Google Summer of Code 2019 Proposal
DataFrame improvements

Author: Atharva Khare
<khareatharva@gmail.com>
<*http://github.com/AtharvaKhare*>

Motivation

DataFrames were introduced in GSOC 2017 and the library has grown since then. However, it is missing advanced features (as found in pandas python library) which make it easy to manipulate data through a easy-to-use API. Features such as joins, multiple export options, handling missing data, applying blocks on dataframes, additional mathematical operators will make use of the library easier.
Additionally, having a easy-to-lookup documentation and tutorials on popular datasets will help make the user comfortable with the API.

This project aims at improving functionality of the dataframe objects along with improving the documentation associated with it.

Goals

Extending Dataframe APIs:

1. Adding import/export to JSON format
2. Adding additional operators:
    a. >, <, >=, <=, ==
    b. Mod operator
    c. Pow operator
3. Merging dataframes and dataseries:
    a. Left join, Right join, Inner join
4. Handling missing data:
    a. dropping nil rows
    b. Filing nil cells with uniform value
    c. Filing “NA/Null/nil/?” cells with dictionary values
    d. Interpolate [mean, median, mode, 0, initialValue(same as `b`)]
5. Adding mathematical operations on DF and DS:
    a. Correlation (pearson, spearman, kendall)
    b. Count (non-NA elements in a frame/series)
    c. Covariance
    d. Cumulative [min, max, product, sum]
    e. Clip [lower/upper] (floor/ceiling function)
6. Toy dataset fetcher

Documentation and Tutorials:

1. Adding additional tests for CsvReader
2. Tutorials on analysis carried out on toy datasets
3. Adding examples and documentations strings
4. Host tutorials and documentation as html and make it SEO-friendly

Implementation Details

Implementation will be carried out in four phases; one in community bonding period and three aligned with the evaluations.

Phase 0: Community Bonding Period *(May 6 - May 27)*

I would like to spend this time on improving the proposal's details by incorporating feedbacks of the community along with getting familiar with the codebase.
Adding examples and documentation strings on existing APIs will be done in this period.

Deliverables:

1. Fix Issue#48, Issue#23.
2. Plan platform for API pages (eg: `Pillar -> HTML -> Github Pages`)

Phase 1: Initial coding phase *(May 27 - June 23)*

`DataFrameJsonReader` and `DataFrameJsonWriter` will be implemented first, along with its tests. I might use the `NeoJson` library to read to/from files. Along with these classes, an example demonstrating use of these, preferably using `Zinc` library to fetch JSON from an internet API (Booklet Issue#1)

Operators like `>, <, >=, <=, ==, mod, pow` will be extension to the current `+, -, *, /` operators, and help perform actions such as:

```
a := DataSeries withValues: #(1 2 3 4) name: #a.
Transcript show: a < 3.
```

```
a DataSeries(1->true 2->true 3->false 4->false)
```

APIs for merging of dataframes will also be done in this phase.

Deliverables:

1. JSON read/write capabilities
2. Adding Boolean operators >, <, >=, <=, == and arithmetic operators mod, pow
3. Left join, Right join, Inner join between dataframes

Phase 2: Intermediate coding phase *(June 23 - July 21)*

I would like to dedicate this phase primarily to make the library capable of handling missing data. The library will be capable of:

```
df dropNull.
df_series fillNullWithValue: 0.
df fillNullWithDictionary: <Dict>.
df fillNaWithMean.
df fillNaWithMedian.
df fillNaWithMode.
```

Next will be capability to handle missing data while initialization. As described in Issue#21, the library will be capable to handle cases like the following:

```
DataFrame fromRows: #(
    (1 2 3)
    (4 5)).
```

After this has been complete, it should work with DataFrameTypeDetector. Issue#66 Issue#14

In the remaining week, correlation methods along with test cases and examples will be implemented.

Deliverables:

1. Handling null values of dataframe/series
2. Initializing incomplete dataframe with null values
3. Compatibility with DataFrameTypeDetector
4. Correlation methods along with documentation

Phase 3: Final coding phase *(July 21 - Aug 19)*

Implementation of Clip(floor/ceiling functions), Count(CountNulls), Covariance and Cumulative transformations will be done in the first week.

Next, I'll focus on recreating functionality similar to `sklearn.datasets.load_xyz()` :

```
iris := Datasets loadIris.
```

Tests to ensure correct retrieval of datasets will be written, along with examples.

The remaining days will be spent on writing tutorials on toy datasets as well as datasets requested by the community.
The existing documentation will be copied to the choice of delivery decided in Phase 0 (such as pillar), and might be hosted on Github Pages.

The following future work will be carried on if time permits.

Future work

1. Adding SQL backend support (initially using SQLite3 drivers)
2. Creating an Excel driver for reading/writing xlsx files

Deployment Plan

Deliverable	Estimated time in days	Week Number	Phase
Explore the codebase	-	-	Community bonding period
Add examples and strings to existing classes and methods			
GSoC officially begins (May 27th)			
JSON Read/Write	7	1	Phase 1
JSON + CSV tests and documentation	4	2	
Extending Arithmetic and Boolean operators	7	2 and 3	
Joins on dataframes	10	3 and 4	
First Evaluation			
Handling missing data	7	5	Phase 2
Writing tests and documenting methods to handle missing data	7	6	
Initializing with missing data + testing compatibility with other classes	4	7	
Correlation methods along with testcases and examples	7	7 and 8	
Buffer period	5	8	
Second Evaluation			
Clip, CountNulls, Covariance, Cumulative transformations	7	9	Phase3
Toy Dataset library	10	10 and 11	
Documenting use of toy dataset library	4	11	
Creating tutorials demonstrating analysis using available APIs on datasets	7	12	