 5, 6]]) # 2D # Zeros / Ones Z = np.zeros((3, 3)) # 3×3 O = np.ones((2, 4)) # 2×4 # Range-like r1 = np.arange(0, 10, 2) # 0 2 4 6 8 r2 = np.linspace(0, 1, 5) # 5 points 0→1
Basic Properties	X.shape # (rows, cols) X.ndim # dims X.dtype # type X.size # num elements X = np.array([[1, 2, 3], [4, 5, 6]]) # X.shape -> (2, 3), X.ndim -> 2
Indexing & Slicing (House Data)	data = np.array([[800,2,15,75000], [950,2,10,90000], [1200,3,8,120000], [1500,3,5,155000]]) # Elements data[0,0] # 800 data[2,1] # 3 data[-1,-1] # 155000 # Rows / Cols data[0,:] # first row data[:,1] # second col # Slicing data[:2,:] # first 2 rows data[1:4,0:2] # rows 1-3, cols 0-1
Features & Targets (ML Style)	X = data[:, :3] # features y = data[:, 3] # target X.shape -> (4, 3) y.shape -> (4,)
Reshape & Flatten	a = np.array([1,2,3,4,5,6]) a_2x3 = a.reshape(2,3) # [[1 2 3],[4 5 6]] a_flat = a_2x3.flatten() # [1 2 3 4 5 6]
Stacking Arrays	A = np.array([[1,2],[3,4]]) B = np.array([[5,6],[7,8]]) v = np.vstack((A,B)) # [[1 2],[3 4],[5 6],[7 8]] h = np.hstack((A,B)) # [[1 2 5 6],[3 4 7 8]]
Scalar Ops	X = np.array([10,20,30,40]) X - 10 # [0 10 20 30] X / 10 # [1. 2. 3. 4.]
Elementwise Ops	a = np.array([1,2,3]) b = np.array([10,20,30]) a + b # [11 22 33]

	<pre> a * b # [10 40 90] a / b # [0.1 0.1 0.1] # Operation per element (same index). </pre>
Statistics (for ML)	<pre> data.mean() # overall mean data.std() # overall std data.mean(axis=0) # mean per column data.min(axis=0) # min per column data.max(axis=0) # max per column # data.mean(axis=0)[3] = avg price </pre>
Linear Algebra	<pre> # Dot product w = np.array([1,2,3]) x = np.array([4,5,6]) np.dot(w,x) # 32 w @ x # 32 # Matrix · vector (simple model) X = np.array([[1,2,3],[4,5,6]]) w = np.array([0.1,0.2,0.3]) b = 0.5 y_pred = X @ w + b # [1.9, 3.7] </pre>
Transpose & Sum	<pre> A = np.array([[1,2,3],[4,5,6]]) A.T # [[1,4],[2,5],[3,6]] A.sum(axis=0) # [5,7,9] (cols) A.sum(axis=1) # [6,15] (rows) </pre>
Boolean Indexing & Shuffling	<pre> prices = np.array([75000,90000,120000,155000,60000]) mask = prices > 100000 expensive = prices[mask] # [120000,155000] expensive.size # 2 # Shuffle dataset X = data[:, :3] y = data[:, 3] idx = np.random.permutation(len(X)) X_shuffled = X[idx] y_shuffled = y[idx] </pre>