Rounding Decimals

There are primarily five ways of rounding off decimals in Numpy:

- truncation
- fix
- rounding
- floor
- ceil

Truncation

Remove the decimals, and return the float number closest to zero. Use the trunc() & fix() function.

e.g

```
import numpy as np
arr = np.trunc([-3.1666, 3.6667])
print(arr)
```

O/P

[-3. 3.]

e.g Same example, using fix():

```
import numpy as np
arr = np.fix([-3.1666, 3.6667])
print(arr)
```

O/P

[-3. 3.]

Rounding

The around() function increments preceding digit or decimal by 1 if >=5 else do nothing.

e.g Round off 3.1666 to 2 decimal places:

```
import numpy as np
arr = np.around(3.1666, 2)
print(arr)
```

O/P

3.17

Floor

The floor() function rounds off decimal to nearest lower integer.

e.g Floor of 3.166 is 3

```
import numpy as np
arr = np.floor([-3.1666, 3.6667])
print(arr)
```

O/P

```
[-4. 3.]
```

* The floor() function returns floats, unlike the trunc() function who returns integers.

Ceil

The ceil() function rounds off decimal to nearest upper integer.

e.g ceil of 3.166 is 4

ceil the elements of following array.

```
import numpy as np
arr = np.ceil([-3.1666, 3.6667])
print(arr)
```

O/P

```
[-3. 4.]
```

Numpy Logs

Log

Numpy provides functions to perform log at the base 2, e and 10.

Log at Base 2

Use the log2() function to perform log at the base 2

```
import numpy as np
arr = np.arange(1,10)
print(np.log2(arr))
```

```
[0.  1.  1.5849625  2.  2.32  2.5
2.
```

Log at Base10

Use the log10() function.

```
import numpy as np
arr = np.arange(1,10)
print(np.log10(arr))
```

O/P

[0 0.30103- ----]

~~Natural Log, or Log at Base e~~

Use the log() function to perform log at the base e.

```
import numpy as np
arr = np.arange(1,10)
print(np.log(arr))
```

Log at Any Base

```
from math import log
import numpy as np
nplog = np.frompyfunc(log, 2, 1)
print(nplog(100, 15))
```

O/P

1.7005483074552052