Here's a compact **NumPy Cheat Sheet** tailored for **Data Analysis**:

□ NumPy Cheat Sheet for Data Analysis

📦 Importing

```
import numpy as np
```

🛠️ Array Creation

```
np.array([1, 2, 3])          # From list
np.zeros((2, 3))             # 2x3 zeros
np.ones((3, 3))              # 3x3 ones
np.full((2, 2), 7)           # Fill with 7s
np.eye(3)                    # 3x3 Identity matrix
np.arange(0, 10, 2)          # [0, 2, 4, 6, 8]
np.linspace(0, 1, 5)         # 5 values between 0 and 1
```

🎲 Random Numbers

```
np.random.rand(2, 3)         # Uniform [0, 1)
np.random.randn(3, 3)        # Standard normal
np.random.randint(1, 10, (2, 2)) # Random ints
```

```
np.random.seed(42)          # Reproducibility
```

🔍 Array Inspection

```
a.shape                     # Shape
a.ndim                      # Dimensions
a.size                      # Total elements
a.dtype                     # Data type
```

🔁 Indexing & Slicing

```
a[0, 1]                     # Access element
a[:, 1]                     # Column slice
a[1:3, :]                   # Row slice
a[a > 5]                    # Boolean filter
a[[0, 2], [1, 3]]           # Fancy indexing
```

🔧 Array Operations

```
a + b                       # Element-wise addition
a * b                       # Multiplication
a @ b                       # Matrix multiplication (dot)
a ** 2                      # Power
```

```
a.T                    # Transpose
```

📈 Math & Stats

```
np.sum(a), np.mean(a)

np.std(a), np.var(a)

np.min(a), np.max(a)

np.median(a)

np.cumsum(a), np.cumprod(a)

np.percentile(a, 50)
```

🔄 Reshaping

```
a.reshape(3, 2)          # New shape

a.flatten()              # 1D view

a.ravel()                # Flattened view
```

🔗 Combining & Splitting

```
np.concatenate([a, b], axis=0)

np.vstack((a, b))

np.hstack((a, b))
```

```
np.split(a, 2)              # Split into 2
np.hsplit(a, 2)
```

📊 Linear Algebra

```
np.dot(a, b)              # Dot product
np.linalg.inv(a)          # Inverse
np.linalg.det(a)          # Determinant
np.linalg.eig(a)          # Eigenvalues/vectors
np.linalg.solve(a, b)     # Solve Ax = b
```

□ Missing/Invalid Data

```
np.isnan(a), np.isinf(a)
np.nan_to_num(a)
np.where(np.isnan(a), 0, a)
```

💾 Save/Load

```
np.save('data.npy', a)
np.load('data.npy')

np.savetxt('data.csv', a, delimiter=',')
```

```
np.loadtxt('data.csv', delimiter=',')
```
