
pyddf Documentation

Release 1.4.9

Adatao

January 12, 2016

1	ddf package	3
1.1	Submodules	3
1.2	ddf.conf module	3
1.3	ddf.dataframe module	3
1.4	ddf.ddf_manager module	5
1.5	ddf.gateway module	6
1.6	ddf.util module	6
2	Indices and tables	7
	Python Module Index	9
	Index	11

Contents:

1.1 Submodules

1.2 ddf.conf module

`ddf.conf.find_ddf()`

1.3 ddf.dataframe module

Created on Jun 22, 2014

@author: nhanitvn

class `ddf.dataframe.DistributedDataFrame` (*jddf*)

Bases: `object`

A Distributed Data Frame, the basic abstraction in DistributedDataFrame library.

aggregate (*aggr_columns, by_columns*)

Split the DistributedDataFrame into sub-sets by some columns and perform aggregation on some columns within each sub-set

colnames

List the column names of this DDF

Returns a list of strings

cols

Get number of columns of this DDF

Deprecated since version Use: `ncol()` instead.

Returns an int

coltypes

The types of all columns of this DDF

Returns a list of strings

correlation (*col1, col2*)

Correlation coefficient of a DistributedDataFrame's two numeric columns

Parameters

- **col1** – a numeric column
- **col2** – a numeric column

Returns a float

drop_na (*axis=u'row', inplace=False*)
Drop NA values

Parameters

- **axis** – the axis by which to drop if NA value exists, ROW represents by Row as default, COLUMN is column.
- **inplace** – whether to treat the ddf inplace, default is FALSE.

Returns a DDF with no NA values.

five_nums ()

Calculate Turkey five number for numeric columns :return: a pandas DataFrame in which each column is a vector containing the summary information

head (*n=10*)

Return this DistributedDataFrame's some first rows :param n: number of rows to get

join (*other, by=None, by_left=None, by_right=None, join_type=u'inner'*)

Join two DistributedDataFrames together.

Join, like merge, is designed for the types of problems where you would use a sql join.

Parameters

- **other** ([DistributedDataFrame](#)) – the other DDF
- **by** – columns used for joining. Default is common columns of *self* and *other*.
- **by_left** – columns used for joining. Default is common columns of *self* and *other*.
- **by_right** – columns used for joining. Default is common columns of *self* and *other*.
- **join_type** – type of join: inner (default), left, right, full or leftsemi - inner: only rows with matching keys in both *self* and *other*. - left: all rows in *self*, adding matching columns from *other*. - right: all rows in *other*, adding matching columns from *self*. - full: all rows in *self* with matching columns in *other*,
then the rows of *other* that don't match *self*
- leftsemi: only rows in *self* with matching keys in *other*.

Returns a DistributedDataFrame

mean (*column*)

Calculate Mean value of DDF Column

Parameters **column** – the column name or index

Returns the mean value

name

Get name of this DDF :return: a str

ncol

Get number of columns of this DDF

Returns an int

nrow

Get number of rows of this DDF

Returns an int

project (*column_names*)

Project on some columns and return a new DistributedDataFrame

rows

Get number of rows of this DDF

Deprecated since version Use: `nrow()` instead.

Returns an int

sample (*size*, *replacement=False*, *seed=123*)

Get a sample of this DistributedDataFrame and return a list of strings

Parameters

- **size** – number of samples
- **replacement** – sample with or without replacement
- **seed** – random seed

Returns a pandas DataFrame

sample2ddf (*fraction*, *replacement=False*, *seed=123*)

Get a sample of this DistributedDataFrame and return a new DistributedDataFrame

Parameters

- **fraction** – fraction to take sample, has to be in the (0, 1] range
- **replacement** – sample with or without replacement
- **seed** – random seed

Returns a new DistributedDataFrame

summary ()

Return a statistical summary of a DistributedDataFrame's columns :return: a pandas DataFrame containing summaries

var (*column*)

Compute variance and standard deviation of a DistributedDataFrame's column

Parameters **column** – the column name or index

Returns a tuple of two elements (variance, standard deviation)

1.4 ddf.ddf_manager module

Created on Jun 22, 2014

@author: nhanitvn

class `ddf.ddf_manager.DDFManager` (*engine_name*)

Bases: `object`

Main entry point for DDF functionality. A SparkDDFManager can be used to create DDFs that are implemented for Spark framework.

get_ddf_by_name (*ddf_name*)

Get a DDF object using its name :param ddf_name: the name of the DDF object to be retrieved :return: a DDF object

list_ddfs ()

List all the DDFs

Returns list of DDF objects

set_ddf_name (*ddf*, *name*)

Set a name for the given DDF

Parameters

- **ddf** – the DDF object
- **name** – name of the DDF

Returns nothing

shutdown()

Shut down the DDF Manager

sql (*command*, *data_source=u'spark'*)

Execute a sql command and return a list of strings :param *command*: the sql command to run :param *data_source*: data source

sql2ddf (*command*, *data_source=u'spark'*)

Create a DistributedDataFrame from an sql command. :param *command*: the sql command to run :param *data_source*: data source :return: a DDF

1.5 ddf.gateway module

`ddf.gateway.compute_classpath(root_path)`

`ddf.gateway.current_gateway()`

`ddf.gateway.list_jar_files(path)`

`ddf.gateway.pre_exec_func()`

`ddf.gateway.start_gateway_server()`

1.6 ddf.util module

`ddf.util.convert_column_types(df, column_types, raise_on_error=False)`

Convert a dataframe into data types specified in *column_types* :param *df*: a data frame containing the sampled data :type *df*: `pd.DataFrame` :param *column_types*: the types of columns, in pE terminology :param *raise_on_error*: :return: a correctly typed data frame

`ddf.util.parse_column(col_names, column)`

Convert a column to index :param *col_names*: list of column names :param *column*: column name or index :return: column index

`ddf.util.parse_column_str(col_names, column)`

Validate a column name or index, return the column name :param *col_names*: list of column names :param *column*: column name or index :return: column index

`ddf.util.parse_ddf_data(rows, colnames, coltypes)`

`ddf.util.parse_sql_result(java_result)`

`ddf.util.to_bool(x)`

Try our best to make *x* into a boolean value :param *x*: the value to be converted :return: a boolean value

`ddf.util.to_java_list(ls, gateway_client)`

Convert a python list into java list :param *ls*: python list to be converted :param *gateway_client*: gateway client object :return: java list

`ddf.util.to_python_type(t)`

`ddf.util.validate_column_generic(col_names, column, get_name=True)`

Validate a column name or index, return the column name :param *col_names*: list of column names :param *column*: column name or index :param *get_name*: :return: column index

Indices and tables

- `genindex`
- `modindex`
- `search`

d

- `ddf`, 3
- `ddf.conf`, 3
- `ddf.dataframe`, 3
- `ddf.ddf_manager`, 5
- `ddf.gateway`, 6
- `ddf.util`, 6

A

aggregate() (ddf.dataframe.DistributedDataFrame method), 3

C

colnames (ddf.dataframe.DistributedDataFrame attribute), 3

cols (ddf.dataframe.DistributedDataFrame attribute), 3

coltypes (ddf.dataframe.DistributedDataFrame attribute), 3

compute_classpath() (in module ddf.gateway), 6

convert_column_types() (in module ddf.util), 6

correlation() (ddf.dataframe.DistributedDataFrame method), 3

current_gateway() (in module ddf.gateway), 6

D

ddf (module), 3

ddf.conf (module), 3

ddf.dataframe (module), 3

ddf.ddf_manager (module), 5

ddf.gateway (module), 6

ddf.util (module), 6

DDFManager (class in ddf.ddf_manager), 5

DistributedDataFrame (class in ddf.dataframe), 3

drop_na() (ddf.dataframe.DistributedDataFrame method), 3

F

find_ddf() (in module ddf.conf), 3

five_nums() (ddf.dataframe.DistributedDataFrame method), 4

G

get_ddf_by_name() (ddf.ddf_manager.DDFManager method), 5

H

head() (ddf.dataframe.DistributedDataFrame method), 4

J

join() (ddf.dataframe.DistributedDataFrame method), 4

L

list_ddfs() (ddf.ddf_manager.DDFManager method), 5

list_jar_files() (in module ddf.gateway), 6

M

mean() (ddf.dataframe.DistributedDataFrame method), 4

N

name (ddf.dataframe.DistributedDataFrame attribute), 4

ncol (ddf.dataframe.DistributedDataFrame attribute), 4

nrow (ddf.dataframe.DistributedDataFrame attribute), 4

P

parse_column() (in module ddf.util), 6

parse_column_str() (in module ddf.util), 6

parse_ddf_data() (in module ddf.util), 6

parse_sql_result() (in module ddf.util), 6

pre_exec_func() (in module ddf.gateway), 6

project() (ddf.dataframe.DistributedDataFrame method), 4

R

rows (ddf.dataframe.DistributedDataFrame attribute), 4

S

sample() (ddf.dataframe.DistributedDataFrame method), 5

sample2ddf() (ddf.dataframe.DistributedDataFrame method), 5

set_ddf_name() (ddf.ddf_manager.DDFManager method), 5

shutdown() (ddf.ddf_manager.DDFManager method), 6

sql() (ddf.ddf_manager.DDFManager method), 6

sql2ddf() (ddf.ddf_manager.DDFManager method), 6

start_gateway_server() (in module ddf.gateway), 6

summary() (ddf.dataframe.DistributedDataFrame method), 5

T

to_bool() (in module ddf.util), 6

to_java_list() (in module ddf.util), 6

to_python_type() (in module ddf.util), 6

V

`validate_column_generic()` (in module `ddf.util`), [6](#)

`var()` (`ddf.dataframe.DistributedDataFrame` method), [5](#)