Assignment 2 - Numpy Array Operations

This assignment is part of the course "Data Analysis with Python: Zero to Pandas". The objective of this assignment is to develop a solid understanding of Numpy array operations. In this assignment you will:

1. Pick 5 interesting Numpy array functions by going through the documentation: https://numpy.org/doc/stable/reference/routines.html
2. Run and modify this Jupyter notebook to illustrate their usage (some explanation and 3 examples for each function). Use your imagination to come up with interesting and unique examples.
3. Upload this notebook to your Jovian profile using `jovian.commit` and make a submission here: https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas/assignment/assignment-2-numpy-array-operations
4. (Optional) Share your notebook online (on Twitter, LinkedIn, Facebook) and on the community forum thread: https://jovian.ml/forum/t/assignment-2-numpy-array-operations-share-your-work/10575 .
5. (Optional) Check out the notebooks shared by other participants and give feedback & appreciation.

The recommended way to run this notebook is to click the "Run" button at the top of this page, and select "Run on Binder". This will run the notebook on mybinder.org, a free online service for running Jupyter notebooks.

Try to give your notebook a catchy title & subtitle e.g. "All about Numpy array operations", "5 Numpy functions you didn't know you needed", "A beginner's guide to broadcasting in Numpy", "Interesting ways to create Numpy arrays", "Trigonometic functions in Numpy", "How to use Python for Linear Algebra" etc.

NOTE: Remove this block of explanation text before submitting or sharing your notebook online - to make it more presentable.

5 numpy functions I learnt

numpy is a library which contains lots of inbuilt functions which help in the calculation and hence, is an important for data analysis. The functions which i chose are:

- fabs
- sort
- bitwise_and
- stack
- argmax

The recommended way to run this notebook is to click the "Run" button at the top of this page, and select "Run on Binder". This will run the notebook on mybinder.org, a free online service for running Jupyter notebooks.

```
!pip install jovian --upgrade -q
```

```
import jovian
```

```
jovian.commit(project='numpy-array-operations')
```

```
[jovian] Updating notebook "ahmedatif655/numpy-array-operations" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/ahmedatif655/numpy-array-operations
```

```
'https://jovian.ai/ahmedatif655/numpy-array-operations'
```

Let's begin by importing Numpy and listing out the functions covered in this notebook.

```
import numpy as np
```

```
function1 = np.fabs
function2 = np.sort
function3 = np.bitwise_and
function4 = np.stack
function5 = np.argmax
```

Function 1 - np.fabs

This function converts the negative values in the data to positive.It fails when data contains complex values.

```
# Example 1 - working (change this)
arr1 = [[-1, -2],
        [3, -4.]]

np.fabs(arr1)
```

```
array([[1., 2.],
       [3., 4.]])
```

In the above example,we can see that all the negative values has been converted to positive.

```
# Example 2 - working
arr2 = [[5.87, -6.5, 7.7],
        [8.5, 9.6, -10.8]]

np.fabs(arr2)
```

```
array([[ 5.87,  6.5 ,  7.7 ],
       [ 8.5 ,  9.6 , 10.8 ]])
```

In the above example, we can see that fabs function work with float values also.

```
arr3 = [[5, -6, 7],
        [8, 9, complex(-10, 3)]]

np.fabs(arr3)
```

```
---------------------------------------------------------------------------
```

```
TypeError                                 Traceback (most recent call last)
/tmp/ipykernel_485/3063145479.py in <module>
      2          [8, 9, complex(-10, 3)]]
      3 
----> 4 np.fabs(arr3)

TypeError: ufunc 'fabs' not supported for the input types, and the inputs could not be
safely coerced to any supported types according to the casting rule ''safe''
```

we can see that fabs doesn't work with complex number.We can use absolute to find absolute values of complex data.

```
jovian.commit()
```

```
[jovian] Updating notebook "ahmedatif655/numpy-array-operations" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/ahmedatif655/numpy-array-operations

'https://jovian.ai/ahmedatif655/numpy-array-operations'
```

Function 2 - np.sort

It returns a sorted copy of an array.

```
# Example 1 - working
z = np.array([[1,4],[3,1]])
np.sort(z)
```

```
array([[1, 4],
       [1, 3]])
```

Above example shows the working of sort function.It sorts the array along the default axis.

```
# Example 2 - working
z = np.array([[1,4],[3,1]])
np.sort(z, axis=0)
```

```
array([[1, 1],
       [3, 4]])
```

In the above example, the array is sorted along the first axis.

```
# Example 3 - breaking (to illustrate when it breaks)
z = np.array([[1,4],[3,1]])
np.sort(z, axis=3)
```

```
---------------------------------------------------------------------------
AxisError                                 Traceback (most recent call last)
/tmp/ipykernel_485/2234985819.py in <module>
      1 # Example 3 - breaking (to illustrate when it breaks)
      2 z = np.array([[1,4],[3,1]])
----> 3 np.sort(z, axis=3)
```

```
<__array_function__ internals> in sort(*args, **kwargs)

/opt/conda/lib/python3.9/site-packages/numpy/core/fromnumeric.py in sort(a, axis, kind,
order)
    994     else:
    995         a = asanyarray(a).copy(order="K")
--> 996     a.sort(axis=axis, kind=kind, order=order)
    997     return a
    998

AxisError: axis 3 is out of bounds for array of dimension 2
```

In the above example,the axis mentioned is out of bounds.

```
jovian.commit()
```

```
[jovian] Updating notebook "ahmedatif655/numpy-array-operations" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/ahmedatif655/numpy-array-operations
```

```
'https://jovian.ai/ahmedatif655/numpy-array-operations'
```

Function 3 - np.bitwise_and

Compute the bitwise and of two arrays element-wise.

```
# Example 1 - working
np.bitwise_and(13, 17)
```

```
1
```

13 = 00001101, 17 = 00010001.The bitwise AND of 13 and 17 is 000000001, which represents 1.

```
# Example 2 - working
x = np.array([2,5,255])
y = np.array([3,14,16])
np.bitwise_and(x, y)
```

```
array([ 2,  4, 16])
```

Explanation about example

```
# Example 3 - breaking (to illustrate when it breaks)
x = np.array([2,255])
y = np.array([3,14,16])
x&y
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_485/1522002833.py in <module>
      2 x = np.array([2,255])
      3 y = np.array([3,14,16])
----> 4 x&y

ValueError: operands could not be broadcast together with shapes (2,) (3,)
```

As can be seen from the error, the size of two operands are different and bitwise and can't be performed.

```
jovian.commit()
```

```
[jovian] Updating notebook "ahmedatif655/numpy-array-operations" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/ahmedatif655/numpy-array-operations

'https://jovian.ai/ahmedatif655/numpy-array-operations'
```

Function 4 - np.stack

It joins the sequences of arrays along a new axis.

```
# Example 1 - working
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
np.stack((a,b))
```

```
array([[1, 2, 3],
       [4, 5, 6]])
```

In the above example, array b is joined to a along axis 0.

```
# Example 2 - working
np.stack((a, b), axis=-1)
```

```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

In the above example, a and b are joined along axis -1 i.e., last dimension.

```
# Example 3 - breaking (to illustrate when it breaks)
np.stack((a, b), axis=5)
```

```
---------------------------------------------------------------------------
AxisError                                 Traceback (most recent call last)
/tmp/ipykernel_485/2570501719.py in <module>
      1 # Example 3 - breaking (to illustrate when it breaks)
----> 2 np.stack((a, b), axis=5)

<__array_function__ internals> in stack(*args, **kwargs)

/opt/conda/lib/python3.9/site-packages/numpy/core/shape_base.py in stack(arrays, axis,
out)
    428
    429     result_ndim = arrays[0].ndim + 1
--> 430     axis = normalize_axis_index(axis, result_ndim)
    431 
    432     sl = (slice(None),) * axis + (_nx.newaxis,)

AxisError: axis 5 is out of bounds for array of dimension 2
```

It fails to join the two arrays as axis is out of bound.It can be resolved by replacing axis with 0 or -1.

```
jovian.commit()
```

```
[jovian] Updating notebook "ahmedatif655/numpy-array-operations" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/ahmedatif655/numpy-array-operations
```

```
'https://jovian.ai/ahmedatif655/numpy-array-operations'
```

Function 5 - np.full

It returns a new array of given shape and type,filled with fill_value.

```
# Example 1 - working
np.full((2,2),fill_value='inf')
```

```
array([['inf', 'inf'],
       ['inf', 'inf']], dtype='<U3')
```

In the above example,we can see that an array of size (2,2) is created and filled with value 'inf'.

```
# Example 2 - working
np.full((2,2),[1,2])
```

```
array([[1, 2],
       [1, 2]])
```

An array of size (2, 2) is created and filled with value 1 and 2.

```
# Example 3 - breaking (to illustrate when it breaks)
np.full((1,3),(2,2))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_485/2869518630.py in <module>
      1 # Example 3 - breaking (to illustrate when it breaks)
----> 2 np.full((1,3),(2,2))

/opt/conda/lib/python3.9/site-packages/numpy/core/numeric.py in full(shape, fill_value,
dtype, order, like)
    341         dtype = fill_value.dtype
    342     a = empty(shape, dtype, order)
--> 343     multiarray.copyto(a, fill_value, casting='unsafe')
    344     return a
    345

<__array_function__ internals> in copyto(*args, **kwargs)

ValueError: could not broadcast input array from shape (2,) into shape (1,3)
```

As can be seen from the error, if there is size difference between shape and fill_value,error occurs.

```
jovian.commit()
```

```
[jovian] Updating notebook "ahmedatif655/numpy-array-operations" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/ahmedatif655/numpy-array-operations
```

```
'https://jovian.ai/ahmedatif655/numpy-array-operations'
```

Conclusion

Summarize what was covered in this notebook, and where to go next

Reference Links

Provide links to your references and other interesting articles about Numpy arrays:

- Numpy official tutorial : https://numpy.org/doc/stable/user/quickstart.html
- ...

```
jovian.commit()
```

```

```