datacamp

Python For Data Science

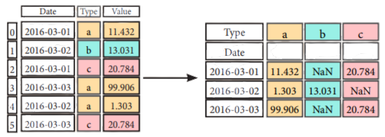
Data Wrangling in Pandas Cheat Sheet

Learn Data Wrangling online at www.DataCamp.com

> Reshaping Data

Pivot

```
>>> df3= df2.pivot(index='Date', #Spread rows into columns
                   columns='Type',
                   values='Value')
```

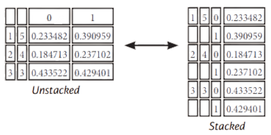


Pivot Table

```
>>> df4 = pd.pivot_table(df2, #Spread rows into
                         columns values='Value',
                         index='Date',
                         columns='Type'])
```

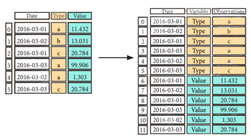
Stack / Unstack

```
>>> stacked = df5.stack() #Pivot a level of column labels
>>> stacked.unstack() #Pivot a level of index labels
```



Melt

```
>>> pd.melt(df2, #Gather columns into rows
            id_vars=["Date"],
            value_vars=["Type", "Value"],
            value_name="Observations")
```



> Iteration

```
>>> df.iteritems() #(Column-index, Series) pairs
>>> df.iterrows() #(Row-index, Series) pairs
```

> Missing Data

```
>>> df.dropna() #Drop NaN values
>>> df3.fillna(df3.mean()) #Fill NaN values with a predetermined value
>>> df2.replace("a", "f") #Replace values with others
```

> Advanced Indexing

Also see NumPy Arrays

Selecting

```
>>> df3.loc[:,(df3>1).any()] #Select cols with any vals >1
>>> df3.loc[:,(df3>1).all()] #Select cols with vals > 1
>>> df3.loc[:,df3.isnull().any()] #Select cols with NaN
>>> df3.loc[:,df3.notnull().all()] #Select cols without NaN
```

Indexing With `isin()`

```
>>> df[(df.Country.isin(df2.Type))] #Find same elements
>>> df3.filter(items="a","b"]) #Filter on values
>>> df.select(lambda x: not x%5) #Select specific elements
```

Where

```
>>> s.where(s > 0) #Subset the data
```

Query

```
>>> df6.query('second > first') #Query DataFrame
```

Setting/Resetting Index

```
>>> df.set_index('Country') #Set the index
>>> df4 = df.reset_index() #Reset the index
>>> df = df.rename(index=str, #Rename
                   DataFrame columns={"Country":"cntry",
                                      "Capital":"cptl",
                                      "Population":"ppltn"})
```

Reindexing

```
>>> s2 = s.reindex(['a','c','d','e','b'])
```

Forward Filling

```
>>> df.reindex(range(4),
               method='ffill')
```

```
   Country    Capital      Population
0 Belgium    Brussels     11190846
1 India      New Delhi    1303171035
2 Brazil     Brasília     207847528
3 Brazil     Brasília     207847528
```

Backward Filling

```
>>> s3 = s.reindex(range(5),
                   method='bfill')
```

```
0    3
1    3
2    3
3    3
4    3
```

MultiIndexing

```
>>> arrays = [np.array([1,2,3]),
              np.array([5,4,3])]
>>> df5 = pd.DataFrame(np.random.rand(3, 2), index=arrays)
>>> tuples = list(zip(*arrays))
>>> index = pd.MultiIndex.from_tuples(tuples,
                                      names=['first', 'second'])
>>> df6 = pd.DataFrame(np.random.rand(3, 2), index=index)
>>> df2.set_index(["Date", "Type"])
```

> Duplicate Data

```
>>> s3.unique() #Return unique values
>>> df2.duplicated('Type') #Check duplicates
>>> df2.drop_duplicates('Type', keep='last') #Drop duplicates
>>> df.index.duplicated() #Check index duplicates
```

> Grouping Data

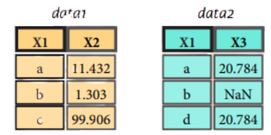
Aggregation

```
>>> df2.groupby(by=['Date','Type']).mean()
>>> df4.groupby(level=0).sum()
>>> df4.groupby(level=0).agg({'a':lambda x:sum(x)/len(x), 'b': np.sum})
```

Transformation

```
>>> customSum = lambda x: (x+x%2)
>>> df4.groupby(level=0).transform(customSum)
```

> Combining Data



Merge

```
>>> pd.merge(data1,
             data2,
             how='left',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN
c	99.906	NaN

```
>>> pd.merge(data1,
             data2,
             how='right',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN
d	NaN	20.784

```
>>> pd.merge(data1,
             data2,
             how='inner',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN

```
>>> pd.merge(data1,
             data2,
             how='outer',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN
c	99.906	NaN
d	NaN	20.784

Join

```
>>> data1.join(data2, how='right')
```

Concatenate

Vertical

```
>>> s.append(s2)
```

Horizontal/Vertical

```
>>> pd.concat([s,s2],axis=1, keys=['One','Two'])
>>> pd.concat([data1, data2], axis=1, join='inner')
```

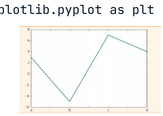
> Dates

```
>>> df2['Date']= pd.to_datetime(df2['Date'])
>>> df2['Date']= pd.date_range('2000-1-1',
                               periods=6,
                               freq='M')
>>> dates = [datetime(2012,5,1), datetime(2012,5,2)]
>>> index = pd.DatetimeIndex(dates)
>>> index = pd.date_range(datetime(2012,2,1), end, freq='BM')
```

> Visualization

Also see Matplotlib

```
>>> import matplotlib.pyplot as plt
>>> s.plot()
>>> plt.show()
```



```
>>> df2.plot()
>>> plt.show()
```