graphlab.SFrame.apply

`SFrame.apply`*(fn, dtype=None, seed=None)*

Transform each row to an `SArray` according to a specified function. Returns a new SArray of `dtype` where each element in this SArray is transformed by *fn(x)* where *x* is a single row in the sframe represented as a dictionary. The `fn` should return exactly one value which can be cast into type `dtype`. If `dtype` is not specified, the first 100 rows of the SFrame are used to make a guess of the target data type.

Parameters:

fn : function

The function to transform each row of the SFrame. The return type should be convertible to *dtype* if *dtype* is not None. This can also be a toolkit extension function which is compiled as a native shared library using SDK.

dtype : dtype, optional

The dtype of the new SArray. If None, the first 100 elements of the array are used to guess the target data type.

seed : int, optional

Used as the seed if a random number generator is included in *fn*.

Returns:

out : SArray

The SArray transformed by fn. Each element of the SArray is of type `dtype`

Examples

Concatenate strings from several columns:

```
>>> sf = graphlab.SFrame({'user_id': [1, 2, 3], 'movie_id': [3, 3, 6],
                          'rating': [4, 5, 1]})
>>> sf.apply(lambda x: str(x['user_id']) + str(x['movie_id']) + str(x['rating']))
dtype: str
Rows: 3
['134', '235', '361']
```

Using native toolkit extension function:

```
#include <graphlab/sdk/toolkit_function_macros.hpp>
double mean(const std::map<flexible_type, flexible_type>& dict) {
  double sum = 0.0;
  for (const auto& kv: dict) sum += (double)kv.second;
  return sum / dict.size();
}

BEGIN_FUNCTION_REGISTRATION
REGISTER_FUNCTION(mean, "row");
END_FUNCTION_REGISTRATION
```

compiled into example.so

```
>>> import example
```

```
>>> sf = graphlab.SFrame({'x0': [1, 2, 3], 'x1': [2, 3, 1],
...                       'x2': [3, 1, 2]})
>>> sf.apply(example.mean)
dtype: float
Rows: 3
[2.0,2.0,2.0]
```