Class 3 Numpy Operations

April 13, 2020

1 NumPy Operations

1.1 Arithmetic

You can easily perform array with array arithmetic, or scalar with array arithmetic. Let's see some examples:

```
[1]: import numpy as np
     arr = np.arange(0,10)
```

```
[2]: arr + arr
```

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

```
[3]: arr * arr
```

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

```
[4]: arr - arr
```

```
[4]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

```
[5]: # Warning on division by zero, but not an error!
     # Just replaced with nan
     arr/arr
```

```
/Users/marci/anaconda/lib/python3.5/site-packages/ipykernel/__main__.py:1:
RuntimeWarning: invalid value encountered in true_divide
  if __name__ == '__main__':
```

```
[5]: array([ nan,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.,   1.])
```

```
[6]: # Also warning, but not an error instead infinity
     1/arr
```

```
/Users/marci/anaconda/lib/python3.5/site-packages/ipykernel/__main__.py:1:
RuntimeWarning: divide by zero encountered in true_divide
  if __name__ == '__main__':
```

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

```
[10]: arr**3
```

```
[10]: array([  0,   1,   8,  27,  64, 125, 216, 343, 512, 729])
```

1.2 Universal Array Functions

Numpy comes with many universal array functions, which are essentially just mathematical operations you can use to perform the operation across the array. Let's show some common ones:

```
[12]: #Taking Square Roots
      np.sqrt(arr)
```

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

```
[13]: #Calcualting exponential (e^)
      np.exp(arr)
```

```
[13]: 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])
```

```
[14]: np.max(arr) #same as arr.max()
```

```
[14]: 9
```

```
[15]: np.sin(arr)
```

```
[15]: array([ 0.        ,  0.84147098,  0.90929743,  0.14112001, -0.7568025 ,
        -0.95892427, -0.2794155 ,  0.6569866 ,  0.98935825,  0.41211849])
```

```
[16]: np.log(arr)
```

```
/Users/marci/anaconda/lib/python3.5/site-packages/ipykernel/__main__.py:1:
RuntimeWarning: divide by zero encountered in log
  if __name__ == '__main__':
```

```
[16]: array([       -inf,  0.        ,  0.69314718,  1.09861229,  1.38629436,
         1.60943791,  1.79175947,  1.94591015,  2.07944154,  2.19722458])
```

2 Great Job!

That's all we need to know for now!

```
[1]: import pandas as pd
     dir(pd)
```

```
[1]: ['Categorical',
      'CategoricalDtype',
      'CategoricalIndex',
      'DataFrame',
      'DateOffset',
      'DatetimeIndex',
      'DatetimeTZDtype',
      'ExcelFile',
      'ExcelWriter',
      'Float64Index',
      'Grouper',
      'HDFStore',
      'Index',
      'IndexSlice',
      'Int16Dtype',
      'Int32Dtype',
      'Int64Dtype',
      'Int64Index',
      'Int8Dtype',
      'Interval',
      'IntervalDtype',
      'IntervalIndex',
      'MultiIndex',
      'NaT',
      'Panel',
      'Period',
      'PeriodDtype',
      'PeriodIndex',
      'RangeIndex',
      'Series',
      'SparseArray',
      'SparseDataFrame',
      'SparseDtype',
      'SparseSeries',
      'TimeGrouper',
      'Timedelta',
      'TimedeltaIndex',
      'Timestamp',
      'UInt16Dtype',
      'UInt32Dtype',
      'UInt64Dtype',
      'UInt64Index',
      'UInt8Dtype',
      '__builtins__',
'__cached__',
'__doc__',
'__docformat__',
'__file__',
'__git_version__',
'__loader__',
'__name__',
'__package__',
'__path__',
'__spec__',
'__version__',
'_hashtable',
'_lib',
'_libs',
'_np_version_under1p13',
'_np_version_under1p14',
'_np_version_under1p15',
'_np_version_under1p16',
'_np_version_under1p17',
'_tslib',
'_version',
'api',
'array',
'arrays',
'bdate_range',
'compat',
'concat',
'core',
'crosstab',
'cut',
'date_range',
'datetime',
'describe_option',
'errors',
'eval',
'factorize',
'get_dummies',
'get_option',
'infer_freq',
'interval_range',
'io',
'isna',
'isnull',
'lreshape',
'melt',
'merge',
'merge_asof',
'merge_ordered',
'notna',
'notnull',
'np',
'offsets',
'option_context',
'options',
'pandas',
'period_range',
'pivot',
'pivot_table',
'plotting',
'qcut',
'read_clipboard',
'read_csv',
'read_excel',
'read_feather',
'read_fwf',
'read_gbq',
'read_hdf',
'read_html',
'read_json',
'read_msgpack',
'read_parquet',
'read_pickle',
'read_sas',
'read_sql',
'read_sql_query',
'read_sql_table',
'read_stata',
'read_table',
'reset_option',
'set_eng_float_format',
'set_option',
'show_versions',
'test',
'testing',
'timedelta_range',
'to_datetime',
'to_msgpack',
'to_numeric',
'to_pickle',
'to_timedelta',
'tseries',
'unique',
'util',
'value_counts',
 'wide_to_long']
```

```
[2]: len(dir(pd))
```

```
[2]: 139
```

```
[ ]:
```