pandas.DataFrame.dropna

`DataFrame.`**`dropna`**(*axis=0*, *how='any'*, *thresh=None*, *subset=None*, *inplace=False*) [source]

Return object with labels on given axis omitted where alternately any or all of the data are missing

Parameters:

axis : *{0 or 'index', 1 or 'columns'}, or tuple/list thereof*

Pass tuple or list to drop on multiple axes

how : *{'any', 'all'}*

- any : if any NA values are present, drop that label
- all : if all values are NA, drop that label

thresh : *int, default None*

int value : require that many non-NA values

subset : *array-like*

Labels along other axis to consider, e.g. if you are dropping rows these would be a list of columns to include

inplace : *boolean, default False*

If True, do operation inplace and return None.

Returns:

dropped : *DataFrame*

Examples

```
>>> df = pd.DataFrame([[np.nan, 2, np.nan, 0], [3, 4, np.nan, 1],
...                    [np.nan, np.nan, np.nan, 5]],
...                   columns=list('ABCD'))
>>> df
     A    B   C  D
0  NaN  2.0 NaN  0
1  3.0  4.0 NaN  1
2  NaN  NaN NaN  5
```

Drop the columns where all elements are nan:

```
>>> df.dropna(axis=1, how='all')
     A    B  D
0  NaN  2.0  0
1  3.0  4.0  1
2  NaN  NaN  5
```

Drop the columns where any of the elements is nan

```
>>> df.dropna(axis=1, how='any')
   D
0  0
```

```
1  1
2  5
```

Drop the rows where all of the elements are nan (there is no row to drop, so df stays the same):

```
>>> df.dropna(axis=0, how='all')
     A    B   C  D
0  NaN  2.0 NaN  0
1  3.0  4.0 NaN  1
2  NaN  NaN NaN  5
```

Keep only the rows with at least 2 non-na values:

```
>>> df.dropna(thresh=2)
     A    B   C  D
0  NaN  2.0 NaN  0
1  3.0  4.0 NaN  1
```