

---

**fsds<sub>1</sub>00719***Documentation*

***Release 0.7.1***

**James Irving**

**Jan 26, 2020**



---

## Contents

---

<b>1</b>	<b>Installation</b>	<b>1</b>
<b>2</b>	<b>Usage</b>	<b>3</b>
<b>3</b>	<b>fsds_100719.ds package</b>	<b>5</b>
<b>4</b>	<b>fsds_100719.ft package</b>	<b>9</b>
<b>5</b>	<b>fsds_100719.pt package</b>	<b>11</b>
<b>6</b>	<b>fsds_100719.jmi package</b>	<b>13</b>
<b>7</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>
	<b>Index</b>	<b>19</b>



# CHAPTER 1

---

## Installation

---

To install `fsds_100719`, run this command in your terminal:

```
# In terminal:  
$ pip install -U fsds_100719  
  
# In Jupyter Notebook / Learn.co  
$ !pip install -U fsds_100719
```

This is the preferred method to install `fsds_100719`, as it will always install the most recent stable release.

If you don't have `pip` installed, this [Python installation guide](#) can guide you through the process.



## CHAPTER 2

---

### Usage

---

To use fsds\_100719 in a project:

```
import fsds_100719 as fs
```

To import common modules as their usual handles e.g. pandas as pd, numpy as np, etc.:

```
from fsds_100719.imports import *
```

Functions worth importing by name:

```
# To easily inspect help and source code
from fsds_100719 import ihelp

#If you're import funcs from a local file.
from fsds_100719 import reload
```

You can load just your cohort or your own module as fs:

```
import fsds_100719.ft.jirvingphd as fs
# or
import fsds_100719.ft as fs
```





## CHAPTER 3

---

### fsds\_100719.ds package

---

A shared collection of tools for general use.

`fsds_100719.ds.add_dir_to_path(abs_path=None, rel_path=None, verbose=True)`

Adds the provided path (or current directory if None provided) to `sys.path`.

**Args:** `path` (str): folder to add to path (May need to be absolute). `rel_path` (str): relative folder path to be converted to absolute and added. `verbose` (bool): Controls display of success/failure messages. Default `=True`

`fsds_100719.ds.arr2series(array, series_index=None, series_name='array')`

Converts an array into a named series.

**Args:** `array` (numpy array): Array to transform. `series_index` (list, optional): List of values to be used as index.

Defaults to None, a numerical index.

`series_name` (str, optional): Name for series. Defaults to 'array'.

**Returns:** `converted_array`: Pandas Series with the name and index specified.

`fsds_100719.ds.capture_text(txt)`

Uses `StringIO` and `sys.stdout` to capture print statements.

**Args:** `txt` (str): pass string or command to display a string to capture

**Returns:** `txt_out` (str): captured print statement

`fsds_100719.ds.check_column(panda_obj, columns=None, nlargest='all')`

Prints column name, datatype, # and % of null values, and unique values for the `nlargest` # of rows (by **value-count\_**. it will only print results for those columns \*\*\*\*\* Params: `panda_object`: pandas DataFrame or Series `columns`: list containing names of columns (strings)

**Returns:** `None` prints values only

`fsds_100719.ds.check_df_for_columns(df, columns=None)`

Checks `df` for presence of columns.

`df`: `pd.DataFrame` to find columns in `columns`: str or list of str. column names

`fsds_100719.ds.check_null(df, columns=None, show_df=False)`  
Iterates through columns and checks for null values and displays # and % of column. Params: \*\*\*\*\*  
df: pandas DataFrame

columns: list of columns to check \*\*\*\*\*> Returns: displayed dataframe

`fsds_100719.ds.check_numeric(df, columns=None, unique_check=False, return_list=False, show_df=False)`  
Iterates through columns and checks for possible numeric features labeled as objects. Params: \*\*\*\*\*  
df: pandas DataFrame

**unique\_check: bool. (default=True)** If true, displays interactive interface for checking unique values in columns.

**return\_list: bool, (default=False)** If True, returns a list of column names with possible numeric types.

\*\*\*\*\*> Returns: dataframe displayed (always), list of column names if return\_list=True

`fsds_100719.ds.check_unique(df, columns=None)`  
Prints unique values for all columns in dataframe. If passed list of columns, it will only print results for those columns \*\*\*\*\* > Params: df: pandas DataFrame, or pd.Series columns: list containing names of columns (strings)

**Returns: None** prints values only

`fsds_100719.ds.column_report(df, index_col=None, sort_column='iloc', ascending=True, interactive=False, return_df=False)`  
Displays a DataFrame summary of each column's: - name, iloc, dtypes, null value count & %, # of 0's, min, max, med, mean, etc

**Args:** df (DataFrame): df to report index\_col (column to set as index, str): Defaults to None. sort\_column (str, optional): [description]. Defaults to 'iloc'. ascending (bool, optional): [description]. Defaults to True. as\_df (bool, optional): [description]. Defaults to False. interactive (bool, optional): [description]. Defaults to False. return\_df (bool, optional): [description]. Defaults to False.

**Returns:** column\_report (df): Non-styled version of displayed df report

`fsds_100719.ds.column_report_qgrid(df, index_col=None, sort_column='iloc', ascending=True, format_dict=None, as_df=False, as_interactive_df=False, show_and_return=True, as_qgrid=True, qgrid_options=None, qgrid_column_options=None, qgrid_col_defs=None, qgrid_callback=None)`

Returns a dataframe summary of the columns, their dtype, a summary dataframe with the column name, column dtypes, and a *decision\_map* dictionary of datatype. [!] Please note if qgrid does not display properly, enter this into your terminal and restart your terminal.

```
'jupyter nbextension enable -py -sys-prefix qgrid'# required for qgrid 'jupyter nbextension enable
-py -sys-prefix widgetsnbextension' # only required if you have not enabled the ipywidgets nbex-
tension yet
```

#### Default qgrid options:

```
default_grid_options={ # SlickGrid options 'fullWidthRows': True, 'syncColumnCellResize': True,
    'forceFitColumns': True, 'defaultColumnWidth': 50, 'rowHeight': 25, 'enableColumnReorder':
    True, 'enableTextSelectionOnCells': True, 'editable': True, 'autoEdit': False, 'explicitInitialization':
    True,

    # Qgrid options 'maxVisibleRows': 30, 'minVisibleRows': 8, 'sortable': True, 'filterable': True,
    'highlightSelectedCell': True, 'highlightSelectedRow': True

}
```

`fsds_100719.ds.compare_duplicates(df1, df2, to_drop=True, verbose=True, return_names_list=False)`

Compare two dfs for duplicate columns, drop if `to_drop=True`, useful to us before concatenating when dtypes are different between matching column names and `df.drop_duplicates` is not an option. Params: \_\_\_\_\_  
`df1, df2` : pandas dataframe suspected of having matching columns  
`to_drop` : bool, (default=True)

If True will give the option of dropping columns one at a time from either column.

**verbose: bool (default=True)** If True prints column names and types, set to false and `return_names_list=True` if only desire a list of column names and no interactive interface.

**return\_names\_list: bool (default=False)**, If True, will return a list of all duplicate column names.

Returns: List of column names if `return_names_list=True`, else nothing.

`fsds_100719.ds.display_side_by_side(*args)`

Display all input dataframes side by side. Also accept captioned styler df object (`df_in = df.style.set_caption('caption')`) Modified from Source: <https://stackoverflow.com/questions/38783027/jupyter-notebook-display-two-pandas-tables-side-by-side>

`fsds_100719.ds.find_outliers_zscore(col)`

Use scipy to calculate absolute Z-scores and return boolean series where True indicates it is an outlier Args:

`col` (Series): a series/column from your DataFrame

**Returns:** `idx_outliers` (Series): series of True/False for each row in `col`

Ex: `>> idx_outs = find_outliers(df['bedrooms']) >> df_clean = df.loc[idx_outs==False]`

`fsds_100719.ds.get_source_code_markdown(function)`

Retrieves the source code as a string and appends the markdown python syntax notation

`fsds_100719.ds.ihelp(function_or_mod, show_help=True, show_code=True, return_code=False, markdown=True, file_location=False)`

Call on any module or function to display the object's help command printout AND/OR source code displayed as Markdown using Python-syntax

`fsds_100719.ds.ihelp_menu(function_list, box_style='warning', to_embed=False)`

Creates a widget menu of the source code and help documentation of the functions in `function_list`.

**Args:** `function_list` (list): list of function object or string names of loaded function. `to_embed` (bool, optional): Returns interface (layout,output) if True. Defaults to False. `to_file` (bool, optional): Save . Defaults to False. `json_file` (str, optional): [description]. Defaults to 'ihelp\_output.txt'.

**Returns:** `full_layout` (ipywidgets GridBox): Layout of interface. `output()`

`fsds_100719.ds.inspect_df(df, n_rows=3, verbose=True)`

EDA: Show all pandas inspection tables. Displays `df.head()`, `df.info()`, `df.describe()`. By default also runs `check_null` and `check_numeric` to inspect columns for null values and to check string columns to detect numeric values. (If `verbose==True`) Parameters:

**df(dataframe):** dataframe to inspect

**n\_rows:** number of header rows to show (Default=3).

**verbose:** If `verbose==True` (default), `check_null` and `check_numeric`.

Ex: `inspect_df(df,n_rows=4)`

```
fsds_100719.ds.inspect_variables (local_vars=None,          sort_col='size',          ex-
                                include_funcs_mods=True,    top_n=10,    return_df=False,
                                always_display=True,         show_how_to_delete=False,
                                print_names=False)
```

Displays a dataframe of all variables and their size in memory, with the largest variables at the top.

**Args:** local\_vars (locals(): Must call locals() as first argument. sort\_col (str, optional): column to sort by. Defaults to 'size'. top\_n (int, optional): how many vars to show. Defaults to 10. return\_df (bool, optional): If True, return df instead of just showing df. Defaults to False. always\_display (bool, optional): Display df even if returned. Defaults to True. show\_how\_to\_delete (bool, optional): Prints out code to copy-paste into cell to del vars. Defaults to False. print\_names (bool, optional): [description]. Defaults to False.

**Raises:** Exception: if locals() not passed as first arg

Example Usage: # Must pass in local variables >> inspect\_variables(locals()) # To see command to delete list of vars" >> inspect\_variables(locals(),show\_how\_to\_delete=True)

```
fsds_100719.ds.is_var (name)
```

```
fsds_100719.ds.list2df (list, index_col=None, caption=None, return_df=True, df_kwds={})
```

Quick turn an appened list with a header (row[0]) into a pretty dataframe.

**Args** list (list of lists): index\_col (string): name of column to set as index; None (Default) has integer index. set\_caption (string): show\_and\_return (bool):

EXAMPLE USE: >> list\_results = [[“Test”,”N”,”p-val”]]

# ... run test and append list of result values ...

>> list\_results.append([test\_Name,length(data),p])

## Displays styled dataframe if caption: >> df = list2df(list\_results, index\_col=”Test”,  
set\_caption=”Stat Test for Significance”)

```
fsds_100719.ds.reload (mod)
```

**Reloads the module from file without restarting kernel.**

**Args:** mod (loaded mod or list of mod objects): name or handle of package (i.e.,[ pd, fs,np])

**Returns:** reload each model.

Example: # You pass in whatever name you imported as. import my\_functions\_from\_file as mf # after editing the source file: # mf.reload(mf)

```
fsds_100719.ds.save_ihelp_to_file (function, save_help=False, save_code=True, as_md=False,
                                as_txt=True,    folder='readme_resources/ihelp_outputs/',
                                filename=None, file_mode='w')
```

Saves the string representation of the ihelp source code as markdown. Filename should NOT have an extension. .txt or .md will be added based on as\_md/as\_txt. If filename is None, function name is used.

```
fsds_100719.ds.show_del_me_code (called_by_inspect_vars=False)
```

Prints code to copy and paste into a cell to delete vars using a list of their names. Companion function inspect\_variables(locals(),print\_names=True) will provide var names to copy/paste

```
fsds_100719.ds.show_off_vs_code ()
```

## CHAPTER 4

---

### fsds\_100719.ft package

---

A collection of submodules by [online-ds-ft-100719](#). Maintained by James Irving (GitHub: [jirvingphd](#))  
[james.irving@flatironschool.com](mailto:james.irving@flatironschool.com)



## CHAPTER 5

---

fsds\_100719.pt package

---

A collection of submodules by online-ds-pt-100719. Maintained by James Irving (GitHub: [jirvingphd](#))  
[james.irving@flatironschool.com](mailto:james.irving@flatironschool.com)

fsds\_100719.pt.**placeholder2**()





## CHAPTER 6

---

fsds\_100719.jmi package

---



## CHAPTER 7

---

### Indices and tables

---

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



### f

`fsds_100719.ds`, [5](#)  
`fsds_100719.ft`, [9](#)  
`fsds_100719.jmi`, [13](#)  
`fsds_100719.pt`, [11](#)



## A

`add_dir_to_path()` (in module *fsds\_100719.ds*), 5  
`arr2series()` (in module *fsds\_100719.ds*), 5

## C

`capture_text()` (in module *fsds\_100719.ds*), 5  
`check_column()` (in module *fsds\_100719.ds*), 5  
`check_df_for_columns()` (in module *fsds\_100719.ds*), 5  
`check_null()` (in module *fsds\_100719.ds*), 5  
`check_numeric()` (in module *fsds\_100719.ds*), 6  
`check_unique()` (in module *fsds\_100719.ds*), 6  
`column_report()` (in module *fsds\_100719.ds*), 6  
`column_report_qgrid()` (in module *fsds\_100719.ds*), 6  
`compare_duplicates()` (in module *fsds\_100719.ds*), 6

## D

`display_side_by_side()` (in module *fsds\_100719.ds*), 7

## F

`find_outliers_zscore()` (in module *fsds\_100719.ds*), 7  
*fsds\_100719.ds* (module), 5  
*fsds\_100719.ft* (module), 9  
*fsds\_100719.jmi* (module), 13  
*fsds\_100719.pt* (module), 11

## G

`get_source_code_markdown()` (in module *fsds\_100719.ds*), 7

## I

`ihelp()` (in module *fsds\_100719.ds*), 7  
`ihelp_menu()` (in module *fsds\_100719.ds*), 7  
`inspect_df()` (in module *fsds\_100719.ds*), 7

`inspect_variables()` (in module *fsds\_100719.ds*), 7  
`is_var()` (in module *fsds\_100719.ds*), 8

## L

`list2df()` (in module *fsds\_100719.ds*), 8

## P

`placeholder2()` (in module *fsds\_100719.pt*), 11

## R

`reload()` (in module *fsds\_100719.ds*), 8

## S

`save_ihelp_to_file()` (in module *fsds\_100719.ds*), 8  
`show_del_me_code()` (in module *fsds\_100719.ds*), 8  
`show_off_vs_code()` (in module *fsds\_100719.ds*), 8