pyspssio

Release 0.4.3+1.ga4cedda.dirty

Steven Spector

Jan 12, 2024

CONTENTS

Please use the navigation panel to access the README and details about pyspssio's features.

CHAPTER

ONE

README

1.1 Overview

pyspssio is a python package for reading and writing SPSS (.sav and .zsav) files to/from pandas dataframes.

This package uses the I/O Module for SPSS Statistics v27 available at https://www.ibm.com/.

WARNING: This is an early release with limited testing. Use with caution.

1.2 Links

- PyPI
- GitHub
- Read the Docs

1.3 Motivation

Main reason for creating this package is to fill gaps by other similar packages.

`savReaderWriter`

- doesn't support python > 3.5
- not particularly user friendly

`pyreadstat`

- doesn't read or write multi response set definitions
- datetime conversion quirks
- issues reading/writing long string variables (https://github.com/Roche/pyreadstat/issues/119)

`pyspssio` supports recent versions of python and can read/write most SPSS file metadata properties. The `usecols` argument when reading files also accepts a callable for more flexible variable selection.

1.4 Basic Usage

Installation

```
pip install pyspssio
```

Import

```
import pyspssio
```

1.4.1 Reading

Read data and metadata

```
df, meta = pyspssio.read_sav("spss_file.sav")
```

Read metadata only

```
meta = pyspssio.read_metadata("spss_file.sav")
```

Read data in chunks of `chunksize` (number of rows/records)

```
for df in pyspssio.read_sav("spss_file.sav", chunksize=1000):
#   do something
```

Note: metadata is not returned when reading in chunks

1.4.2 Writing

Write dataframe to file.

```
pyspssio.write_sav("spss_file.sav", df)
```

1.4.3 Appending

Append existing SPSS file with new records.

```
pyspssio.write_sav("spss_file.sav", df)
```

Note: Cannot modify metadata when appending new records. Be careful with strings that might be longer than the already defined width as they may be automatically truncated without warning.

1.5 Other Notes

1.5.1 Date/Time Variables

Date and datetime variables - These are converted to/from full datetime objects, even for formats like DATE, QYR, and WKYR which don't display a time component. Users can opt to use Pandas' `.dt` accessor to extract specific components or force a specific accuracy (e.g., minute, day, hour) after reading the data (ex. `.dt.floor`). The `var_formats` and/or `var_formats_tuple` metadata attributes can be used to see the original SPSS formats.

Time variables - These are converted to/from timestamp objects.

Python/Pandas stores datetimes in nanseconds while SPSS stores them in seconds. Due to conversions that must take place, there may be some small (ms) discrepancies between an original dataframe used to write an SPSS file and a dataframe read back from the same SPSS file.

1.6 I/O Module Procedures

List of available I/O module procedures and class for which they fall under. See official documentation for details on each one.

Some of these procedures are implemented as hidden methods referenced within a more generalized function/property. For example, instead of calling `spssSetVarLabel` manually for each variable, users should assign all variable labels at once by setting `self.var_labels = {var1: label1, var2: label2, ...}`.

All I/O module procedures can be accessed directly with `self.spssio.[procedure]`.

1.6.1 SPSSFile

spssOpenRead

spssCloseRead

spssOpenWrite

spssCloseWrite

spssOpenAppend

spssCloseAppend

spssHostSysmisVal

spssLowHighVal

spssSetLocale

spssGetInterfaceEncoding

spssSetInterfaceEncoding

spssGetFileEncoding

spssIsCompatibleEncoding

spssGetCompression

spssSetCompression

spssGetReleaseInfo

spssGetNumberofCases

spssGetNumberofVariables

1.6.2 Header

spssGetFileAttributes

spssSetFileAttributes

spssGetVarNames

spssSetVarName

spssGetVarHandle

spssGetVarPrintFormat

spssSetVarPrintFormat

spssSetVarWriteFormat

spssGetVarMeasureLevel

spssSetVarMeasureLevel

spssGetVarAlignment

spssSetVarAlignment

spssGetVarColumnWidth

spssSetVarColumnWidth

spssGetVarLabelLong

spssSetVarLabel

spssGetVarRole

spssSetVarRole

spssGetVarCValueLabels

spssSetVarCValueLabel

spssGetVarNValueLabels

spssSetVarNValueLabel

spssGetVarCMissingValues

spssSetVarCMissingValues

spssGetVarNMissingValues

spssSetVarNMissingValues

spssGetMultRespCount

spssGetMultRespDefs

spssGetMultRespDefsEx - replaces spssGetMultRespDefs

spssSetMultRespDefs

spssAddMultRespDefExt

spssGetCaseSize

spssGetCaseWeightVar

spssSetCaseWeightVar

spssGetVarAttributes

spssSetVarAttributes

spssGetVarCompatName

spssGetVariableSets

spssSetVariableSets

spssCommitHeader

1.6.3 Reader

spssSeekNextCase

spssWholeCaseIn

1.6.4 Writer

spssWholeCaseOut

spssSetValueChar

spssSetValueNumeric

spssCommitCaseRecord

1.6.5 Not Implemented (yet)

spssAddMultRespDefC

spssAddMultRespDefN

spssGetMultRespDefByIndex

spssConvertDate - manual conversion instead

spssConvertSPSSDate - manual conversion instead

spssConvertSPSSTime - manual conversion instead

spssConvertTime - manual conversion instead

spssCopyDocuments

spssGetDEWFirst

spssGetDEWGUID

spssGetDewInfo

spssGetDEWNext

spssSetDEWFirst

spssSetDEWGUID

spssSetDEWNext

spssGetDateVariables

spssGetEstimatedNofCases

spssGetFileAttribute - uses spssGetFileAttributes instead

spssGetFileCodePage

spssGetIdString

spssGetSystemString

spssGetTextInfo

spssGetTimeStamp

spssGetValueChar - uses spssWholeCaseIn instead

spssGetValueNumeric - uses spssWholeCaseIn instead

spssAddVarAttribute - uses spssSetVarAttributes instead

spssGetVarCValueLabel - uses spssGetVarCValueLabels instead

spssGetVarCValueLabelLong - uses spssGetVarCValueLabels instead

spssGetVarInfo

spssGetVarLabel - uses spssGetVarLabelLong instead

spssGetVarNValueLabel - uses spssGetVarNValueLabels instead

spssGetVarNValueLabelLong - uses spssGetVarNValueLabels instead

spssGetVarWriteFormat - uses spssGetVarPrintFormat instead (print/write formats tied together)

spssOpenAppendEx

spssOpenReadEx

spssOpenWriteCopy

spssOpenWriteCopyEx

spssOpenWriteCopyExDict

spssOpenWriteCopyExFile

spssOpenWriteEx

spssQueryType7

spssReadCaseRecord - uses spssWholeCaseIn instead

spssSetDateVariables

spssSetIdString

spssSetTempDir

spssSetTextInfo

spssSetVarCValueLabels - uses spssSetVarCValueLabel instead

spssSetVarNValueLabels - uses spssSetVarNValueLabel instead

spssSysmisVal - uses spssHostSysmisVal instead

spssValidateVarname

CHAPTER

TWO

FUNCTIONS

2.1 Reading

read_sav(spss_file[, row_offset, row_limit, ...])	Read data and metadata from SPSS file
read_metadata(spss_file[, usecols, locale])	Reads metadata attributes from SPSS file

2.1.1 pyspssio.read_sav

pyspssio.**read_sav**(*spss_file*, *row_offset=0*, *row_limit=None*, *usecols=None*, *convert_datetimes=True*, *include_user_missing=True*, *chunksize=None*, *locale=None*, *string_nan=''*)

Read data and metadata from SPSS file

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **row_offset** (int (default: 0)) – Number of rows to skip
- **row_limit** (int (default: None)) – Maximum number of rows to return
- **usecols** (Union[list, tuple, str, callable, None] (default: None)) – Columns to use (None for all columns)
- **convert_datetimes** (bool (default: True)) – Convert SPSS datetimes to Python/Pandas datetime columns; False returns seconds from October 15, 1582 (SPSS start date)
- **include_user_missing** (bool (default: True)) – Whether to keep user missing values or replace them with NaN (numeric) and "" (strings)
- **chunksize** (int (default: None)) – Number of rows to return per chunk
- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode
- **string_nan** (Any (default: '')) – Value to return for empty strings

Return type

Union[Tuple[DataFrame, dict], Generator[DataFrame, None, None]]

Returns

- *tuple* – DataFrame, metadata
- *generator* – DataFrame(s) with chunksize number of rows (only if chunksize is specified)

Examples

Read data and metadata:

```
df, meta = pyspssio.read_sav("spss_file.sav")
```

Read metadata only:

```
meta = pyspssio.read_metadata("spss_file.sav")
```

Read data in chunks of chunksize (number of rows/records):

```
for df in pyspssio.read_sav("spss_file.sav", chunksize=1000):
    # do something
```

2.1.2 pyspssio.read_metadata

pyspssio.**read_metadata**(*spss_file*, *usecols=None*, *locale=None*)

Reads metadata attributes from SPSS file

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **usecols** (Union[list, tuple, str, callable, None] (default: None)) – Columns to use (None for all columns)
- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode

Returns

Header properties (see Header class for more detail)

Return type

dict

Examples

```
>>> meta = pyspssio.read_metadata("spss_file.sav")
```

pyspssio.**read_sav**(*spss_file*, *row_offset=0*, *row_limit=None*, *usecols=None*, *convert_datetimes=True*, *include_user_missing=True*, *chunksize=None*, *locale=None*, *string_nan=''*)

Read data and metadata from SPSS file

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **row_offset** (int (default: 0)) – Number of rows to skip
- **row_limit** (int (default: None)) – Maximum number of rows to return
- **usecols** (Union[list, tuple, str, callable, None] (default: None)) – Columns to use (None for all columns)
- **convert_datetimes** (bool (default: True)) – Convert SPSS datetimes to Python/Pandas datetime columns; False returns seconds from October 15, 1582 (SPSS start date)

- **include_user_missing** (bool (default: True)) – Whether to keep user missing values or replace them with NaN (numeric) and "" (strings)
- **chunksize** (int (default: None)) – Number of rows to return per chunk
- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode
- **string_nan** (Any (default: '')) – Value to return for empty strings

Return type
Union[Tuple[DataFrame, dict], Generator[DataFrame, None, None]]

Returns

- *tuple* – DataFrame, metadata
- *generator* – DataFrame(s) with chunksize number of rows (only if chunksize is specified)

Examples

Read data and metadata:

```
df, meta = pyspssio.read_sav("spss_file.sav")
```

Read metadata only:

```
meta = pyspssio.read_metadata("spss_file.sav")
```

Read data in chunks of chunksize (number of rows/records):

```
for df in pyspssio.read_sav("spss_file.sav", chunksize=1000):
    # do something
```

pyspssio.**read_metadata**(*spss_file*, *usecols=None*, *locale=None*)

Reads metadata attributes from SPSS file

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **usecols** (Union[list, tuple, str, callable, None] (default: None)) – Columns to use (None for all columns)
- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode

Returns
Header properties (see Header class for more detail)

Return type
dict

Examples

```
>>> meta = pyspssio.read_metadata("spss_file.sav")
```

2.2 Writing

write_sav(spss_file, df[, metadata, ...])	Write SPSS file (.sav or .zsav) from DataFrame
append_sav(spss_file, df[, locale])	Append existing SPSS file (.sav or .zsav) with additional records

2.2.1 pyspssio.write_sav

pyspssio.**write_sav**(*spss_file*, *df*, *metadata=None*, *unicode=True*, *locale=None*, ***kwargs*)

Write SPSS file (.sav or .zsav) from DataFrame

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **df** (DataFrame) – DataFrame
- **metadata** (dict (default: None)) – Dictionary of Header attributes to use (see Header class for more detail)
- **unicode** (bool (default: True)) – Whether to write the file in unicode (True) or codepage (False) mode
- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode
- ****kwargs** – Additional arguments, including individual metadata attributes. Note that metadata attributes supplied here take precedence.

Return type

None

Examples

```
>>> pyspssio.write_sav("spss_file.sav", df, metadata=meta)
```

2.2.2 pyspssio.append_sav

pyspssio.**append_sav**(*spss_file*, *df*, *locale=None*, ***kwargs*)

Append existing SPSS file (.sav or .zsav) with additional records

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **df** (DataFrame) – DataFrame

- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode
- ****kwargs** – Additional arguments

Return type
None

Notes

Cannot modify metadata when appending new records. Be careful with strings that might be longer than the allowed width.

It may or may not be necessary to manually set locale since file encoding information is obtained from the SPSS header information.

Examples

```
>>> pyspssio.append_sav("spss_file.sav", df)
```

pyspssio.**write_sav**(*spss_file*, *df*, *metadata=None*, *unicode=True*, *locale=None*, ***kwargs*)

Write SPSS file (.sav or .zsav) from DataFrame

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **df** (DataFrame) – DataFrame
- **metadata** (dict (default: None)) – Dictionary of Header attributes to use (see Header class for more detail)
- **unicode** (bool (default: True)) – Whether to write the file in unicode (True) or codepage (False) mode
- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode
- ****kwargs** – Additional arguments, including individual metadata attributes. Note that metadata attributes supplied here take precedence.

Return type
None

Examples

```
>>> pyspssio.write_sav("spss_file.sav", df, metadata=meta)
```

pyspssio.**append_sav**(*spss_file*, *df*, *locale=None*, ***kwargs*)

Append existing SPSS file (.sav or .zsav) with additional records

Parameters

- **spss_file** (str) – SPSS filename (.sav or .zsav)
- **df** (DataFrame) – DataFrame

- **locale** (str (default: None)) – Locale to use when I/O module is operating in codepage mode
- ****kwargs** – Additional arguments

Return type
None

Notes

Cannot modify metadata when appending new records. Be careful with strings that might be longer than the allowed width.

It may or may not be necessary to manually set locale since file encoding information is obtained from the SPSS header information.

Examples

```
>>> pyspssio.append_sav("spss_file.sav", df)
```

CHAPTER

THREE

CLASSES

SPSSFile(spss_file[, mode, unicode, locale])	Base class for opening and closing SPSS files
Header(*args, **kwargs)	Class for getting and setting metadata attributes
Reader(*args[, row_offset, row_limit, ...])	Class for reading metadata and data
Writer(*args, **kwargs)	Class for writing SPSS file

3.1 pyspssio.SPSSFile

class pyspssio.**SPSSFile**(*spss_file*, *mode='rb'*, *unicode=True*, *locale=None*)

Bases: `object`

Base class for opening and closing SPSS files

__init__(*spss_file*, *mode='rb'*, *unicode=True*, *locale=None*)

Methods

__init__(spss_file[, mode, unicode, locale])	
close()	Close file
open()	Open file
set_locale(locale)	Set I/O module to a specific locale

Attributes

case_count	Number of cases
compression	Compression level
file_encoding	File encoding reported by I/O module
interface_encoding	I/O interface mode (Unicode or code page)
is_compatible_encoding	Check encoding compatibility
release_info	Basic file information
var_count	Number of variables

property interface_encoding: int

I/O interface mode (Unicode or code page)

- 0 = SPSS_ENCODING_CODEPAGE
- 1 = SPSS_ENCODING_UTF8

property file_encoding: str

File encoding reported by I/O module

set_locale(*locale*)

Set I/O module to a specific locale

Return type
str

property is_compatible_encoding: bool

Check encoding compatibility

From I/O module documentation: "This function determines whether the file's encoding is compatible with the current interface encoding. The result value ... will be false when reading a code page file in UTF-8 mode, when reading a UTF-8 file in code page mode when reading a code page file encoded in other than the current locale's code page, or when reading a file with numbers represented in reverse bit order. If the encoding is incompatible, data stored in the file by other applications, particularly Data Entry for Windows, may be unreliable."

open()

Open file

Returns file handle that is used for most other I/O module functions.

Return type
int

Notes

Filenames are always encoded in UTF-8 regardless of interface mode and locale settings. This is to avoid issues where a filename uses special characters that aren't available in the encoding defined by the file itself. For example, a Windows-1252 .sav file which uses Chinese (or other special multibyte characters) in its filename.

close()

Close file

property compression: int

Compression level

- 0 = No compression
- 1 = SAV
- 2 = ZSAV

property release_info: dict

Basic file information

- release number
- release subnumber
- fixpack number

- machine code
- floating-point representation code
- compression scheme code
- big/little-endian code
- character representation code

property var_count: int

Number of variables

property case_count: int

Number of cases

3.2 pyspssio.Header

class pyspssio.**Header**(**args*, ***kwargs*)

Bases: *SPSSFile*

Class for getting and setting metadata attributes

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)	
close()	Close file
commit_header()	Finalize metadata
open()	Open file
set_locale(locale)	Set I/O module to a specific locale

Attributes

case_count	Number of cases
case_size	Record case size (in bytes)
case_weight_var	Case weight variable
compression	Compression level
file_attributes	Arbitrary user-defined file attributes
file_encoding	File encoding reported by I/O module
interface_encoding	I/O interface mode (Unicode or code page)
is_compatible_encoding	Check encoding compatibility
mrsets	Multi response set definitions
mrsets_count	Number of multi response set definitions
release_info	Basic file information
var_alignments	Variable alignments
var_attributes	Variable attributes
var_column_widths	Column display widths
var_compat_names	Short (8-byte) variable names
var_count	Number of variables
var_formats	Variable formats as strings
var_formats_tuple	Variable formats as tuples in the form (type, width, decimals)
var_handles	Variable handles references
var_labels	Variable labels
var_measure_levels	Variable measure levels
var_missing_values	Missing values
var_names	Variable names
var_roles	Variable roles
var_sets	Variable sets
var_types	Variable types
var_value_labels	Variable value labels

property file_attributes: dict

Arbitrary user-defined file attributes

property var_names: list

Variable names

May return a filtered list when returned as part of a metadata object if only a subset of variables are specified to be used (e.g., usecols in read_sav).

property var_types: dict

Variable types

May return a filtered dictionary when returned as part of a metadata object if only a subset of variables are specified to be used (e.g., usecols in read_sav).

property var_handles: dict

Variable handles references

Used when calling I/O module procedures that use variable handles instead of variable names as arguments

property var_formats_tuple: dict

Variable formats as tuples in the form (type, width, decimals)

ex. (5, 8, 2) instead of F8.2

property var_formats: dict

Variable formats as strings

Use var_formats_tuple property for formats as tuples

property var_measure_levels: dict

Variable measure levels

Measure levels are returned as strings. When setting, input accepts either strings or numerics.

- 0 = unknown
- 1 = nominal
- 2 = ordinal
- 3 = scale

property var_alignments: dict

Variable alignments

Alignments are returned as strings. When setting, input accepts either strings or numerics.

- 0 = left
- 1 = right
- 2 = center

property var_column_widths: dict

Column display widths

Manually set column widths or specify 0 to use SPSS' algorithm to assign a width

property var_labels: dict

Variable labels

property var_roles: dict

Variable roles

Roles are returned as strings. When setting, input accepts either strings or numerics.

- 0 = input
- 1 = target
- 2 = both
- 3 = none
- 4 = partition
- 5 = split
- 6 = frequency
- 7 = recordid

property var_value_labels: dict

Variable value labels

Nested dictionary of variables with their value labels (if defined) as sub-dictionaries

Note: value labels only work for numeric and short string variables (length <= 8)

property mrsets_count: int

Number of multi response set definitions

Needed if using spssGetMultRespDefByIndex. Otherwise, len(mrsets) should be equivalent.

property mrsets: dict

Multi response set definitions

Multi response sets contain the following attributes

- label : set label
- is_dichotomy : whether set is dichotomous (True) or Category (False)
- counted_value : counted value for dichotomous sets
- use_category_labels : whether to use counted value labels instead of variable labels
- use_first_var_label : whether to use first var label as set label
- variable_list : list of variables in the set

Notes

mrset name must begin with a "$".

variable_list is the only required attribute. However, if this is the only included attribute, then is_dichotomy is assumed to be False.

If is_dichotomy is True, counted_value must be specified. If is_dichotomy is None and counted_value is not None, is_dichotomy is assumed to be True.

Numeric dichotomous sets only accept integers for a counted value.

use_category_labels is only applicable for dichotomous sets. Setting this to True turns the set into an "extended" mrset definition.

use_first_var_label is only applicable when use_category_labels is True. Specifying a set label when use_first_var_label is True might result in an invalid mrset definition.

Examples

Category (C) Set:

```
{"$mc_mrset": {
    "label": "This is an MC set",
    "variable_list": ["var1", "var2", "var3"]
}}
```

Dichotomous (D) Set:

```
{"$md_mrset": {
    "label": "This is an MD set",
    "counted_value": 1,
    "variable_list": ["resp1", "resp2", "resp3"]
}}
```

Dichotomous (E - Extended) Set:

```
{"$md_mrset": {
    "counted_value": 1,
    "use_category:labels": True,
    "use_first_var_label": True,
    "variable_list": ["cat1", "cat2", "cat3"]
}}
```

property case_size: int

Record case size (in bytes)

Raw number of bytes for a single case record. It can be calculated manually by adding all variable types rounded up to the nearest multiple of 8.

This is the buffer size used to read a whole case record at once. It is not necessily the number of bytes used to store a case record on disk (depending on compression).

property case_weight_var: str

Case weight variable

Variable set as the "weight" variable in SPSS. Must be a scale numeric variable.

property var_missing_values: dict

Missing values

Missing value definitions may contain three keys

1. lo = Low value used in missing range
2. hi = high value used in missing range
3. values = list of discrete values set as user missing

For missing ranges, the following keywords can be used inplace of numeric values

- low = -inf, lo, low, lowest
- high = inf, hi, high, highest

property var_attributes: dict

Variable attributes

These are arbitrary variable properties, analagous to file attributes

property var_compat_names: dict

Short (8-byte) variable names

Dictionary of variable names with their "compatible" short 8-byte counterparts

property var_sets: dict

Variable sets

These are NOT multi response sets. These variable sets are groupings of variables that can be selected in the SPSS application as a sort of view filter.

SPSS apparently may use the 8 byte compatible variable names for this property. It's currently not possible to obtain the auto-generated compatible names until the dictionary is committed, which means setting this property potentially requires first comitting a dictionary with all variables, and then rewriting it after obtaining the compatible variable names.

Set names when created in the normal SPSS application allow spaces and special characters. However, The I/O module returns an SPSS_INVALID_VARSETDEF error when these are included. When an "=" sign is included in the set name, the set name is truncated.

commit_header()

Finalize metadata

This function is used to finalize the header information before writing data. Once this function is called, no further metadata modification is allowed.

3.3 pyspssio.Reader

class pyspssio.**Reader**(**args*, *row_offset=0*, *row_limit=None*, *usecols=None*, *chunksize=None*, *convert_datetimes=True*, *include_user_missing=True*, *string_nan=''*, ***kwargs*)

Bases: *Header*

Class for reading metadata and data

__init__(**args*, *row_offset=0*, *row_limit=None*, *usecols=None*, *chunksize=None*, *convert_datetimes=True*, *include_user_missing=True*, *string_nan=''*, ***kwargs*)

Methods

__init__(*args[, row_offset, row_limit, ...])	
close()	Close file
commit_header()	Finalize metadata
open()	Open file
read_data([row_limit, convert_datetimes, ...])	Read data
set_locale(locale)	Set I/O module to a specific locale

Attributes

case_count	Number of cases
case_size	Record case size (in bytes)
case_weight_var	Case weight variable
compression	Compression level
file_attributes	Arbitrary user-defined file attributes
file_encoding	File encoding reported by I/O module
interface_encoding	I/O interface mode (Unicode or code page)
is_compatible_encoding	Check encoding compatibility
metadata	Metadata object
mrsets	Multi response set definitions
mrsets_count	Number of multi response set definitions
release_info	Basic file information
var_alignments	Variable alignments
var_attributes	Variable attributes
var_column_widths	Column display widths
var_compat_names	Short (8-byte) variable names
var_count	Number of variables
var_formats	Variable formats as strings
var_formats_tuple	Variable formats as tuples in the form (type, width, decimals)
var_handles	Variable handles references
var_labels	Variable labels
var_measure_levels	Variable measure levels
var_missing_values	Missing values
var_names	Variable names
var_roles	Variable roles
var_sets	Variable sets
var_types	Variable types
var_value_labels	Variable value labels

property metadata: dict

Metadata object

This object contains properties/attributes from the Header class mostly pertaining to variable information and data structure.

read_data(*row_limit=None*, *convert_datetimes=None*, *include_user_missing=None*)

Read data

Parameters

- **row_limit** (int (default: None)) – Maximum number of rows to return
- **convert_datetimes** (bool (default: None)) – Convert SPSS datetimes to Python/Pandas datetime columns; False returns seconds from October 15, 1582 (SPSS start date)
- **include_user_missing** (bool (default: None)) – Whether to keep user missing values or replace them with NaN (numeric) and "" (strings)

Return type

DataFrame

3.4 pyspssio.Writer

class pyspssio.**Writer**(**args*, ***kwargs*)

Bases: *Header*

Class for writing SPSS file

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)	
close()	Close file
commit_case_record()	Commit case record
commit_header()	Finalize metadata
open()	Open file
set_locale(locale)	Set I/O module to a specific locale
write_data(df, **kwargs)	Write data to file
write_data_by_val(df)	Write data by variable/value
write_header(df[, metadata])	Write metadata properties

Attributes

case_count	Number of cases
case_size	Record case size (in bytes)
case_weight_var	Case weight variable
compression	Compression level
file_attributes	Arbitrary user-defined file attributes
file_encoding	File encoding reported by I/O module
interface_encoding	I/O interface mode (Unicode or code page)
is_compatible_encoding	Check encoding compatibility
mrsets	Multi response set definitions
mrsets_count	Number of multi response set definitions
release_info	Basic file information
var_alignments	Variable alignments
var_attributes	Variable attributes
var_column_widths	Column display widths
var_compat_names	Short (8-byte) variable names
var_count	Number of variables
var_formats	Variable formats as strings
var_formats_tuple	Variable formats as tuples in the form (type, width, decimals)
var_handles	Variable handles references
var_labels	Variable labels
var_measure_levels	Variable measure levels
var_missing_values	Missing values
var_names	Variable names
var_roles	Variable roles
var_sets	Variable sets
var_types	Variable types
var_value_labels	Variable value labels

commit_case_record()

Commit case record

Call function after setting values with set_value Do not use with whole_case_out

write_header(*df*, *metadata=None*, ***kwargs*)

Write metadata properties

Parameters

- **df** (DataFrame) – DataFrame
- **metadata** (Union[dict, SimpleNamespace] (default: None)) – Dictionary of Header attributes to use (see Header class for more detail)
- ****kwargs** – Additional arguments, including individual metadata attributes. Note that metadata attributes supplied here take precedence.

write_data_by_val(*df*)

Write data by variable/value

Parameters

df (DataFrame) – DataFrame

Notes

Slower than whole_case_out Use when appending to an existing data set and variable order doesn't align

write_data(*df*, ***kwargs*)

Write data to file

Parameters

df (DataFrame) – DataFrame

CHAPTER

FOUR

EXCEPTIONS AND WARNINGS

exception pyspssio.**SPSSError**

Bases: Exception

Capture I/O module errors

exception pyspssio.**SPSSWarning**

Bases: Warning

Capture I/O module warnings

CHAPTER

FIVE

CONSTANTS

5.1 Max Lengths

pyspssio.SPSS_MAX_VARNAME
64

pyspssio.SPSS_MAX_SHORTVARNAME
8

pyspssio.SPSS_MAX_SHORTSTRING
8

pyspssio.SPSS_MAX_IDSTRING
64

pyspssio.SPSS_MAX_LONGSTRING
32767

pyspssio.SPSS_MAX_VALLABEL
120

pyspssio.SPSS_MAX_VARLABEL
256

pyspssio.SPSS_MAX_ENCODING
64

pyspssio.SPSS_MAX_7SUBTYPE
40

pyspssio.SPSS_MAX_PASSWORD
10

5.2 Missing Values

pyspssio.SPSS_NO_MISSVAL
0

pyspssio.SPSS_ONE_MISSVAL
1

pyspssio.SPSS_TWO_MISSVAL
2

pyspssio.SPSS_THREE_MISSVAL
3

pyspssio.SPSS_MISS_RANGE
-2

pyspssio.SPSS_MISS_RANGEANDVAL
-3

5.3 Formats

pyspssio.SPSS_FMT_A
1

pyspssio.SPSS_FMT_AHEX
2

pyspssio.SPSS_FMT_COMMA
3

pyspssio.SPSS_FMT_DOLLAR
4

pyspssio.SPSS_FMT_F
5

pyspssio.SPSS_FMT_IB
6

pyspssio.SPSS_FMT_PIBHEX
7

pyspssio.SPSS_FMT_P
8

pyspssio.SPSS_FMT_PIB
9

pyspssio.SPSS_FMT_PK
10

pyspssio.SPSS_FMT_RB
11

pyspssio.SPSS_FMT_RBHEX
12

pyspssio.SPSS_FMT_Z
15

pyspssio.SPSS_FMT_N
16

pyspssio.SPSS_FMT_E
17

pyspssio.SPSS_FMT_DATE
20

pyspssio.SPSS_FMT_TIME
21

pyspssio.SPSS_FMT_DATETIME
22

pyspssio.SPSS_FMT_ADATE
23

pyspssio.SPSS_FMT_JDATE
24

pyspssio.SPSS_FMT_DTIME
25

pyspssio.SPSS_FMT_WKDAY
26

pyspssio.SPSS_FMT_MONTH
27

pyspssio.SPSS_FMT_MOYR
28

pyspssio.SPSS_FMT_QYR
29

pyspssio.SPSS_FMT_WKYR
30

pyspssio.SPSS_FMT_PCT
31

pyspssio.SPSS_FMT_DOT
32

pyspssio.SPSS_FMT_CCA
33

pyspssio.SPSS_FMT_CCB
34

pyspssio.SPSS_FMT_CCC
35

pyspssio.SPSS_FMT_CCD
36

pyspssio.SPSS_FMT_CCE
37

pyspssio.SPSS_FMT_EDATE
38

pyspssio.SPSS_FMT_SDATE
39

pyspssio.SPSS_FMT_MTIME
85

pyspssio.SPSS_FMT_YMDHMS
86

5.4 Measure Levels

pyspssio.SPSS_MLVL_UNK
0

pyspssio.SPSS_MLVL_NOM
1

pyspssio.SPSS_MLVL_ORD
2

pyspssio.SPSS_MLVL_RAT
3

5.5 Alignments

pyspssio.SPSS_ALIGN_LEFT
0

pyspssio.SPSS_ALIGN_RIGHT
1

pyspssio.SPSS_ALIGN_CENTER
2

5.6 Roles

pyspssio.SPSS_ROLE_INPUT
0

pyspssio.SPSS_ROLE_TARGET
1

pyspssio.SPSS_ROLE_BOTH
2

pyspssio.SPSS_ROLE_NONE
3

pyspssio.SPSS_ROLE_PARTITION
4

pyspssio.SPSS_ROLE_SPLIT
5

pyspssio.SPSS_ROLE_FREQUENCY
6

pyspssio.SPSS_ROLE_RECORDID
7

CHAPTER

SIX

INDEX AND SEARCH

- genindex
- search

INDEX

Symbols

A

C

F

H

I

M

O

R

S

V

W