

Proposed API for tech.ml.dataset

GenerateMe

2020-05-19

Introduction

tech.ml.dataset is a great and fast library which brings columnar dataset to the Clojure. Chris Nuernberger has been working on this library for last year as a part of bigger **tech.ml** stack.

I've started to test the library and help to fix uncovered bugs. My main goal was to compare functionalities with the other standards from other platforms. I focused on R solutions: dplyr, tidyr and data.table.

During conversions of the examples I've come up how to reorganized existing **tech.ml.dataset** functions into simple to use API. The main goals were:

- Focus on dataset manipulation functionality, leaving other parts of **tech.ml** like pipelines, datatypes, readers, ML, etc.
- Single entry point for common operations - one function dispatching on given arguments.
- **group-by** results with special kind of dataset - a dataset containing subsets created after grouping as a column.
- Most operations recognize regular dataset and grouped dataset and process data accordingly.
- One function form to enable thread-first on dataset.

All proposed functions are grouped in tabs below. Select group to see examples and details.

```
(require '[techtest.api :as api])
```

Functionality

Dataset

Dataset is a special type which can be considered as a map of columns implemented around **tech.ml.datatype** library. Each column can be considered as named sequence of typed data. Supported types include integers, floats, string, boolean, date/time, objects etc.

Dataset creation

Dataset can be created from various of types of Clojure structures and files:

- single values
- sequence of maps
- map of sequences or values
- sequence of columns (taken from other dataset or created manually)
- sequence of pairs
- file types: raw/gzipped csv/tsv, json, xls(x) taken from local file system or URL
- input stream

api/dataset accepts:

- data
- options (see documentation of `tech.ml.dataset/->dataset` function for full list):
 - `:dataset-name` - name of the dataset
 - `:num-rows` - number of rows to read from file
 - `:header-row?` - indication if first row in file is a header
 - `:key-fn` - function applied to column names (eg. `keyword`, to convert column names to keywords)
 - `:separator` - column separator
 - `:single-value-column-name` - name of the column when single value is provided

Empty dataset.

```
(api/dataset)
```

```
_unnamed [0 0]:
```

Dataset from single value.

```
(api/dataset 999)
```

```
_unnamed [1 1]:
```

:\$value
999

Set column name for single value. Also set the dataset name.

```
(api/dataset 999 {:single-value-column-name "my-single-value"})
(api/dataset 999 {:single-value-column-name ""
                  :dataset-name "Single value"})
```

```
_unnamed [1 1]:
```

my-single-value
999

Single value [1 1]:

0
999

Sequence of pairs (first = column name, second = value(s)).

```
(api/dataset [[:A 33] [:B 5] [:C :a]])
```

```
_unnamed [1 3]:
```

:A	:B	:C
33	5	:a

Not sequential values are repeated row-count number of times.

```
(api/dataset [[:A [1 2 3 4 5 6]] [:B "X"] [:C :a]])
```

__unnamed [6 3]:

:A	:B	:C
1	X	:a
2	X	:a
3	X	:a
4	X	:a
5	X	:a
6	X	:a

Dataset created from map (keys = column name, second = value(s)). Works the same as sequence of pairs.

```
(api/dataset { :A 33 })
(api/dataset { :A [1 2 3] })
(api/dataset { :A [3 4 5] :B "X" })
```

__unnamed [1 1]:

:A
33

__unnamed [3 1]:

:A
1
2
3

__unnamed [3 2]:

:A	:B
3	X
4	X
5	X

You can put any value inside a column

```
(api/dataset { :A [[3 4 5] [:a :b]] :B "X"})
```

__unnamed [2 2]:

:A	:B
[3 4 5]	X
[:a :b]	X

Sequence of maps

```
(api/dataset [{ :a 1 :b 3} { :b 2 :a 99}])  
(api/dataset [{ :a 1 :b [1 2 3]} { :a 2 :b [3 4]}])
```

__unnamed [2 2]:

:a	:b
1	3
99	2

__unnamed [2 2]:

:a	:b
1	[1 2 3]
2	[3 4]

Missing values are marked by nil

```
(api/dataset [{ :a nil :b 1} { :a 3 :b 4} { :a 11}])
```

__unnamed [3 2]:

:a	:b
	1
3	4
11	

Import CSV file

```
(api/dataset "data/family.csv")
```

data/family.csv [5 5]:

family	dob_child1	dob_child2	gender_child1	gender_child2
1	1998-11-26	2000-01-29	1	2
2	1996-06-22		2	

family	dob_child1	dob_child2	gender_child1	gender_child2
3	2002-07-11	2004-04-05	2	2
4	2004-10-10	2009-08-27	1	1
5	2000-12-05	2005-02-28	2	1

Import from URL

```
(defonce ds (api/dataset "https://vega.github.io/vega-lite/examples/data/seattle-weather.csv"))
ds
```

<https://vega.github.io/vega-lite/examples/data/seattle-weather.csv> [1461 6]:

date	precipitation	temp_max	temp_min	wind	weather
2012-01-01	0.000	12.80	5.000	4.700	drizzle
2012-01-02	10.90	10.60	2.800	4.500	rain
2012-01-03	0.8000	11.70	7.200	2.300	rain
2012-01-04	20.30	12.20	5.600	4.700	rain
2012-01-05	1.300	8.900	2.800	6.100	rain
2012-01-06	2.500	4.400	2.200	2.200	rain
2012-01-07	0.000	7.200	2.800	2.300	rain
2012-01-08	0.000	10.00	2.800	2.000	sun
2012-01-09	4.300	9.400	5.000	3.400	rain
2012-01-10	1.000	6.100	0.6000	3.400	rain
2012-01-11	0.000	6.100	-1.100	5.100	sun
2012-01-12	0.000	6.100	-1.700	1.900	sun
2012-01-13	0.000	5.000	-2.800	1.300	sun
2012-01-14	4.100	4.400	0.6000	5.300	snow
2012-01-15	5.300	1.100	-3.300	3.200	snow
2012-01-16	2.500	1.700	-2.800	5.000	snow
2012-01-17	8.100	3.300	0.000	5.600	snow
2012-01-18	19.80	0.000	-2.800	5.000	snow
2012-01-19	15.20	-1.100	-2.800	1.600	snow
2012-01-20	13.50	7.200	-1.100	2.300	snow
2012-01-21	3.000	8.300	3.300	8.200	rain
2012-01-22	6.100	6.700	2.200	4.800	rain
2012-01-23	0.000	8.300	1.100	3.600	rain
2012-01-24	8.600	10.00	2.200	5.100	rain
2012-01-25	8.100	8.900	4.400	5.400	rain

Saving

Export dataset to a file or output stream can be done by calling `api/write-csv!`. Function accepts:

- dataset
- file name with one of the extensions: `.csv`, `.tsv`, `.csv.gz` and `.tsv.gz` or output stream
- options:
 - `:separator` - string or separator char.

```
(api/write-csv! ds "output.tsv.gz")
(.exists (clojure.java.io/file "output.csv.gz"))
```

`nil`

true

Dataset related functions

Summary of dataset functions like number of rows, columns and basic stats.

Number of rows

```
(api/row-count ds)
```

1461

Number of columns

```
(api/column-count ds)
```

6

Names of columns.

```
(api/column-names ds)
```

```
("date" "precipitation" "temp_max" "temp_min" "wind" "weather")
```

Shape of the dataset, [row count, column count]

```
(api/shape ds)
```

[1461 6]

General info about dataset. There are three variants:

- default - containing information about columns with basic statistics
- `:basic` - just name, row and column count and information if dataset is a result of `group-by` operation
- `:columns` - columns' metadata

```
(api/info ds)
```

```
(api/info ds :basic)
```

```
(api/info ds :columns)
```

<https://vega.github.io/vega-lite/examples/data/seattle-weather.csv>: descriptive-stats [6 10]:

:col-name	:datatype	:n-valid	:n-missing	:mean	:mode	:min	:max	:standard-deviation	:skew
date	:packed-local-date	1461	0	2013-12-31		2012-01-01	2015-12-31		
precipitation	:float32	1461	0	3.029		0.000	55.90	6.680	3.506
temp_max	:float32	1461	0	16.44		-1.600	35.60	7.350	0.2809
temp_min	:float32	1461	0	8.235		-7.100	18.30	5.023	-
weather	:string	1461	0		sun				0.2495
wind	:float32	1461	0	3.241		0.4000	9.500	1.438	0.8917

<https://vega.github.io/vega-lite/examples/data/seattle-weather.csv> :basic info [1 4]:

:name	:grouped?	:rows	:columns
https://vega.github.io/vega-lite/examples/data/seattle-weather.csv	false	1461	6

<https://vega.github.io/vega-lite/examples/data/seattle-weather.csv> :column info [6 4]:

:name	:size	:datatype	:categorical?
date	1461	:packed-local-date	
precipitation	1461	:float32	
temp_max	1461	:float32	
temp_min	1461	:float32	
wind	1461	:float32	
weather	1461	:string	true

Getting a dataset name

```
(api/dataset-name ds)
```

```
"https://vega.github.io/vega-lite/examples/data/seattle-weather.csv"
```

Setting a dataset name (operation is immutable).

```
(->> "seattle-weather"  
  (api/set-dataset-name ds)  
  (api/dataset-name))
```

```
"seattle-weather"
```

Columns and rows

Select column.

```
(ds "wind")  
(api/column ds "date")
```

```
#tech.ml.dataset.column<float32>[1461]
```

```
wind
```

```
[4.700, 4.500, 2.300, 4.700, 6.100, 2.200, 2.300, 2.000, 3.400, 3.400, 5.100, 1.900, 1.300, 5.300, 3.200]
```

```
#tech.ml.dataset.column<packed-local-date>[1461]
```

```
date
```

```
[2012-01-01, 2012-01-02, 2012-01-03, 2012-01-04, 2012-01-05, 2012-01-06, 2012-01-07, 2012-01-08, 2012-01-09, 2012-01-10, 2012-01-11, 2012-01-12, 2012-01-13, 2012-01-14, 2012-01-15, 2012-01-16, 2012-01-17, 2012-01-18, 2012-01-19, 2012-01-20, 2012-01-21, 2012-01-22, 2012-01-23, 2012-01-24, 2012-01-25, 2012-01-26, 2012-01-27, 2012-01-28, 2012-01-29, 2012-01-30, 2012-01-31, 2012-02-01, 2012-02-02, 2012-02-03, 2012-02-04, 2012-02-05, 2012-02-06, 2012-02-07, 2012-02-08, 2012-02-09, 2012-02-10, 2012-02-11, 2012-02-12, 2012-02-13, 2012-02-14, 2012-02-15, 2012-02-16, 2012-02-17, 2012-02-18, 2012-02-19, 2012-02-20, 2012-02-21, 2012-02-22, 2012-02-23, 2012-02-24, 2012-02-25, 2012-02-26, 2012-02-27, 2012-02-28, 2012-03-01, 2012-03-02, 2012-03-03, 2012-03-04, 2012-03-05, 2012-03-06, 2012-03-07, 2012-03-08, 2012-03-09, 2012-03-10, 2012-03-11, 2012-03-12, 2012-03-13, 2012-03-14, 2012-03-15, 2012-03-16, 2012-03-17, 2012-03-18, 2012-03-19, 2012-03-20, 2012-03-21, 2012-03-22, 2012-03-23, 2012-03-24, 2012-03-25, 2012-03-26, 2012-03-27, 2012-03-28, 2012-03-29, 2012-03-30, 2012-03-31, 2012-04-01, 2012-04-02, 2012-04-03, 2012-04-04, 2012-04-05, 2012-04-06, 2012-04-07, 2012-04-08, 2012-04-09, 2012-04-10, 2012-04-11, 2012-04-12, 2012-04-13, 2012-04-14, 2012-04-15, 2012-04-16, 2012-04-17, 2012-04-18, 2012-04-19, 2012-04-20, 2012-04-21, 2012-04-22, 2012-04-23, 2012-04-24, 2012-04-25, 2012-04-26, 2012-04-27, 2012-04-28, 2012-04-29, 2012-04-30, 2012-05-01, 2012-05-02, 2012-05-03, 2012-05-04, 2012-05-05, 2012-05-06, 2012-05-07, 2012-05-08, 2012-05-09, 2012-05-10, 2012-05-11, 2012-05-12, 2012-05-13, 2012-05-14, 2012-05-15, 2012-05-16, 2012-05-17, 2012-05-18, 2012-05-19, 2012-05-20, 2012-05-21, 2012-05-22, 2012-05-23, 2012-05-24, 2012-05-25, 2012-05-26, 2012-05-27, 2012-05-28, 2012-05-29, 2012-05-30, 2012-05-31, 2012-06-01, 2012-06-02, 2012-06-03, 2012-06-04, 2012-06-05, 2012-06-06, 2012-06-07, 2012-06-08, 2012-06-09, 2012-06-10, 2012-06-11, 2012-06-12, 2012-06-13, 2012-06-14, 2012-06-15, 2012-06-16, 2012-06-17, 2012-06-18, 2012-06-19, 2012-06-20, 2012-06-21, 2012-06-22, 2012-06-23, 2012-06-24, 2012-06-25, 2012-06-26, 2012-06-27, 2012-06-28, 2012-06-29, 2012-06-30, 2012-07-01, 2012-07-02, 2012-07-03, 2012-07-04, 2012-07-05, 2012-07-06, 2012-07-07, 2012-07-08, 2012-07-09, 2012-07-10, 2012-07-11, 2012-07-12, 2012-07-13, 2012-07-14, 2012-07-15, 2012-07-16, 2012-07-17, 2012-07-18, 2012-07-19, 2012-07-20, 2012-07-21, 2012-07-22, 2012-07-23, 2012-07-24, 2012-07-25, 2012-07-26, 2012-07-27, 2012-07-28, 2012-07-29, 2012-07-30, 2012-07-31, 2012-08-01, 2012-08-02, 2012-08-03, 2012-08-04, 2012-08-05, 2012-08-06, 2012-08-07, 2012-08-08, 2012-08-09, 2012-08-10, 2012-08-11, 2012-08-12, 2012-08-13, 2012-08-14, 2012-08-15, 2012-08-16, 2012-08-17, 2012-08-18, 2012-08-19, 2012-08-20, 2012-08-21, 2012-08-22, 2012-08-23, 2012-08-24, 2012-08-25, 2012-08-26, 2012-08-27, 2012-08-28, 2012-08-29, 2012-08-30, 2012-08-31, 2012-09-01, 2012-09-02, 2012-09-03, 2012-09-04, 2012-09-05, 2012-09-06, 2012-09-07, 2012-09-08, 2012-09-09, 2012-09-10, 2012-09-11, 2012-09-12, 2012-09-13, 2012-09-14, 2012-09-15, 2012-09-16, 2012-09-17, 2012-09-18, 2012-09-19, 2012-09-20, 2012-09-21, 2012-09-22, 2012-09-23, 2012-09-24, 2012-09-25, 2012-09-26, 2012-09-27, 2012-09-28, 2012-09-29, 2012-09-30, 2012-10-01, 2012-10-02, 2012-10-03, 2012-10-04, 2012-10-05, 2012-10-06, 2012-10-07, 2012-10-08, 2012-10-09, 2012-10-10, 2012-10-11, 2012-10-12, 2012-10-13, 2012-10-14, 2012-10-15, 2012-10-16, 2012-10-17, 2012-10-18, 2012-10-19, 2012-10-20, 2012-10-21, 2012-10-22, 2012-10-23, 2012-10-24, 2012-10-25, 2012-10-26, 2012-10-27, 2012-10-28, 2012-10-29, 2012-10-30, 2012-10-31, 2012-11-01, 2012-11-02, 2012-11-03, 2012-11-04, 2012-11-05, 2012-11-06, 2012-11-07, 2012-11-08, 2012-11-09, 2012-11-10, 2012-11-11, 2012-11-12, 2012-11-13, 2012-11-14, 2012-11-15, 2012-11-16, 2012-11-17, 2012-11-18, 2012-11-19, 2012-11-20, 2012-11-21, 2012-11-22, 2012-11-23, 2012-11-24, 2012-11-25, 2012-11-26, 2012-11-27, 2012-11-28, 2012-11-29, 2012-11-30, 2012-12-01, 2012-12-02, 2012-12-03, 2012-12-04, 2012-12-05, 2012-12-06, 2012-12-07, 2012-12-08, 2012-12-09, 2012-12-10, 2012-12-11, 2012-12-12, 2012-12-13, 2012-12-14, 2012-12-15, 2012-12-16, 2012-12-17, 2012-12-18, 2012-12-19, 2012-12-20, 2012-12-21, 2012-12-22, 2012-12-23, 2012-12-24, 2012-12-25, 2012-12-26, 2012-12-27, 2012-12-28, 2012-12-29, 2012-12-30, 2012-12-31]
```

Columns as sequence

```
(take 2 (api/columns ds))
```


Columns

Rows

Groups

Aggregate

Order

Unique

Missing

Join/Split Columns

Fold/Unroll Rows

Reshape

Join/Concat