Kaggle Data Science Cheat Sheet

by blueberryshortcake via cheatography.com/162779/cs/43739/

Creating, Reading, Writing

df = pd.DataFrame({"col0": [val0, val1], "col-1": [val0, val1]}, index=[0, 1])	Create a dataframe
series = pd.Series(["val0", "val1", "val2"], index=[0, 1, 2, 3], name="name")	Create a series
read = pd.read_csv("../folder/folder/file.csv", index_col=0)	Read a csv
save.to_csv("file.csv")	Save an existing dataframe as a csv

Indexing, Selecting, Assigning

table.head	Show first 5 rows of a dataframe
table["col"]	Select the col from table
table.col.iloc[0]	Select 1st value of a col from table
table.iloc[0]	Select 1st row of data from table
table.col.iloc[:10]	Select 1st 10 values from col in table (index-based select)
table.col.loc[:10]	Select 1st 10 values from col in table (label-based select)
table.loc[indices, cols]	Select certain rows from certain cols
table[table.col == 'val']	Select cols have a certain val (conditional select)
table.col.isin(['val1,' 'val2'])	Select cols have certain vals (conditional select)

Summary Functions & Maps

table.col.describe()	Get high-lvl summary of given col's attributes
table.col.mean()	Get mean of a col with numerical vals
table.col.unique()	Get each unique val of a col w/ no dupes
table.col.value_co-unts()	Get frequency of each val in col

Summary Functions & Maps (cont)

table.col.map(l-ambda p: p - s)	Map function to remap a Series of point vals (p) by using a transformation (p-s) -> returns new Series
table.apply(-func, axis='col-umns')	Apply function to transform entire df by calling custom method (func taking a row) on each row

Grouping & Sorting

table.groupby('col').c-ol.count()	Group data w/ same vals in the given col -> count frequency of given col (same as value_counts())
table.groupby('col').s-ize()	Same as above
table.groupby('col').a-pply(lambda df: df.tit-le.iloc[0])	Select name (title) of the 1st thing in col
table.col.idxmax()	Get index of max val in col
table.groupby(['co-l0']).col1.agg([f1, f2, f3])	agg() runs diff. funcs. simultaneously on a df
table.groupby(['col0', 'col1']).col2.agg([len])	Multi-index output has tiered structure. Require 2 levels of labels to retrieve a val
df.reset_index()	Muti-index method used to converting back to regular index
df.sort_values(-by='col')	Sort rows of data by vals in col (ascending)
df.sort_values(-by='col', ascending-=False)	Sort rows of data by vals in col (desce-nding)
df.sort_values(by=['c-ol0', 'col1'])	Sort rows by more than 1 col at a time
df.sort_index()	Sort rows by index (default order; ascending)

Data Types & Missing Values	
table.col.dtype	Get data type of a col
table.dtypes	Get data types of each col in table
table.col.astype('datatype')	Convert col to datatype if allowed (e,g, int64 -> float64)
table.index.dtype	Number indices are int64
table[pd.isnull(table.col)]	Select NaN entries in a col
table.col.fillna("filler")	Replace all NaN vals in a col with a sentinel val ("Unknown", "Undisclosed", "Invalid") or non-null val
table.col.replace("init_val", "new_val")	Replace, in col, all existing vals with new_vals

Renaming & Combining	
table.rename(columns={'init': 'new'})	Rename col or index col names
table.rename(index={0: 'firstEntry', 1: 'secondEntry})	Rename index or col vals by specifying an index or col param
table.rename_axis("name", axis='rows').rename_axis("name1", axis='columns')	Rename row index &/or col index
pd.concat(list, of, els)	Smush together the list of elements along an axis
left.join(right, lsuffix='strL', rsuffix='strR')	Combine diff df objects that have an index in common. left and right are df.s defined beforehand