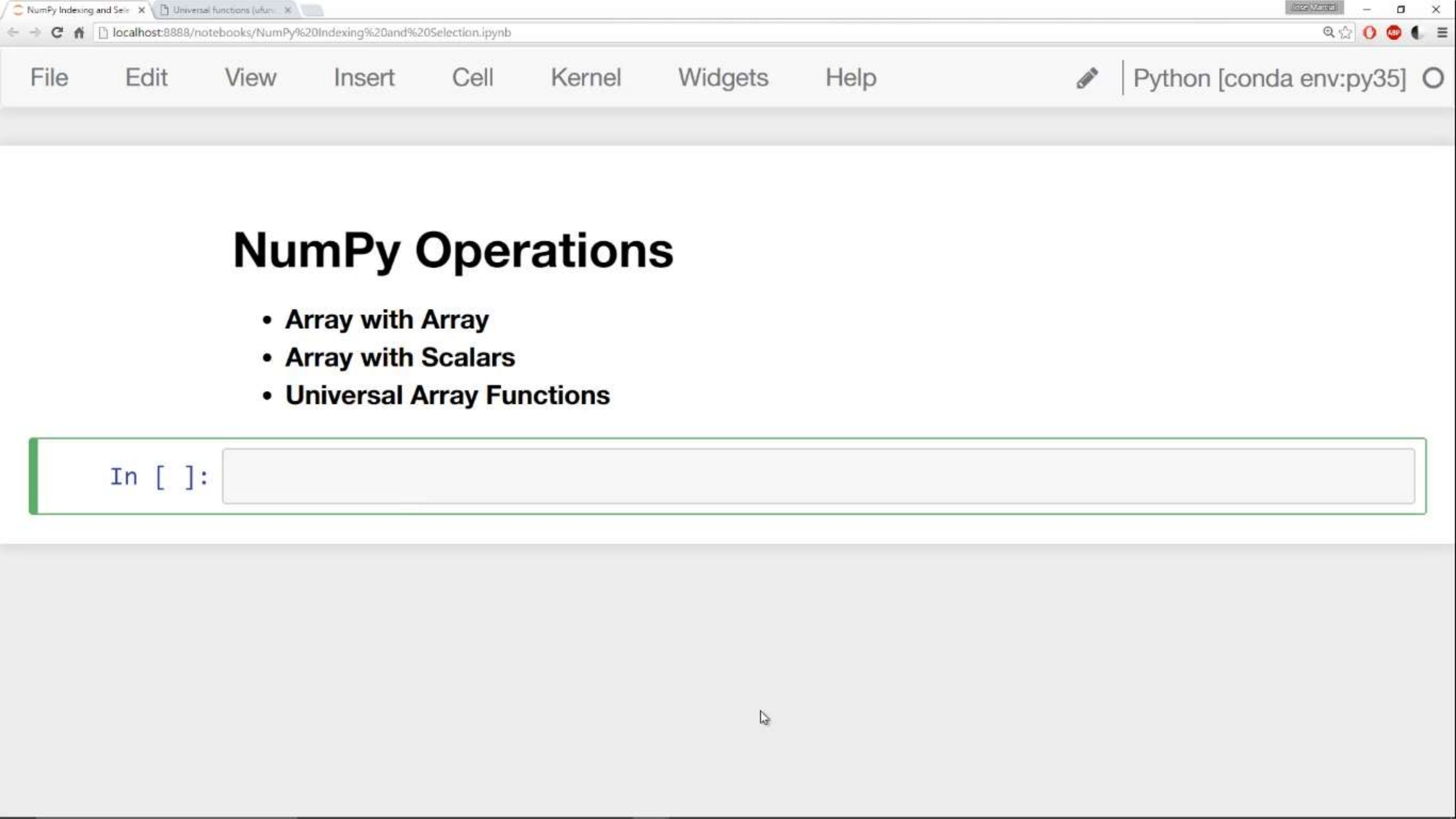
Python for Data Science

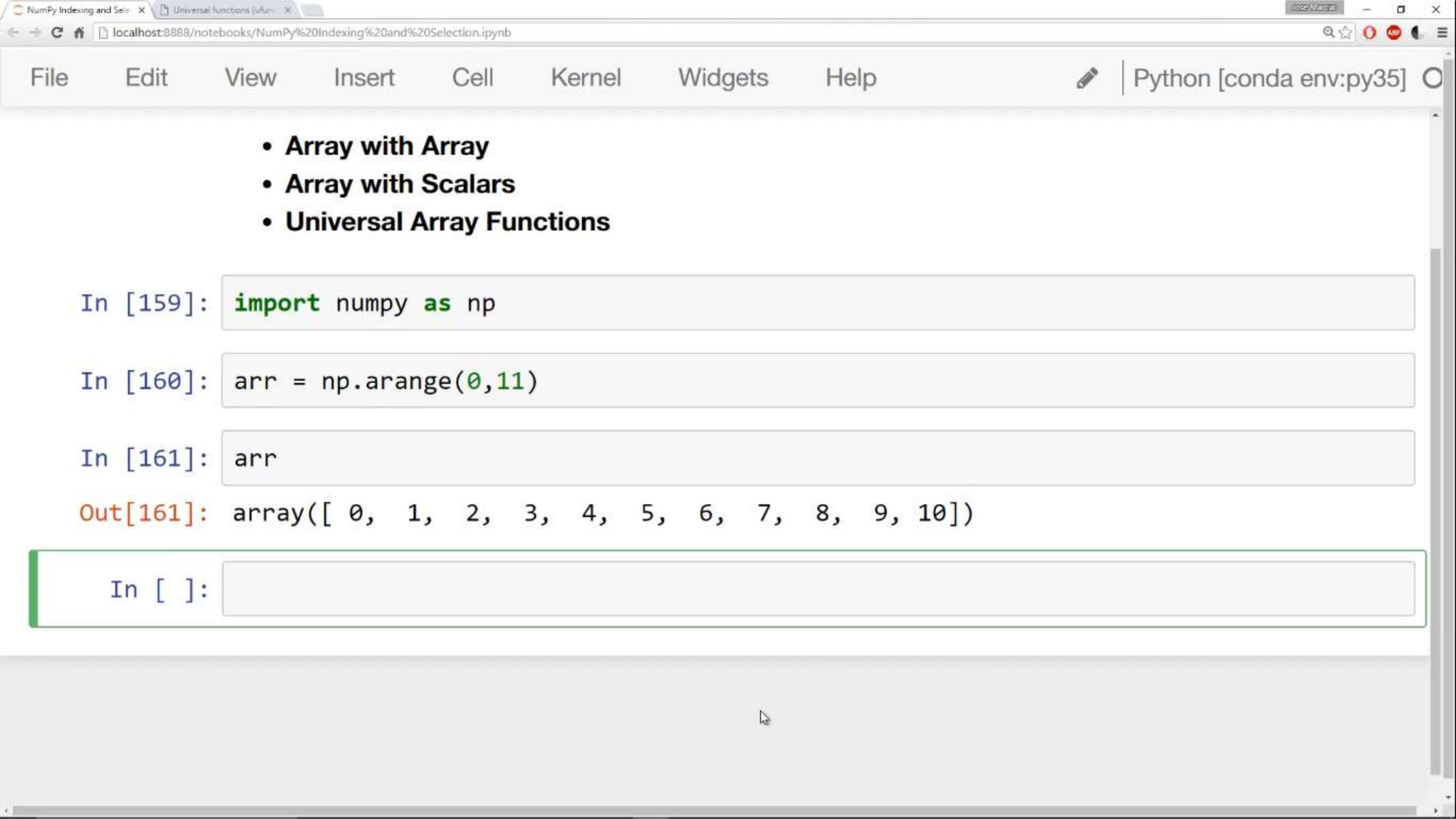
NumPy Operations

PIERIAN DATA
www.pieriandata.com



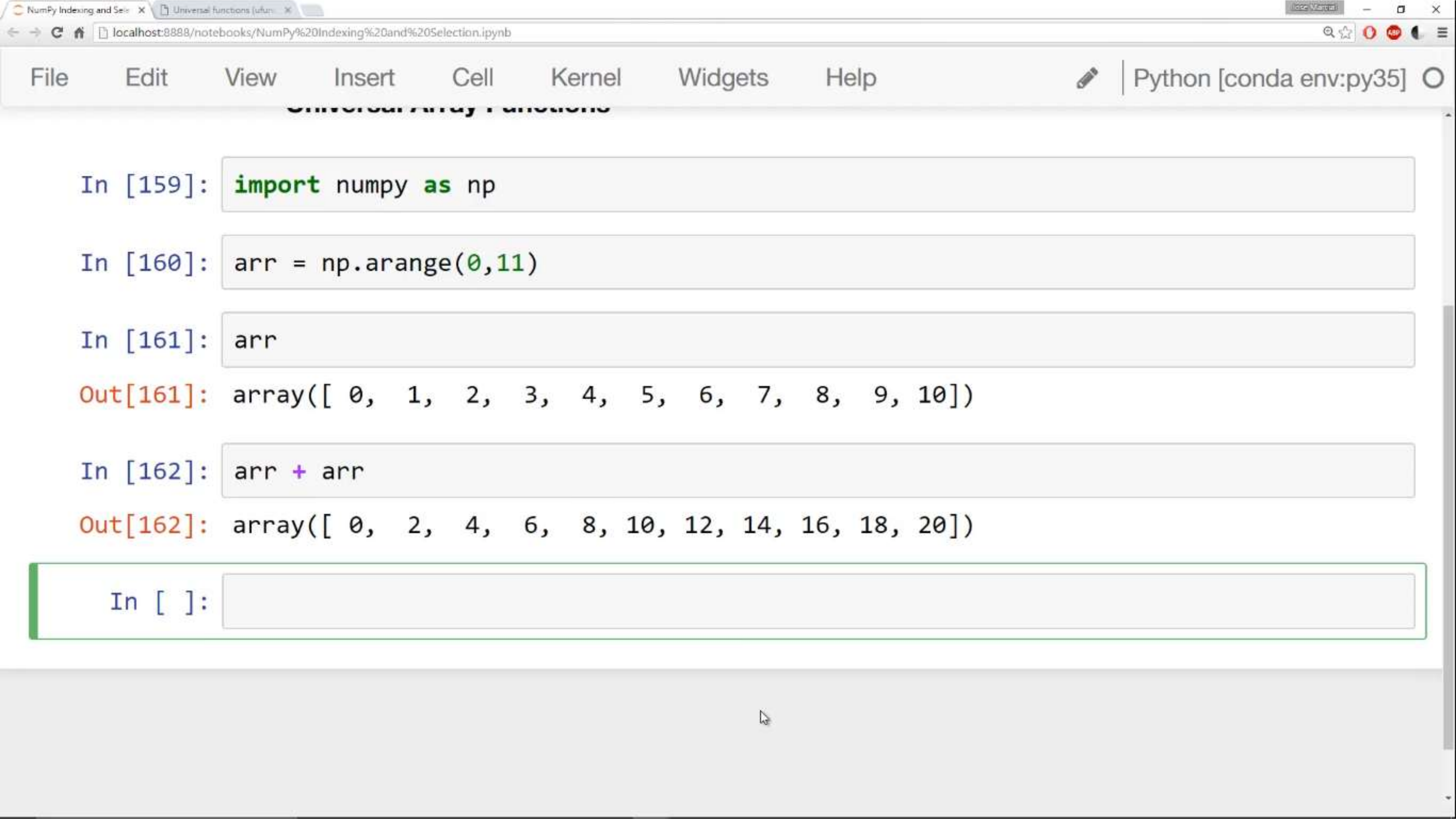
NumPy Operations

- **Array with Array**
- **Array with Scalars**
- **Universal Array Functions**


NumPy Indexing and Sel
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

- **Array with Array**
- **Array with Scalars**
- **Universal Array Functions**

```
In [159]: import numpy as np

In [160]: arr = np.arange(0,11)

In [161]: arr

Out[161]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])

In [ ]:
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

Universal Array Functions

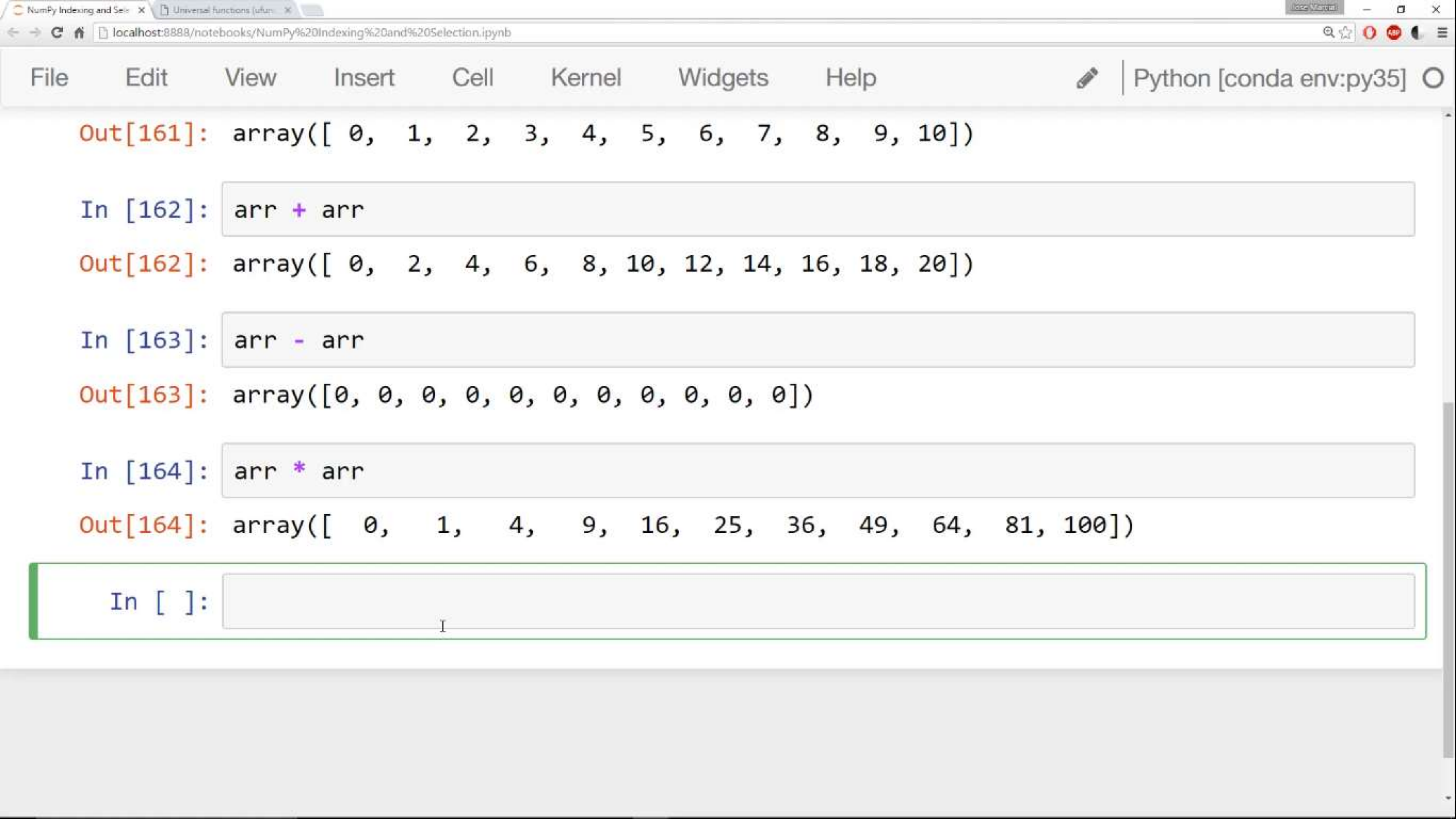
```
In [159]: import numpy as np
```

```
In [160]: arr = np.arange(0,11)
```

```
In [161]: arr

Out[161]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

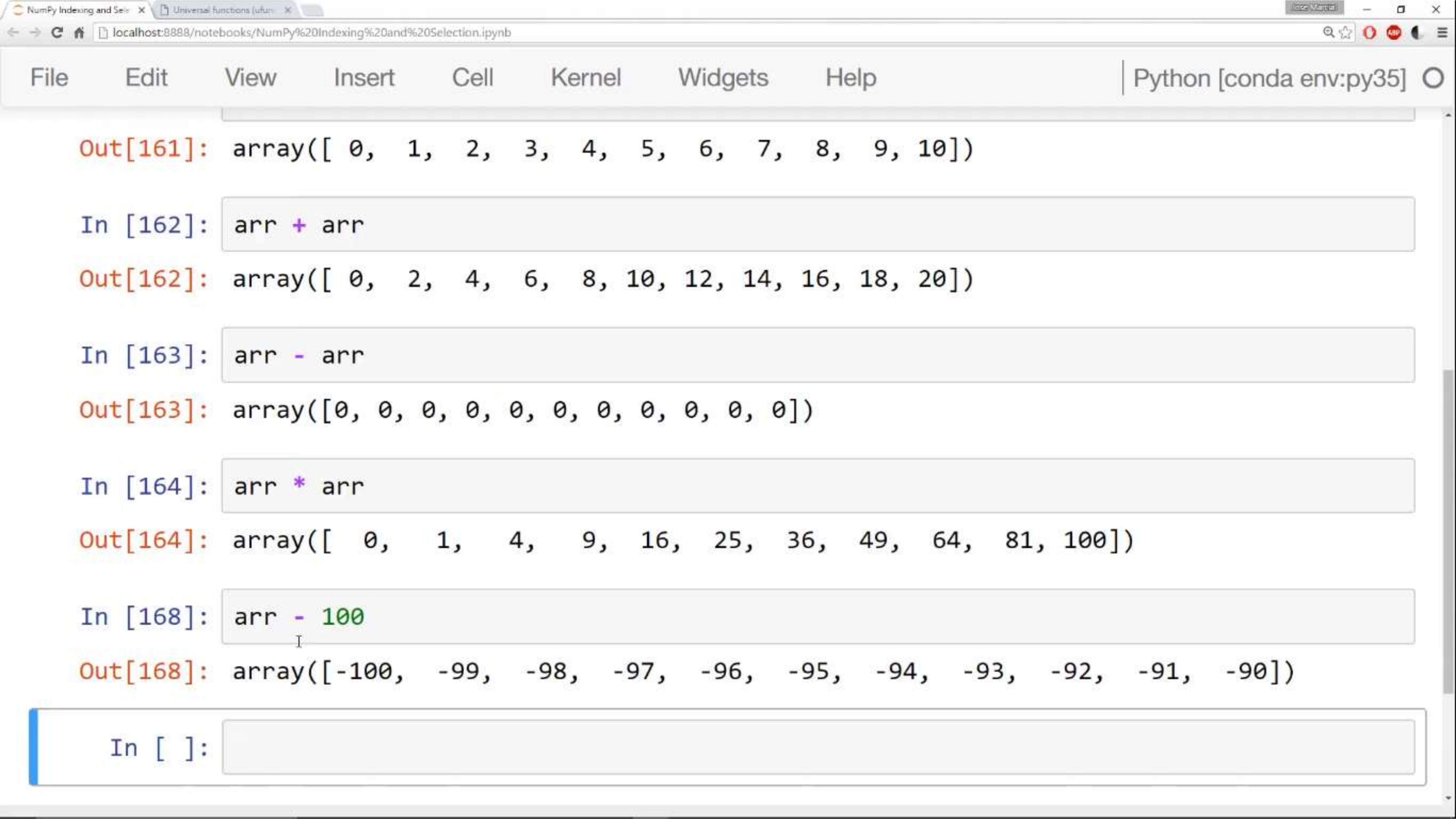
```
In [162]: arr + arr

Out[162]: array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18, 20])
```

```
In [ ]:
```



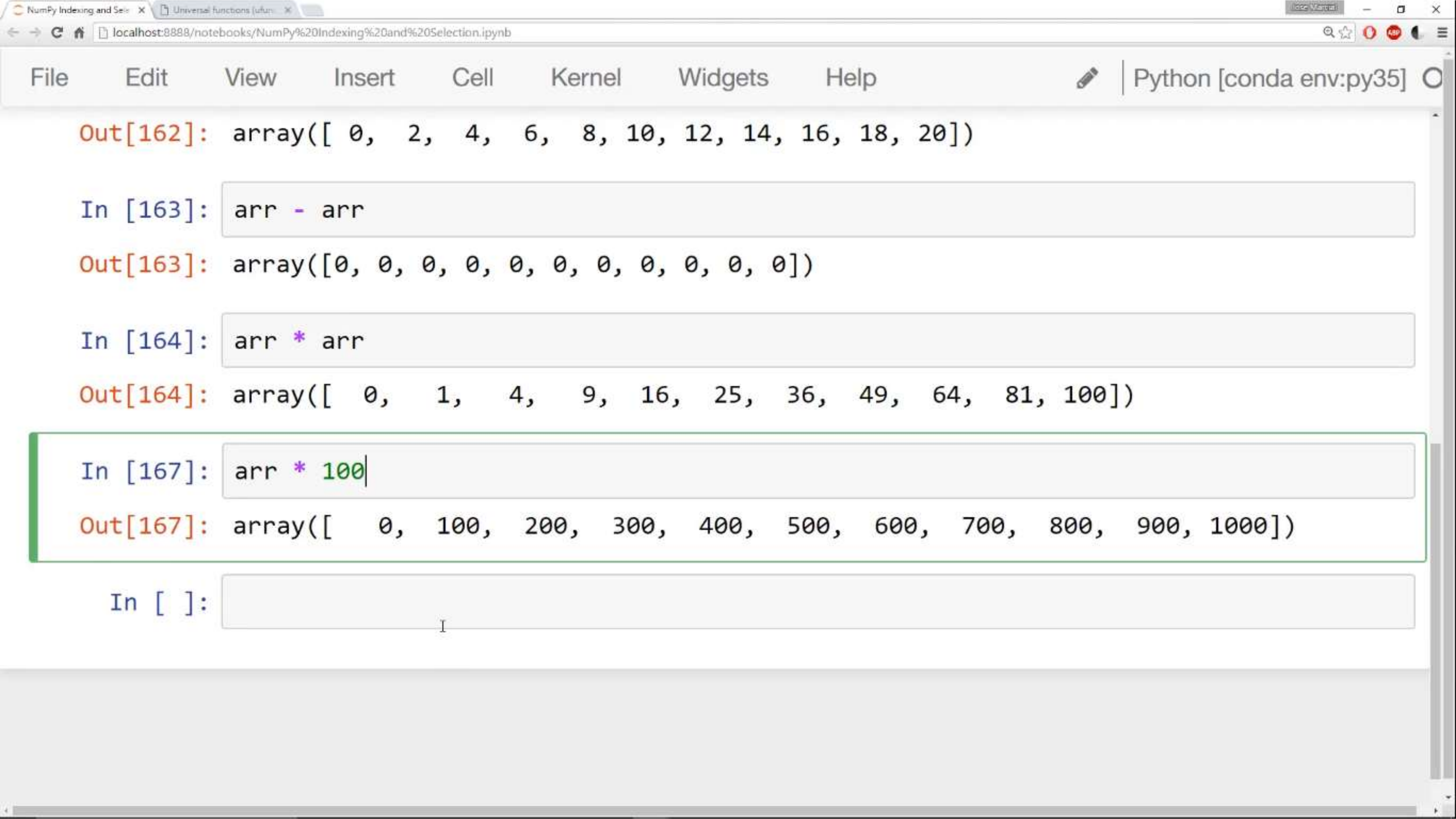
NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
Out[161]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])

In [162]: arr + arr

Out[162]: array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18, 20])

In [163]: arr - arr

Out[163]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

In [164]: arr * arr

Out[164]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])

In [ ]:
```

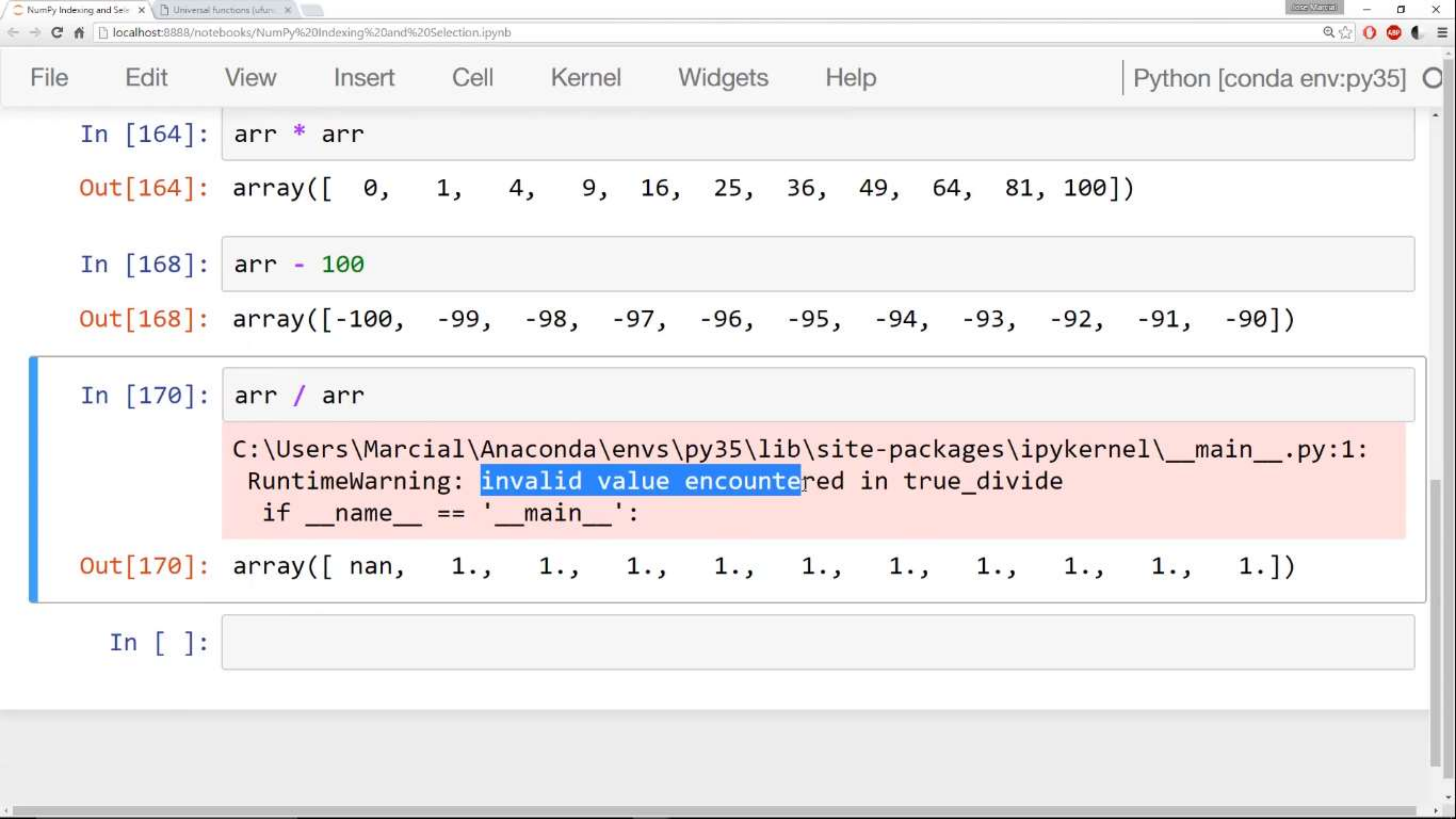


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
Out[161]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])

In [162]: arr + arr

Out[162]: array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18, 20])

In [163]: arr - arr

Out[163]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

In [164]: arr * arr

Out[164]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])

In [168]: arr - 100

Out[168]: array([-100,  -99,  -98,  -97,  -96,  -95,  -94,  -93,  -92,  -91,  -90])

In [ ]:
```

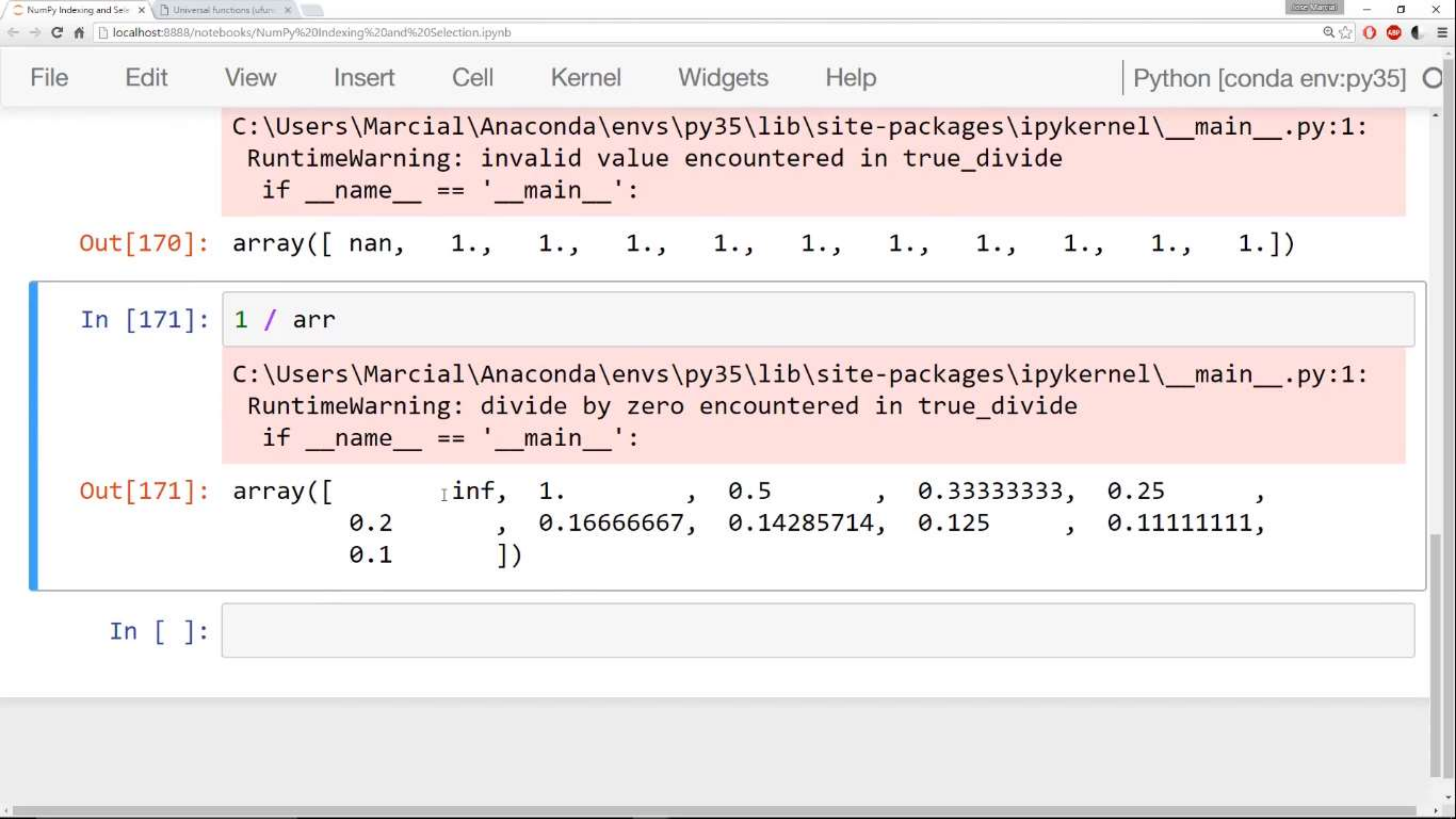

NumPy Indexing and Sel
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
Out[162]: array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18, 20])

In [163]: arr - arr

Out[163]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

In [164]: arr * arr

Out[164]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])

In [167]: arr * 100

Out[167]: array([   0,  100,  200,  300,  400,  500,  600,  700,  800,  900, 1000])

In [ ]:
```

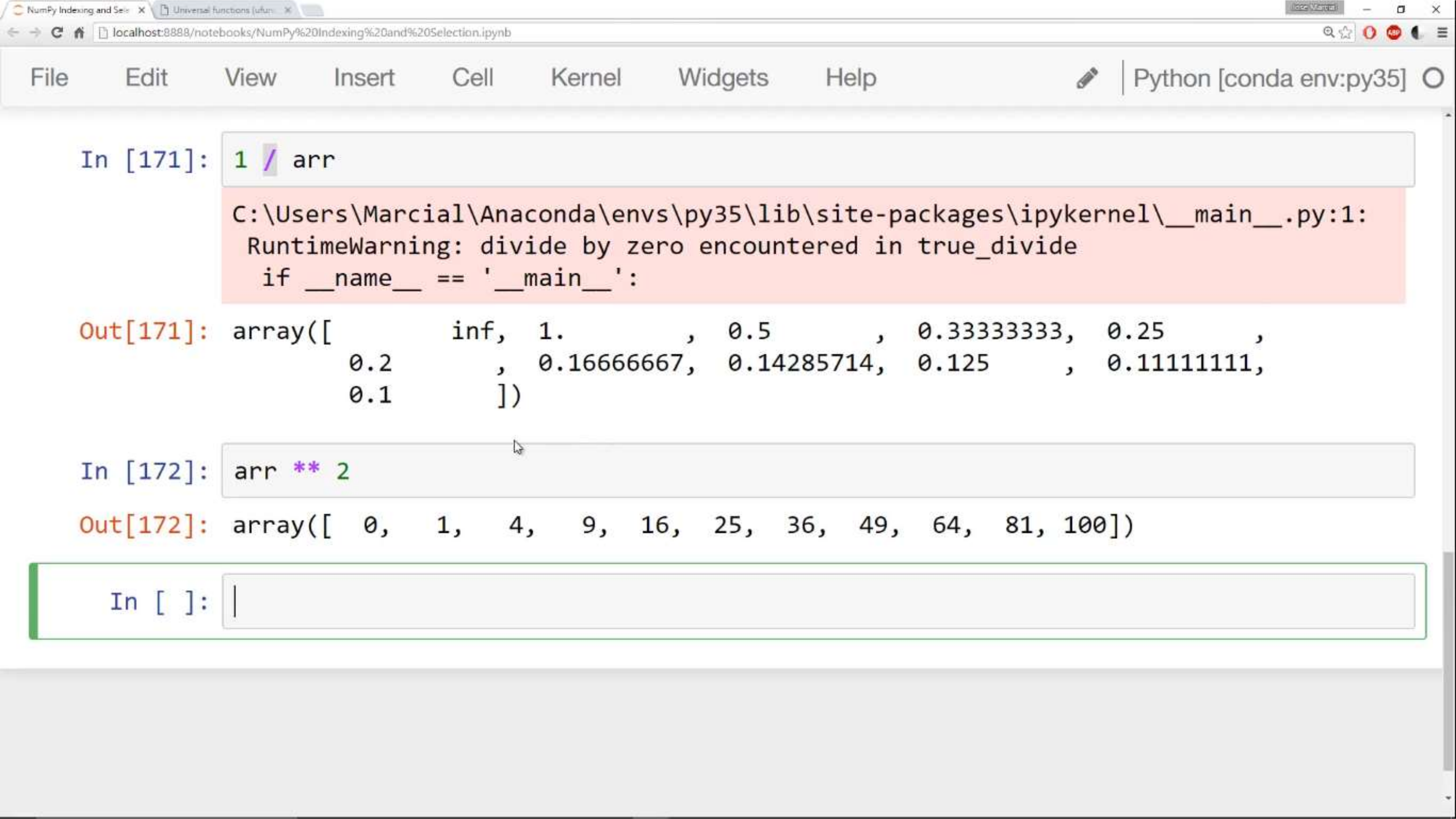


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
In [164]: arr * arr

Out[164]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])

In [168]: arr - 100

Out[168]: array([-100,  -99,  -98,  -97,  -96,  -95,  -94,  -93,  -92,  -91,  -90])

In [170]: arr / arr

C:\Users\Marcial\Anaconda\envs\py35\lib\site-packages\ipykernel\__main__.py:1:
 RuntimeWarning: invalid value encountered in true_divide
  if __name__ == '__main__':

Out[170]: array([ nan,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.])

In [ ]:
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
C:\Users\Marcial\Anaconda\envs\py35\lib\site-packages\ipykernel\__main__.py:1:
 RuntimeWarning: invalid value encountered in true_divide
  if __name__ == '__main__':

Out[170]: array([ nan,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.])

In [171]: 1 / arr

C:\Users\Marcial\Anaconda\envs\py35\lib\site-packages\ipykernel\__main__.py:1:
 RuntimeWarning: divide by zero encountered in true_divide
  if __name__ == '__main__':

Out[171]: array([        inf,  1.        ,  0.5       ,  0.33333333,  0.25      ,
                 0.2       ,  0.16666667,  0.14285714,  0.125     ,  0.11111111,
                 0.1       ])

In [ ]:
```



NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
In [171]: 1 / arr
```

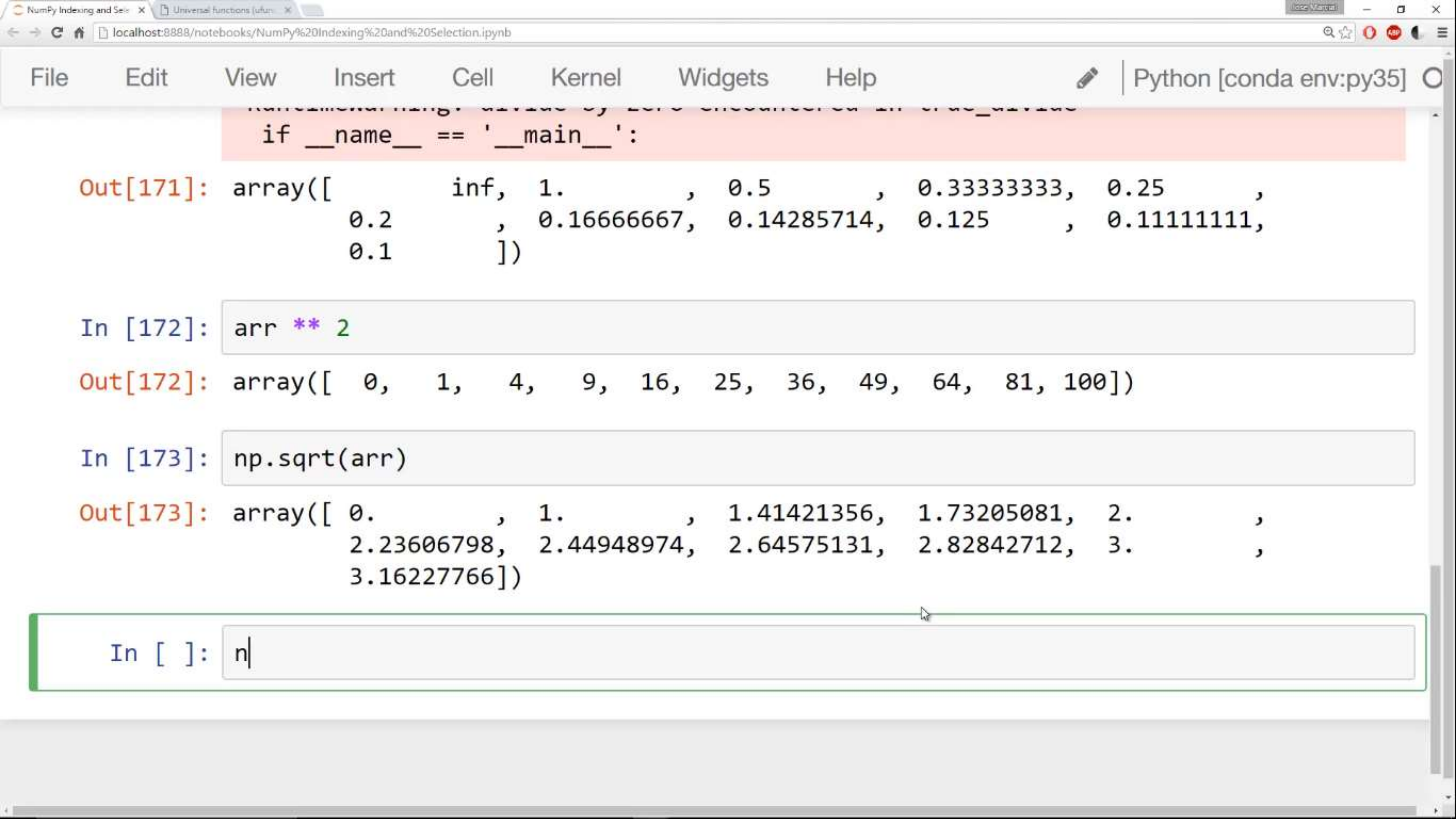
```
C:\Users\Marcial\Anaconda\envs\py35\lib\site-packages\ipykernel\__main__.py:1:
 RuntimeWarning: divide by zero encountered in true_divide
  if __name__ == '__main__':
```

```
Out[171]: array([        inf,  1.        ,  0.5       ,  0.33333333,  0.25      ,
                 0.2       ,  0.16666667,  0.14285714,  0.125     ,  0.11111111,
                 0.1       ])
```

```
In [172]: arr ** 2
```

```
Out[172]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])
```

```
In [ ]:
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
RuntimeWarning: divide by zero encountered in true_divide
  if __name__ == '__main__':
```

```
Out[171]: array([        inf,  1.        ,  0.5       ,  0.33333333,  0.25      ,
                  0.2       ,  0.16666667,  0.14285714,  0.125     ,  0.11111111,
                  0.1       ])
```

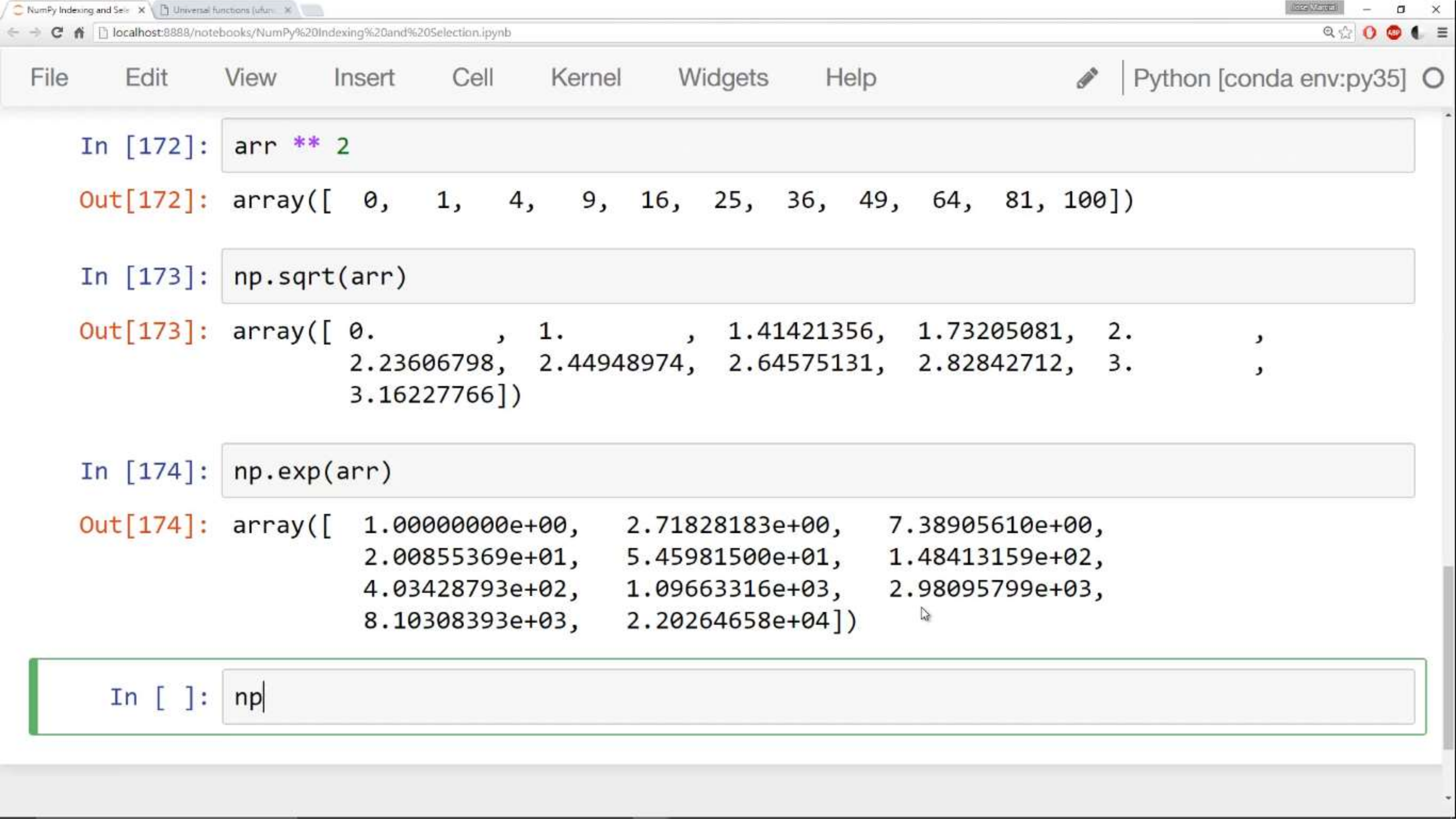
```
In [172]: arr ** 2
```

```
Out[172]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])
```

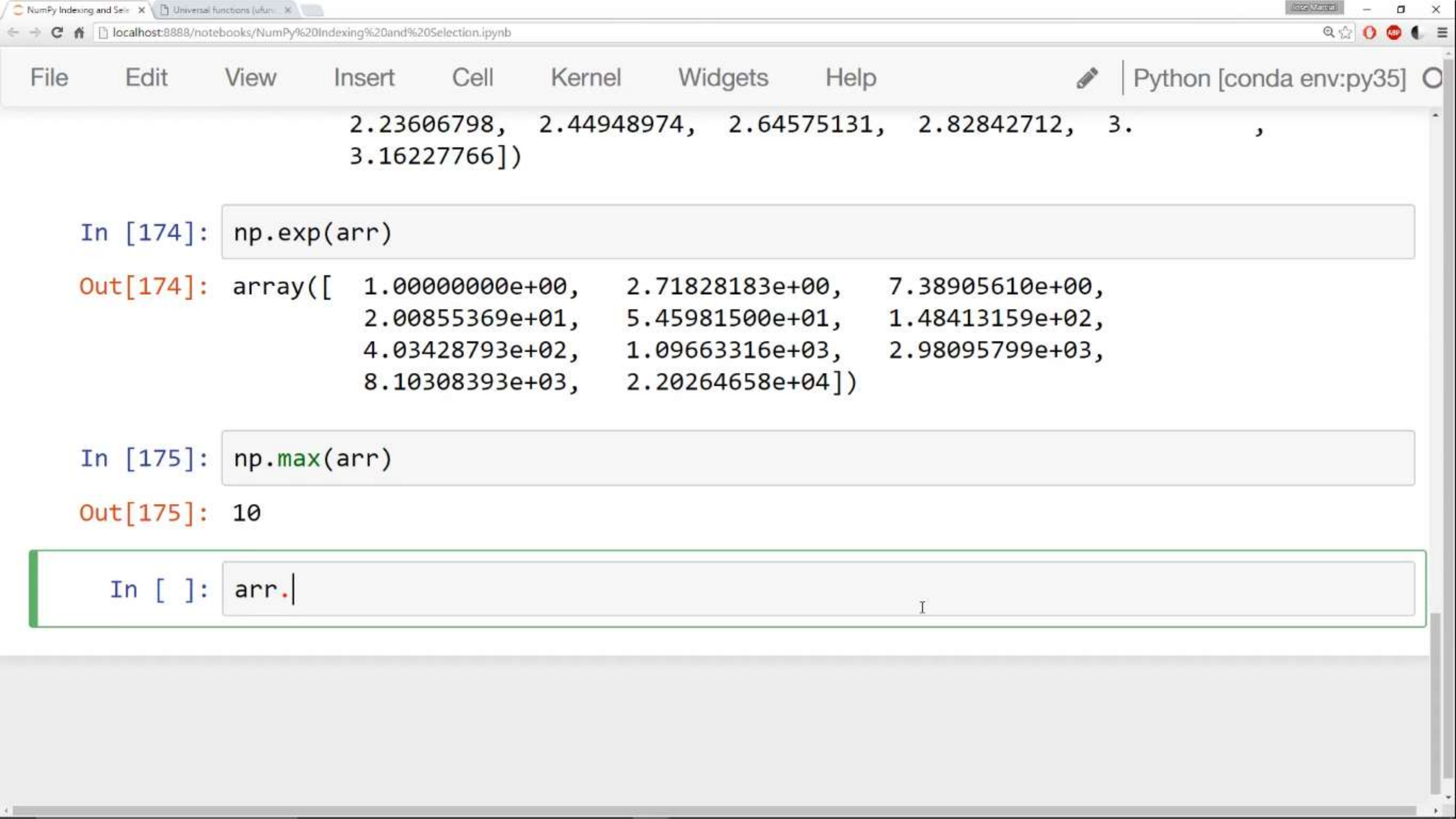
```
In [173]: np.sqrt(arr)
```

```
Out[173]: array([ 0.        ,  1.        ,  1.41421356,  1.73205081,  2.        ,
                  2.23606798,  2.44948974,  2.64575131,  2.82842712,  3.        ,
                  3.16227766])
```

```
In [ ]: n
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
In [172]: arr ** 2

Out[172]: array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100])

In [173]: np.sqrt(arr)

Out[173]: array([ 0.        ,  1.        ,  1.41421356,  1.73205081,  2.        ,
                  2.23606798,  2.44948974,  2.64575131,  2.82842712,  3.        ,
                  3.16227766])

In [174]: np.exp(arr)

Out[174]: array([  1.00000000e+00,   2.71828183e+00,   7.38905610e+00,
                   2.00855369e+01,   5.45981500e+01,   1.48413159e+02,
                   4.03428793e+02,   1.09663316e+03,   2.98095799e+03,
                   8.10308393e+03,   2.20264658e+04])

In [ ]: np
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
        2.23606798,  2.44948974,  2.64575131,  2.82842712,  3.        ,
        3.16227766])
```

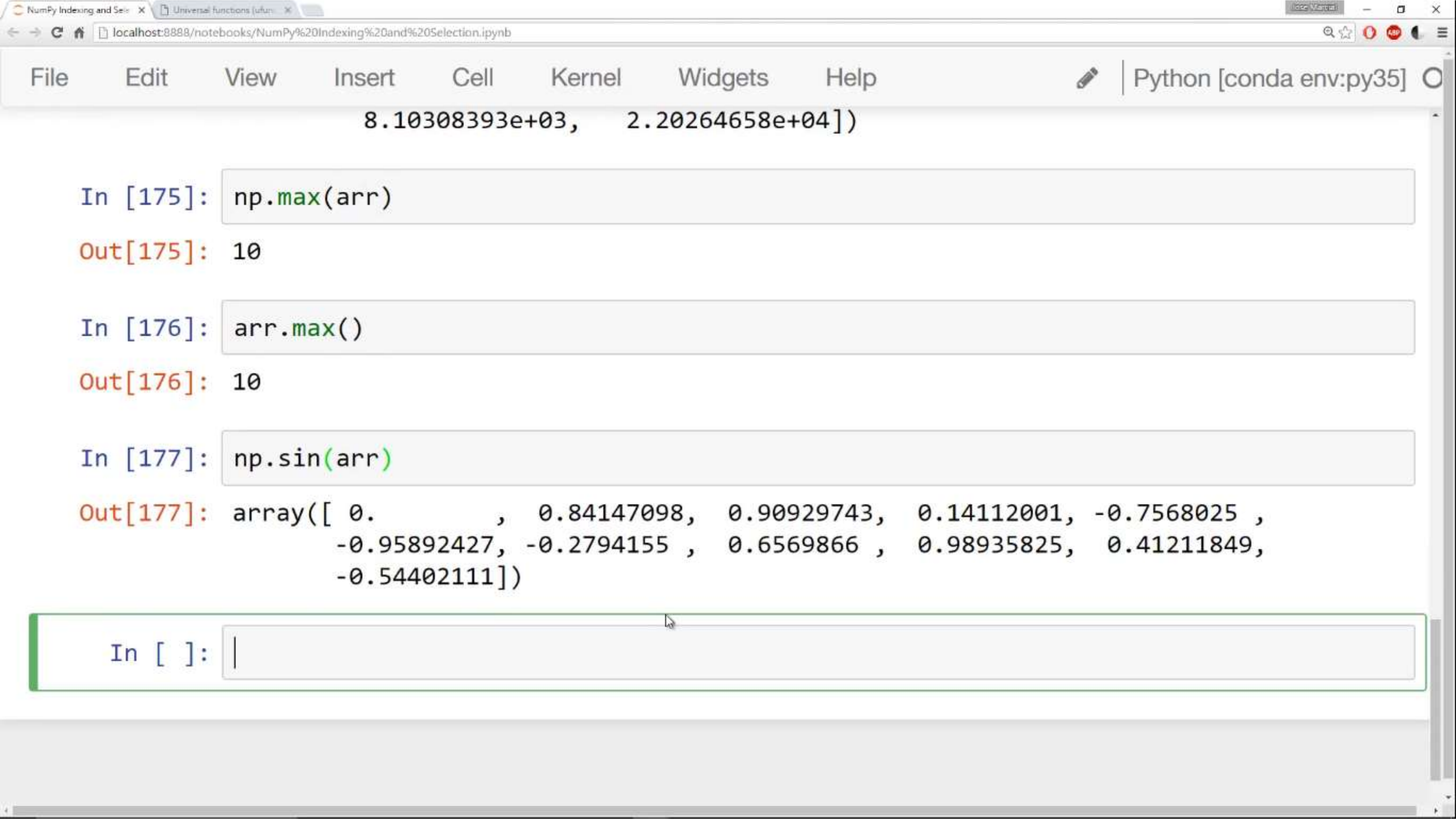
```
In [174]: np.exp(arr)
```

```
Out[174]: array([  1.00000000e+00,   2.71828183e+00,   7.38905610e+00,
                   2.00855369e+01,   5.45981500e+01,   1.48413159e+02,
                   4.03428793e+02,   1.09663316e+03,   2.98095799e+03,
                   8.10308393e+03,   2.20264658e+04])
```

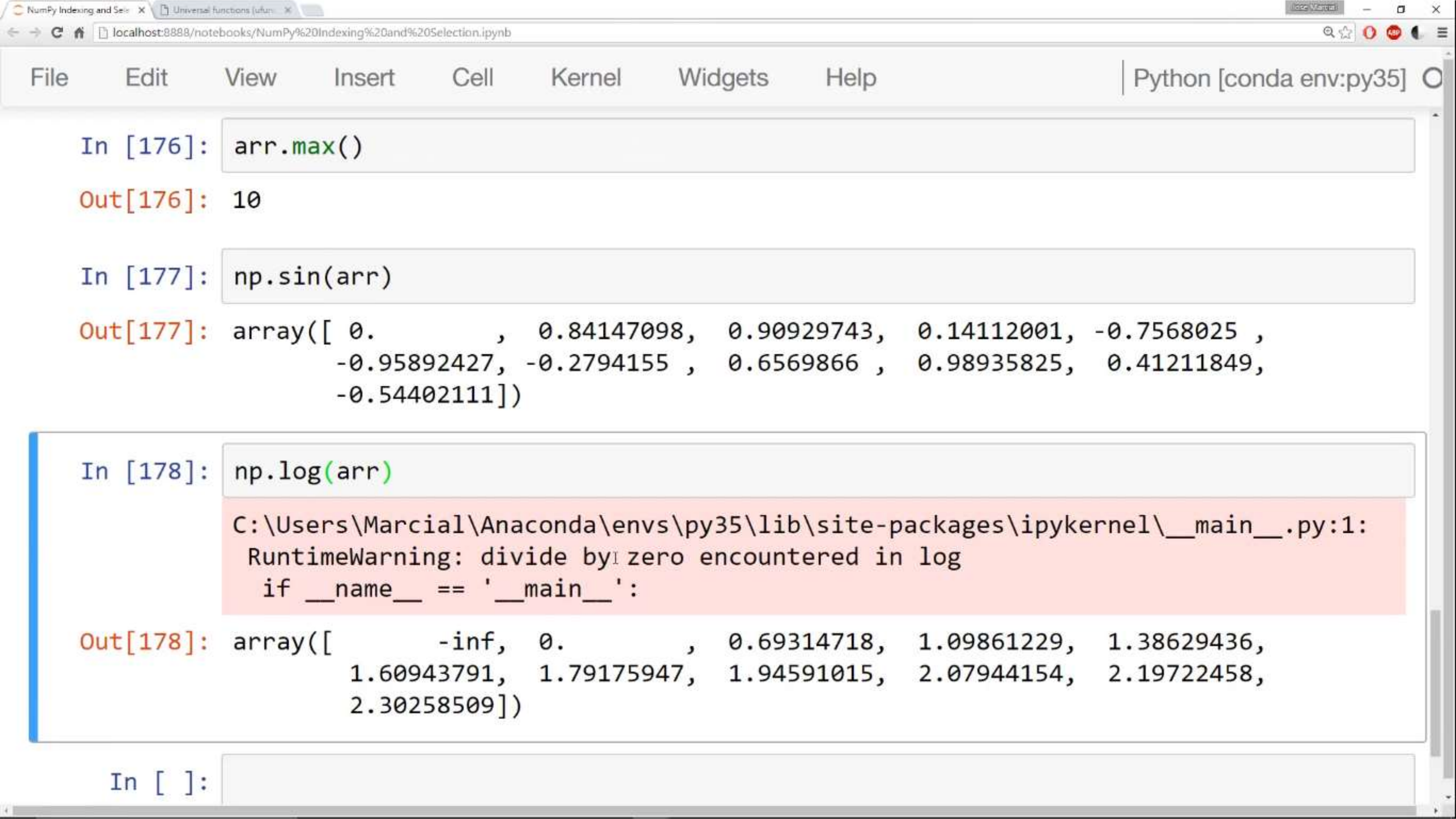
```
In [175]: np.max(arr)
```

```
Out[175]: 10
```

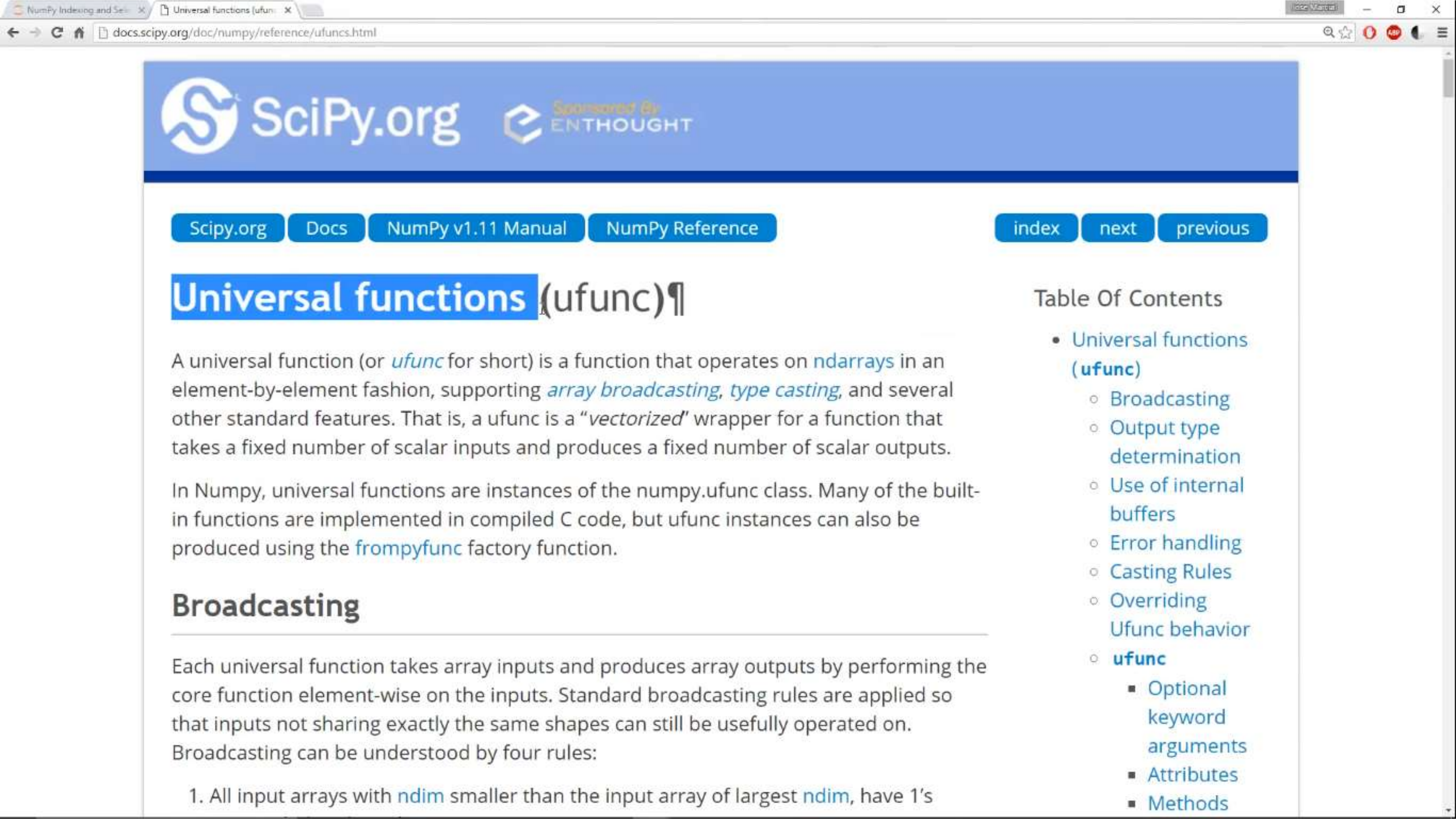
```
In [ ]: arr.
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
        8.10308393e+03,   2.20264658e+04])

In [175]: np.max(arr)

Out[175]: 10

In [176]: arr.max()

Out[176]: 10

In [177]: np.sin(arr)

Out[177]: array([ 0.        ,  0.84147098,  0.90929743,  0.14112001, -0.7568025 ,
                 -0.95892427, -0.2794155 ,  0.6569866 ,  0.98935825,  0.41211849,
                 -0.54402111])

In [ ]:
```


NumPy Indexing and Sele
Universal functions (ufun
localhost:8888/notebooks/NumPy%20Indexing%20and%20Selection.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python [conda env:py35]

```
In [176]: arr.max()

Out[176]: 10

In [177]: np.sin(arr)

Out[177]: array([ 0.        ,  0.84147098,  0.90929743,  0.14112001, -0.7568025 ,
                 -0.95892427, -0.2794155 ,  0.6569866 ,  0.98935825,  0.41211849,
                 -0.54402111])

In [178]: np.log(arr)

          C:\Users\Marcial\Anaconda\envs\py35\lib\site-packages\ipykernel\__main__.py:1:
           RuntimeWarning: divide by zero encountered in log
            if __name__ == '__main__':

Out[178]: array([       -inf,  0.        ,  0.69314718,  1.09861229,  1.38629436,
                  1.60943791,  1.79175947,  1.94591015,  2.07944154,  2.19722458,
                  2.30258509])

In [ ]:
```



Universal functions (ufunc)¶

A universal function (or *ufunc* for short) is a function that operates on ndarrays in an element-by-element fashion, supporting *array broadcasting*, *type casting*, and several other standard features. That is, a ufunc is a "*vectorized*" wrapper for a function that takes a fixed number of scalar inputs and produces a fixed number of scalar outputs.

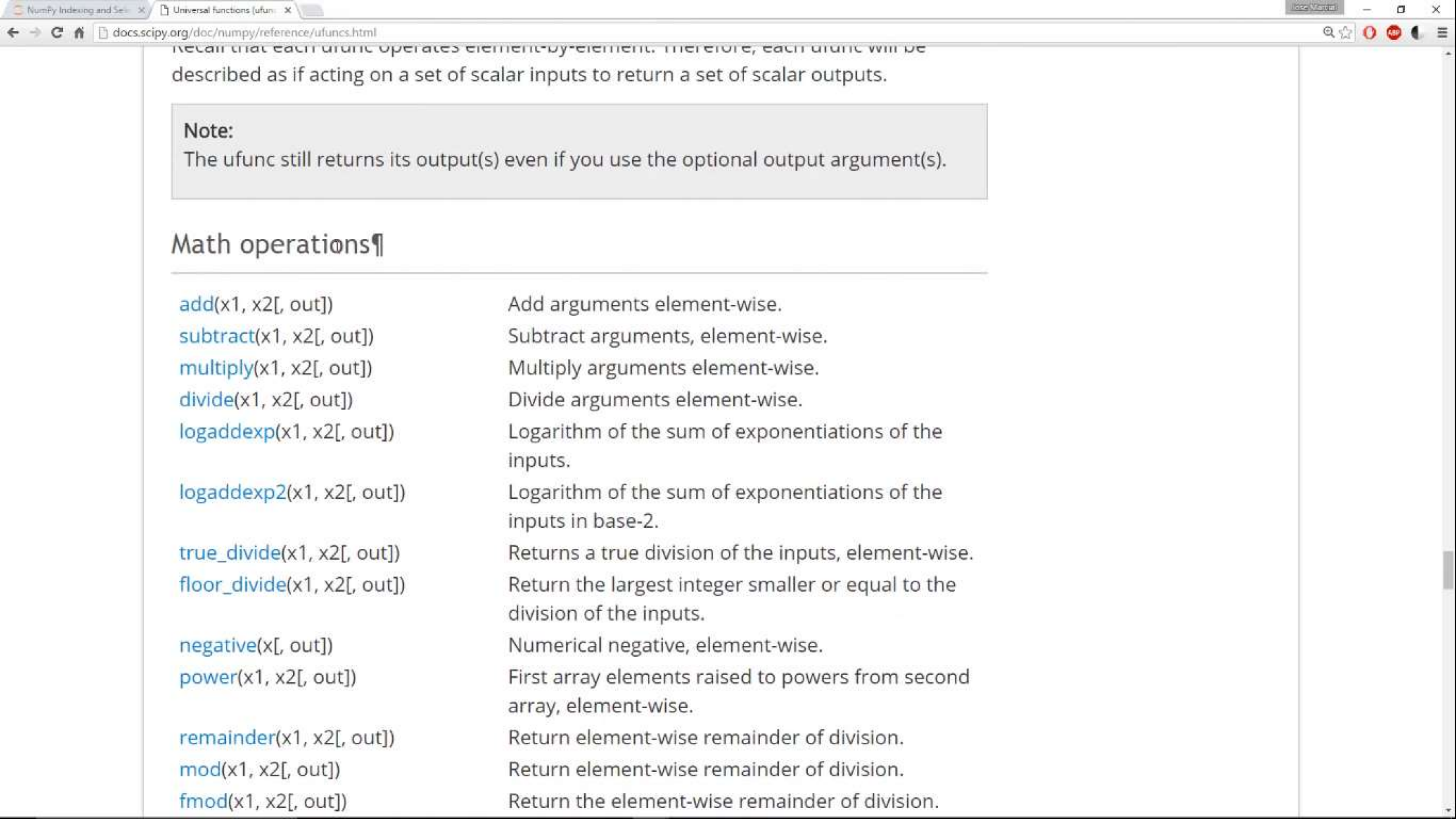
In Numpy, universal functions are instances of the numpy.ufunc class. Many of the built-in functions are implemented in compiled C code, but ufunc instances can also be produced using the frompyfunc factory function.

Broadcasting

Each universal function takes array inputs and produces array outputs by performing the core function element-wise on the inputs. Standard broadcasting rules are applied so that inputs not sharing exactly the same shapes can still be usefully operated on. Broadcasting can be understood by four rules:

1. All input arrays with ndim smaller than the input array of largest ndim, have 1's

Table Of Contents


NumPy Indexing and Se
Universal functions (ufun
docs.scipy.org/doc/numpy/reference/ufuncs.html

Recall that each ufunc operates element-by-element. Therefore, each ufunc will be described as if acting on a set of scalar inputs to return a set of scalar outputs.

Note:
The ufunc still returns its output(s) even if you use the optional output argument(s).

Math operations¶

add(x1, x2[, out])	Add arguments element-wise.
subtract(x1, x2[, out])	Subtract arguments, element-wise.
multiply(x1, x2[, out])	Multiply arguments element-wise.
divide(x1, x2[, out])	Divide arguments element-wise.
logaddexp(x1, x2[, out])	Logarithm of the sum of exponentiations of the inputs.
logaddexp2(x1, x2[, out])	Logarithm of the sum of exponentiations of the inputs in base-2.
true_divide(x1, x2[, out])	Returns a true division of the inputs, element-wise.
floor_divide(x1, x2[, out])	Return the largest integer smaller or equal to the division of the inputs.
negative(x[, out])	Numerical negative, element-wise.
power(x1, x2[, out])	First array elements raised to powers from second array, element-wise.
remainder(x1, x2[, out])	Return element-wise remainder of division.
mod(x1, x2[, out])	Return element-wise remainder of division.
fmod(x1, x2[, out])	Return the element-wise remainder of division.