## PANDAS

## Key and Imports

In this cheat sheet, we use the following shorthand:

df | Any pandas DataFrame object  
s | Any pandas Series object

You'll also need to perform the following imports to get started:

import pandas as pd

import numpy as np

## Importing Data

pd.read\_csv(filename) | From a CSV file  
pd.read\_table(filename) | From a delimited text file (like TSV)  
pd.read\_excel(filename) | From an Excel file  
pd.read\_sql(query, connection\_object) | Read from a SQL table/database  
pd.read\_json(json\_string) | Read from a JSON formatted string, URL or file.  
pd.read\_html(url) | Parses an html URL, string or file and extracts tables to a list of dataframes  
pd.read\_clipboard() | Takes the contents of your clipboard and passes it to read\_table()  
pd.DataFrame(dict) | From a dict, keys for columns names, values for data as lists

## Exporting Data

df.to\_csv(filename) | Write to a CSV file  
df.to\_excel(filename) | Write to an Excel file  
df.to\_sql(table\_name, connection\_object) | Write to a SQL table  
df.to\_json(filename) | Write to a file in JSON format

## Create Test Objects

Useful for testing code segements

pd.DataFrame(np.random.rand(20,5)) | 5 columns and 20 rows of random floats  
pd.Series(my\_list) | Create a series from an iterable my\_list  
df.index = pd.date\_range('1900/1/30', periods=df.shape[0]) | Add a date index

## Viewing/Inspecting Data

df.head(n) | First n rows of the DataFrame  
df.tail(n) | Last n rows of the DataFrame  
df.shape() | Number of rows and columns  
df.info() | Index, Datatype and Memory information  
df.describe() | Summary statistics for numerical columns  
s.value\_counts(dropna=False) | View unique values and counts  
df.apply(pd.Series.value\_counts) | Unique values and counts for all columns

## Selection

df[col] | Returns column with label col as Series  
df[[col1, col2]] | Returns columns as a new DataFrame  
s.iloc[0] | Selection by position  
s.loc['index\_one'] | Selection by index  
df.iloc[0,:] | First row  
df.iloc[0,0] | First element of first column

## Data Cleaning

df.columns = ['a','b','c'] | Rename columns  
pd.isnull() | Checks for null Values, Returns Boolean Arrray  
pd.notnull() | Opposite of pd.isnull()  
df.dropna() | Drop all rows that contain null values  
df.dropna(axis=1) | Drop all columns that contain null values  
df.dropna(axis=1,thresh=n) | Drop all rows have have less than n non null values  
df.fillna(x) | Replace all null values with x  
s.fillna(s.mean()) | Replace all null values with the mean (mean can be replaced with almost any function from the statistics section)  
s.astype(float) | Convert the datatype of the series to float  
s.replace(1,'one') | Replace all values equal to 1 with 'one'  
s.replace([1,3],['one','three']) | Replace all 1 with 'one' and 3 with 'three'  
df.rename(columns=lambda x: x + 1) | Mass renaming of columns  
df.rename(columns={'old\_name': 'new\_ name'}) | Selective renaming  
df.set\_index('column\_one') | Change the index  
df.rename(index=lambda x: x + 1) | Mass renaming of index

## Filter, Sort, and Groupby

df[df[col] > 0.5] | Rows where the column col is greater than 0.5  
df[(df[col] > 0.5) & (df[col] < 0.7)] | Rows where 0.7 > col > 0.5  
df.sort\_values(col1) | Sort values by col1 in ascending order  
df.sort\_values(col2,ascending=False) | Sort values by col2 in descending order  
df.sort\_values([col1,col2],ascending=[True,False]) | Sort values by col1 in ascending order then col2 in descending order  
df.groupby(col) | Returns a groupby object for values from one column  
df.groupby([col1,col2]) | Returns groupby object for values from multiple columns  
df.groupby(col1)[col2] | Returns the mean of the values in col2, grouped by the values in col1 (mean can be replaced with almost any function from the statistics section)  
df.pivot\_table(index=col1,values=[col2,col3],aggfunc=mean) | Create a pivot table that groups by col1 and calculates the mean of col2 and col3  
df.groupby(col1).agg(np.mean) | Find the average across all columns for every unique col1 group  
df.apply(np.mean) | Apply the function np.mean() across each column  
nf.apply(np.max,axis=1) | Apply the function np.max() across each row

## Join/Combine

df1.append(df2) | Add the rows in df1 to the end of df2 (columns should be identical)  
pd.concat([df1, df2],axis=1) | Add the columns in df1 to the end of df2 (rows should be identical)  
df1.join(df2,on=col1,how='inner') | SQL-style join the columns in df1 with the columns on df2 where the rows for col have identical values. how can be one of 'left', 'right', 'outer', 'inner'

## Statistics

These can all be applied to a series as well.

df.describe() | Summary statistics for numerical columns  
df.mean() | Returns the mean of all columns  
df.corr() | Returns the correlation between columns in a DataFrame  
df.count() | Returns the number of non-null values in each DataFrame column  
df.max() | Returns the highest value in each column  
df.min() | Returns the lowest value in each column  
df.median() | Returns the median of each column  
df.std() | Returns the standard deviation of each column

## NUMPY

## Key and Imports

In this cheat sheet, we use the following shorthand:

arr | A NumPy Array object

You'll also need to import numpy to get started:

import numpy as np

## Importing/exporting

np.loadtxt('file.txt') | From a text file  
np.genfromtxt('file.csv',delimiter=',') | From a CSV file  
np.savetxt('file.txt',arr,delimiter=' ') | Writes to a text file  
np.savetxt('file.csv',arr,delimiter=',') | Writes to a CSV file

## Creating Arrays

np.array([1,2,3]) | One dimensional array  
np.array([(1,2,3),(4,5,6)]) | Two dimensional array  
np.zeros(3) | 1D array of length 3 all values 0  
np.ones((3,4)) | 3x4 array with all values 1  
np.eye(5) | 5x5 array of 0 with 1 on diagonal (Identity matrix)  
np.linspace(0,100,6) | Array of 6 evenly divided values from 0 to 100  
np.arange(0,10,3) | Array of values from 0 to less than 10 with step 3 (eg [0,3,6,9])  
np.full((2,3),8) | 2x3 array with all values 8  
np.random.rand(4,5) | 4x5 array of random floats between 0-1  
np.random.rand(6,7)\*100 | 6x7 array of random floats between 0-100  
np.random.randint(5,size=(2,3)) | 2x3 array with random ints between 0-4

## Inspecting Properties

arr.size | Returns number of elements in arr  
arr.shape | Returns dimensions of arr (rows,columns)  
arr.dtype | Returns type of elements in arr  
arr.astype(dtype) | Convert arr elements to type dtype  
arr.tolist() | Convert arr to a Python list  
np.info(np.eye) | View documentation for np.eye

## Copying/sorting/reshaping

np.copy(arr) | Copies arr to new memory  
arr.view(dtype) | Creates view of arr elements with type dtype  
arr.sort() | Sorts arr  
arr.sort(axis=0) | Sorts specific axis of arr  
two\_d\_arr.flatten() | Flattens 2D array two\_d\_arr to 1D  
arr.T | Transposes arr (rows become columns and vice versa)  
arr.reshape(3,4) | Reshapes arr to 3 rows, 4 columns without changing data  
arr.resize((5,6)) | Changes arr shape to 5x6 and fills new values with 0

## Adding/removing Elements

np.append(arr,values) | Appends values to end of arr  
np.insert(arr,2,values) | Inserts values into arr before index 2  
np.delete(arr,3,axis=0) | Deletes row on index 3 of arr  
np.delete(arr,4,axis=1) | Deletes column on index 4 of arr

## Combining/splitting

np.concatenate((arr1,arr2),axis=0) | Adds arr2 as rows to the end of arr1  
np.concatenate((arr1,arr2),axis=1) | Adds arr2 as columns to end of arr1  
np.split(arr,3) | Splits arr into 3 sub-arrays  
np.hsplit(arr,5) | Splits arr horizontally on the 5th index

## Indexing/slicing/subsetting

arr[5] | Returns the element at index 5  
arr[2,5] | Returns the 2D array element on index [2][5]  
arr[1]=4 | Assigns array element on index 1 the value 4  
arr[1,3]=10 | Assigns array element on index [1][3] the value 10  
arr[0:3] | Returns the elements at indices 0,1,2 (On a 2D array: returns rows 0,1,2)  
arr[0:3,4] | Returns the elements on rows 0,1,2 at column 4  
arr[:2] | Returns the elements at indices 0,1 (On a 2D array: returns rows 0,1)  
arr[:,1] | Returns the elements at index 1 on all rows  
arr<5 | Returns an array with boolean values  
(arr1<3) & (arr2>5) | Returns an array with boolean values  
~arr | Inverts a boolean array  
arr[arr<5] | Returns array elements smaller than 5

## Scalar Math

np.add(arr,1) | Add 1 to each array element  
np.subtract(arr,2) | Subtract 2 from each array element  
np.multiply(arr,3) | Multiply each array element by 3  
np.divide(arr,4) | Divide each array element by 4 (returns np.nan for division by zero)  
np.power(arr,5) | Raise each array element to the 5th power

## Vector Math

np.add(arr1,arr2) | Elementwise add arr2 to arr1  
np.subtract(arr1,arr2) | Elementwise subtract arr2 from arr1  
np.multiply(arr1,arr2) | Elementwise multiply arr1 by arr2  
np.divide(arr1,arr2) | Elementwise divide arr1 by arr2  
np.power(arr1,arr2) | Elementwise raise arr1 raised to the power of arr2  
np.array\_equal(arr1,arr2) | Returns True if the arrays have the same elements and shape  
np.sqrt(arr) | Square root of each element in the array  
np.sin(arr) | Sine of each element in the array  
np.log(arr) | Natural log of each element in the array  
np.abs(arr) | Absolute value of each element in the array  
np.ceil(arr) | Rounds up to the nearest int  
np.floor(arr) | Rounds down to the nearest int  
np.round(arr) | Rounds to the nearest int

## Statistics

np.mean(arr,axis=0) | Returns mean along specific axis  
arr.sum() | Returns sum of arr  
arr.min() | Returns minimum value of arr  
arr.max(axis=0) | Returns maximum value of specific axis  
np.var(arr) | Returns the variance of array  
np.std(arr,axis=1) | Returns the standard deviation of specific axis  
arr.corrcoef() | Returns correlation coefficient of array