Assignment: Difference between **None**, **np.inf** and **np.nan**

np.inf, **np.nan**, and **none** are all special values in Python and NumPy with different meanings and use cases.

1. np.inf (Infinity):

- np.inf is a representation of positive infinity, indicating a value greater than any finite number.
- In NumPy, it is employed for mathematical operations involving infinity, like handling cases of division by zero.
- **Example:**

```
import numpy as np
a = 10
b = 0
result = a / b
```

This results in np.inf

2. np.nan (Not-a-Number):

- np.nan is used in NumPy to signify missing or undefined data within arrays, signifying that a particular value is either not available or meaningful in a given context.
- It's important to note that np.nan exhibits a contagious behavior; any operation involving np.nan results in the outcome being np.nan.
- **Example:**

```
import numpy as np
dataset = np.array([3.0, 1.5, np.nan, 2.5, 4.0])
result = np.sum(dataset)
```

This results in np.nan, as the sum is undefined with NaN values.

3. None:

- None is a fundamental Python object denoting the absence of a value or a null value.
- While it is primarily used in Python and not NumPy, it serves the purpose of indicating the non-existence of a value or functioning as a placeholder for uninitialized variables.
- **Example:**

```
variable = None
if variable is None:
print("The variable is not defined")
```