Select multiple columns:
```python
df[['col1', 'col2']]
```

Show first n rows:
```python
df.head(2)
```

Show last n rows:
```python
df.tail(2)
```

Select rows by index values:
```python
df.ix['A']
df.ix[['A', 'B']]
```

Select rows by position:
```python
df.ix[1]
df.ix[1:]
```

Data wrangling

Filter by value:
```python
df[df['col1'] > 1]
```

Sort by columns:
```python
df.sort(['col2', 'col2'], ascending=[False, True])
```

Identify duplicate rows:
```python
df.duplicated()
```

Identify unique rows:
```python
df['col1'].unique()
```

Swap rows and columns:
```python
df = df.transpose()
```

Remove a column:
```python
del df['col2']
```

Clone a data frame:
```python
clone = df.copy()
```

Connect multiple data frames vertically:
```python
df2 = df + 10
pd.concat([df, df2])
```

Merge multiple data frames horizontally:
```python
df3 = pd.DataFrame([[1, 7], [8, 9]],
                   index=['B', 'D'],
                   columns=['col1', 'col3'])
```

Only merge complete rows (INNER JOIN):
```python
df.merge(df3)
```

Left column stays complete (LEFT OUTER JOIN):
```python
df.merge(df3, how='left')
```

Right column stays complete (RIGHT OUTER JOIN):
```python
df.merge(df3, how='right')
```

Preserve all values (OUTER JOIN):
```python
df.merge(df3, how='outer')
```

Merge rows by index:
```python
df.merge(df3, left_index=True, right_index=True)
```

Fill NaN values:
```python
df.fillna(0.0)
```

Apply your own function:
```python
def func(x): return 2**x
df.apply(func)
```

Arithmetics and statistics

Add to all values:
```python
df + 10
```

Sum over columns:
```python
df.sum()
```

Cumulative sum over columns:
```python
df.cumsum()
```

Mean over columns:
```python
df.mean()
```

Standard devieation over columns:
```python
df.std()
```

Count all values that occurr:
```python
df['col1'].value_counts()
```

Summarize descriptive statistics:
```python
df.describe()
```

Hierarchical indexing

Create hierarchical index:
```python
df.stack()
```

Dissolve hierarchical index:
```python
df.unstack()
```

Aggregation

Create group object:
```python
g = df.groupby('col1')
```

Iterate over groups:
```python
for i, group in g:
    print(i, group)
```

Aggregate groups:
```python
g.sum()
g.prod()
g.mean()
g.std()
g.describe()
```

Select columns from groups:
```python
g['col2'].sum()
g[['col2', 'col3']].sum()
```

Transform values:
```python
import math
g.transform(math.log)
```

Apply a list function on each group:
```python
def strsum(group):
    return ''.join([str(x) for x in group.values])
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 data frame as tabular string:
```
df.to_string()
```

Convert a data frame to a dictionary:
```
df.to_dict()
```

Save a data frame as an Excel table:
```
df.to_excel('output.xlsx')
```

(requires package xlwt)

Visualization

Import matplotlib:
```
import pylab 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)
```

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

Set tick marks:
```
labels = ['A', 'B', 'C', 'D']
positions = [1.0, 2.0, 3.0, 4.0]
plt.xticks(positions, labels)
plt.yticks(positions, labels)
```
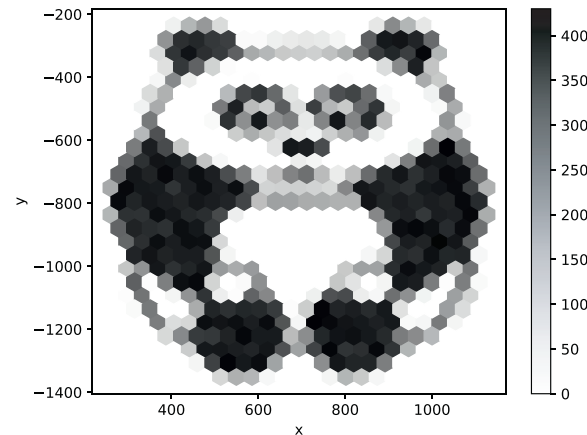
Select area to plot:
```
plt.axis([0.0, 2.5, 0.0, 10.0])
# [from x, to x, from y, to y]
```

Label diagram and axes:
```
plt.title('Correlation')
plt.xlabel('Nunstück')
plt.ylabel('Slotermeyer')
```

Save most recent diagram:
```
plt.savefig('plot.png')
plt.savefig('plot.png', dpi=300)
plt.savefig('plot.svg')
```


Text by Kristian Rother, Thomas Lotze (CC-BY-SA 4.0)

https://www.veit-schiele.de/seminare

Pandas cheat sheet

All of the following code examples refer to this table:

df=	col1	col2
A	1	4
B	2	5
C	3	6

Getting started

Import pandas:
```
import pandas as pd
```

Create a series:
```
s = pd.Series([1, 2, 3], index=['A', 'B', 'C'],
              name='col1')
```

Create a data frame:
```
data = [[1, 4], [2, 5], [3, 6]]
index = ['A', 'B', 'C']
df = pd.DataFrame(data, index=index,
             columns=['col1', 'col2'])
```

Load a data frame:
```
df = pd.read_csv('filename.csv',
            sep=',',
            names=['col1', 'col2'],
            index_col=0,
            encoding='utf-8',
            nrows=3)
```

Selecting rows and columns

Select single column:
```
df['col1']
```