Pandas cheat sheet

Pandas Provides data analysis tools for Python. All of the following code examples refer to the data-frame below.

df =

	col1	col2	← axis1
A	1	4	
B	2	5	← axis0
C	3	6	

Import Pandas:

```
import Pandas as Pd
```

create a series:

```
S = Pd.Series([1,2,3],
              index=['A','B','C'],
              name='col1')
```

Create a dataframe:

```
data = [[1,4],[2,5],[3,6]]
index = ['A','B','C']
df = Pd.DataFrame(data, index=index,
                  Columns=['col1','col2'])
```

Load a dataframe:

```
df = Pd.read_csv('filename.csv', sep=',',
                 names=['col1','col2'],
                 index_col=0,
                 encoding='utf-8',
                 nrows=3)
```

Scanned with OKEN Scanner

Selecting rows and columns

Select single column:

```
df['col1']
```

Select multiple column:

```
df[['col1', 'col2']]
```

Show first n rows:

```
df.head(2)
```

Show last n rows:

```
df.tail(2)
```

Select rows by index values:

```
df.loc['A']  df.loc[['A','B']]
```

Select rows by Position:

```
df.loc[1]  df.loc[1:]
```

Data wrangling

Filter by value:

```
df[df['col1'] > 1]
```

Sort by columns:

```
df.sort_values(['col2', 'col2'],
               ascending=[False, True])
```

Identify duplicate rows:

```
df.duplicated()
```

Identify unique rows:

```
df['col1'].unique()
```

Swap rows and columns:

```
df = df.transpose()
df = df.T
```

Drop a column:

```
df = df.drop('col1', axis=1)
```

Clone a data frame:

```
clone = df.copy()
```

Scanned with OKEN Scanner

Connect multiple data frames vertically:

```
df2 = df + 5 # new dataframe
Pd.Concat([df,df2])
```

Merge multiple data frames horizontally:

```
df3 = Pd.DataFrame([[1,7],[8,9]],
                   index=['B','D'],
                   Columns=['Col1','Col2'])
#df3: new dataframe
```

Only merge complete rows (INNER JOIN):

```
df.merge(df3)
```

Left column stays complete (LEFT OUTER JOIN):

```
df.merge(df3, how='left')
```

Right column stays complete (RIGHT OUTER JOIN):

```
df.merge(df3, how='right')
```

Preserve all values (OUTER JOIN):

```
df.merge(df3, how='outer')
```

Merge rows by index:

```
df.merge(df3, left_index=True,
         right_index=True)
```

Fill NaN values:

```
df.fillna(0)
```

Apply your own function:

```
df func(x):
    return 2**x
df.apply(func)
```

Arithmetics and statistics

Add to all values:

```
df+10
```

Sum over columns:

```
df.sum()
```

Cumulative sum over columns:

```
df.cumsum()
```

Scanned with OKEN Scanner

Mean over columns:

```
df.mean()
```

Standard deviation over columns:

```
df.std()
```

Count unique values:

```
df['col1'].value_counts()
```

Summarize descriptive statistics:

```
df.describe()
```

Hierarchical indexing

Create hierarchical index:

```
df.stack()
```

Dissolve hierarchical index:

```
df.unstack()
```

Aggregation

Create group object:

```
g=df.groupby('col1')
```

Iterate over groups:

```
for i, group in g:
    print(i, group)
```

Aggregate groups:

```
g.sum()
g.prod()
g.mean()
g.std()
g.describe()
```

Select columns from groups:

```
g['col2'].sum()
g[['col2', 'col3']].sum()
```

Scanned with OKEN Scanner

Transform values:

```
import math
g.transform(math.log)
```

Apply a list function on each group:

```
df strsum(group):
  return ''.join([str(x) for x in group.value])
g['col2'].apply(strsum)
```

Data export

Data as NumPy array:

```
df.values
```

Save data as csv file:

```
df.to_csv('output.csv', sep=",")
```

Format a dataframe as tabular string:

```
df.to_string()
```

Convent a dataframe to a dictionary:

```
df.to_dict()
```

Save a dataframe as an Excel table:

```
df.to_excel('output.xlsx')
```

Visualization

Import matplotlib:

```
import matplotlib.pyplot as plt
```

Start a new diagram:

```
Plt.figure()
```

Scatter plot:

```
df.plot.scatter('col1','col2',
                style='ro')
```

Bar plot:

```
df.plot.bar(x='col1', y='col2',
            width=0.7)
```

Scanned with OKEN Scanner

Area plot:

```
df.plot.area(stacked = True,
             alpha = 1.0)
```

Box-and-whisker plot:

```
df.plot.box()
```

Histogram over one column:

```
df['COL1'].plot.hist(bins=3)
```

Histogram over all column:

```
df.plot.hist(bins = 3, alpha = 0.5)
```

Set tick marks:

```
labels = ['A', 'B', 'C', 'D']
positions = [1, 2, 3, 4]
plt.xticks(positions, labels)
plt.yticks(positions, labels)
```

Select area to plot:

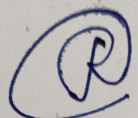
```
plt.axis([0, 2.5, 0, 10]) # [from x, to x,
                            from y, to y]
```

Label diagram and axes:

```
plt.title('correlation')
plt.xlabel('Nunstuck')
plt.ylabel('Slotermeyer')
```

Save most recent diagram:

```
plt.Savefig("plot.png')
plt.Savefig('plot.png', dpi = 300)
plt.Savefig('plot.svg')
```



Scanned with OKEN Scanner