dframe Documentation

Release 0.1

Ankur Gupta

Jun 13, 2017

Contents

dframe is a Python implementation of indexless dataframe data structure. dframe was built to favor ease-of-use over computational speed. It's specifically aimed to be simple and unambiguous for interactive use.

dframe provides `DataFrame` that is more similar to R's inbuilt dataframes (without the ambiguity) than Python's pandas.

Some notable differences from pandas dataframes and R dataframes are:

1. **No index/No rownames.** dframe provides indexless dataframes. There is no index and no rownames. Rows can only be indexed by row numbers (`int`) and by logicals (coming soon). This means there is no ambiguity whether row "index" or row "number" is used. There is no `.iloc` or `.loc`; use familiar, regular indexing, for example, `df[0:3, 2]`.
2. **String column names only.** Dataframes in dframe can only have have column names that are string type (`str`, `unicode`). You can never have a column named with an `int` such as `1`.
3. **No duplicate column names.** Dataframes in dframe cannot have duplicate column names. This means there can only be **one** column named `colname` and `df['colname']` will **always** return exactly one column without any ambiguity.
4. **Simple stacking operations.** Since there is no index, there is no ambiguity when performing simple matrix-like horizontal and vertical stacking operations. You will not have to reindex dataframes to horizontal stack them. No index cleanup required after a stacking operation. Database-style merge operations are completely separate from matrix-like stacking operations – use whichever one suits the task at hand. No need to cast matrix-like stacking operations into database-style merge operations.
5. **Almost first-class missing value support.** dframe handles missing data using Python's in-built `None` instead of defining a new missing value type. You can have missing values in any `dtype`, not just in `float`. Marking one element of a column as missing value will not change the `dtype` of that column.
6. **No Series vs DataFrame.** There is no object like `Series` in pandas. In dframe, `df['colname']` returns a `list`-like `Array` object. `Array` is sub-classed from `list` and behaves like a `list` is most ways. If you would rather have a dataframe with only one column, use `df[['colname']]`.