Jason Schlottman

HWRS 401

10/21/2021

Cheat Sheet 4 -Pandas\_Dataframes

* **Define:** Pandas:

A pandas\_dataframe (aka pandas) refers to a two-dimensional labeled data structure consisting of columns which may contain varying types (heterogeneous types) of tabular formatted data. They are extremely useful for accessing, editing, and modifying large amounts of data. Common uses include use for time series analysis, involvement in machine learning applications, etc.

\*Note= Since these dataframes are often very large, try using the head() and tail() functions to view certain sections since all the rows might not be practical or visible at once.

* Pandas dataframes are generally created using two common methods. They are either composed from scratch or accessed by reading through a 'comma-separated value' file (.csv).

To create a pandas from scratch:

In a Python script write a function composed of a variable for both the columns and rows after importing Pandas. The first variable may commonly be the function data, and the second variable may define labels, for example. The data can be passed as a list, tuple, NumPy Array, or dictionary. Other less common types exist but these are the primary important choices for our needs.

To access a pandas by reading a .csv file:

Recall there is the read\_csv() function and we can call using pandas.read\_csv() or pd.read\_csv() after we have ensured that pandas has been imported. Useful instructions related to access through reading a .csv file can be found in course materials for Week9 or at the following URL:

https://pythonbasics.org/read-csv-with-pandas/

**Returns:** vary depending on what data type was utilized, but expect to access dictionaries and form new variables many times as there may be several layers of dictionaries in the data.

**Loc, iloc, slicing:**

Very useful site for help with operations/property function: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.loc.html

Slice with labels for row, single label for the column

loc example: df.loc['a':'z', 'q']

Recall that when using the .loc property, if we call using a number this refers to the label of the index rather than an integer position of the index.

Iloc:

**Index of pandas dataframes**:

Pandas supports 3 major types of 'multi-axis' indexing which are used with '.loc' '.iloc. and simple [] formatting. See above sections for major elements relating to .loc and .iloc functions. The primary function of indexing with [] (aka \_\_get\_item\_) for implementing class behavior is for selecting out lower dimensional slices.

For example:

if we index using [] for a series oject type, such as series[label] the return type will result in a scalar.

If we index with [] for a Dataframe like frame[colname], the return type will be a series which reflects the colname variable.

**Key methods associated with pandas:**

Rather than attempt to write out just a few key methods here's a list of some of the most common key methods used for pandas dataframes.

Image sourced from 'pandas.pydata.org'

See attached URL below for a more complete list!

https://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.DataFrame.html

Table

Description automatically generated

**Key attributes associated with pandas**

While there are even more key attributes and many which are very specific to certain instances, here's a useful list with some commonly seen attributes rather than writing out a few on my own.

Credit= image sourced from 'pandas.pydata.org'

Again follow hyperlink to view useful and more complete list of key attributes!

https://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.DataFrame.html

Graphical user interface, application

Description automatically generated

**Extra helpful functions not mentioned in above sections:**

There are numerous functions related to pandas dataframes used for many different returns and specific designs when using pandas to work with or structure large datasets.

Some of the most helpful and common functions are grouped into categories describing their uses such as functions related to manipulating data, detecting missing data, converting data, fnctions related to datetime, hashing, testing, etc. The list goes on.

An example of detecting missing data = isnull(obj) which detects missing values for an array-like object

An example of a function for manipulating data may be concat(), merge() etc.

Examples of datetime functions may be date\_range() which returns a fixed frequency dattime\_index

And many, many more.

See the following helpful resources more more examples of common pandas related functions and their returns and uses.

<https://pandas.pydata.org/docs/reference/general_functions.html>

<https://www.datacamp.com/community/tutorials/pandas-tutorial-dataframe-python>