Pandas

- Concepts
  - Series
  - DataFrame
- Series
  - Creation
    - From scalar
    - From dict
    - From numpy array
  - Operations
    - Element Access
    - List Slice Access
    - Math
- DataFrame
  - Import and Export
    - CSV
    - Excel
  - Access by Column
    - By Column Name
    - By Column Index
    - As Attribute Name
  - Access by Row
    - Select Row as Object
    - Select Rows as Series
    - Select Non-contiguous Rows as Series
    - Select Individual Column from Row
    - Select Individual Columns from Rows
    - Select Rows by Condition
  - Manipulation
    - Rename Columns
    - Re-arrange Columns
    - Change Index
    - Delete Columns
    - Delete Rows
    - Delete Rows by Condition
    - Insert Columns

- Insert Rows
- Visualization
  - Series Plot
  - Bar Plot
  - Histogram
  - Box
  - Scatter
  - Heat Map (HexBin)
  - Pie
  - Area
- Array Transformations
  - Transpose
  - Vertical Concatenation (normal)
  - Horizontal Concatenation
  - Split Horizontally
  - Split Vertically

Concepts

Series

Series is a one-dimensional labeled array capable of holding any data type (integers, strings, floating point numbers, Python objects, etc.). The axis labels are collectively referred to as the **index**.

DataFrame

DataFrame is a 2-dimensional labeled data structure with columns of potentially different types. You can think of it like a spreadsheet or SQL table, or a dict of Series objects. It is generally the most commonly used pandas object. Like Series, DataFrame accepts many different kinds of input:

- Dict of 1D ndarrays, lists, dicts, or Series
- 2-D numpy.ndarray
- Structured or record ndarray
- A Series
- Another DataFrame

Along with the data, you can optionally pass index (row labels) and columns (column labels) arguments. If you pass an index and / or columns, you are guaranteeing the index and / or

columns of the resulting DataFrame. Thus, a dict of Series plus a specific index will discard all data not matching up to the passed index.
If axis labels are not passed, they will be constructed from the input data based on common sense rules.

Series

Creation

From scalar

```
pd.Series(3.14)
0    3.14
dtype: float64

pd.Series(3.14, index=['a','b','c'])
a    3.14
b    3.14
c    3.14
dtype: float64
```

From dict

```
d = {'John':'Emma', 'Edward':'Molly', 'Alex':'Iris'}
pd.Series(d)
Alex       Iris
Edward    Molly
John       Emma
dtype: object
```

From numpy array

```
arr=np.random.randn(5)
pd.Series(arr)
0    1.281289
1    1.235460
2    1.106290
3    0.863517
4   -0.430387
dtype: float64

arr=np.random.randn(5)
pd.Series(arr, index=['h', 'e', 'l', 'l', 'o'])
```

```
h    0.722203
e    0.104977
l    0.508504
l   -0.998397
o    0.540913
dtype: float64
```

Operations

Element Access

Whatever you can do with a numpy array, you can do with a pandas array.

```
arr=np.random.randn(5)
srs=pd.Series(arr)
print(srs)
print(srs[1])
0    0.708338
1   -0.080908
2   -0.356362
3   -1.394590
4   -0.514729
dtype: float64
-0.0809084490034
```

List Slice Access

```
arr=np.random.randn(5)
srs=pd.Series(arr)
print(srs[2:])
2   -0.356362
3   -1.394590
4   -0.514729
dtype: float64
```

Math

```
arr=np.random.randn(5)
srs=pd.Series(arr)
print(srs ** 2)
0    0.557996
1    0.070252
2    0.637656
3    0.430528
```

4 2.447004
dtype: float64

NOTE: Notice how pow of 2 is applied to every element in the series.

DataFrame

Import and Export

CSV

data=pd.read_csv('Data/customers.csv')
data

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country	
0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico

data.to_csv('Data/out.csv', index=False)

Excel

data=pd.read_excel('Data/customers.xlsx') # add sheetname='..' arg to load a specific sheet
Data

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country	

0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	German y
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico

data.to_excel('Data/out.xlsx', index=False)

Access by Column

By Column Name

```
data=pd.read_excel('Data/customers.xlsx')
data['CustomerName']
```

```
0                Alfreds Futterkiste
1      Ana Trujillo Emparedados y helados
2              Antonio Moreno Taquera
3                  Around the Horn
4                  Berglunds snabbkp
5              Blauer See Delikatessen
6                  Blondel pre et fils
```

```
data=pd.read_csv('Data/customers.csv')
data[['City', 'CustomerName']]
```

	City	CustomerName
0	Berlin	Alfreds Futterkiste
1	Mxico D.F.	Ana Trujillo Emparedados y helados
2	Mxico D.F.	Antonio Moreno Taquera

3	London	Around the Horn
4	Lule	Berglunds snabbkp

By Column Index

data=pd.read_csv('Data/customers.csv')
data.iloc[:, [1,5,2]] # MUST USE DOUBLE SQUARE BRACKETS WHEN ACCESSING MORE THAN 1 INDEX

	CustomerName	PostalCode	ContactName
0	Alfreds Futterkiste	12209	Maria Anders
1	Ana Trujillo Emparedados y helados	5021	Ana Trujillo
2	Antonio Moreno Taquera	5023	Antonio Moreno
3	Around the Horn	WA1 1DP	Thomas Hardy

As Attribute Name

data=pd.read_csv('Data/customers.csv')
data.CustomerName

```
0                 Alfreds Futterkiste
1      Ana Trujillo Emparedados y helados
2                 Antonio Moreno Taquera
3                      Around the Horn
4                      Berglunds snabbkp
5                 Blauer See Delikatessen
6                      Blondel pre et fils
```

Access by Row

Select Row as Object

data=pd.read_csv('Data/customers.csv')
data.loc[0]

```
CustomerID                  1
CustomerName    Alfreds Futterkiste
ContactName         Maria Anders
Address             Obere Str. 57
City                    Berlin
PostalCode               12209
Country                 Germany
Name: 0, dtype: object
```

Select Rows as Series

```
data=pd.read_csv('Data/customers.csv')
data.loc[0:4]
```

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico
3	4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
4	5	Berglunds snabbkp	Christina Berglund	Berguvsvgen 8	Lule	S-958 22	Sweden

Select Non-contiguous Rows as Series

```
data=pd.read_csv('Data/customers.csv')
data.loc[[0,4]] # MUST BE DOUBLE SQUARE BRACKETS
```

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country

0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
4	5	Berglunds snabbkp	Christina Berglund	Berguvsvgen 8	Lule	S-958 22	Sweden

Select Individual Column from Row

```
data=pd.read_csv('Data/customers.csv')
data.iloc[0,4] # first is row number, second is column
```
'Berlin'

```
data=pd.read_csv('Data/customers.csv')
data.iloc[0,'City'] # first is row number, second is column… NOTE THIS IS ILOC, NOT LOC
```
'Berlin'

Select Individual Columns from Rows

```
data=pd.read_csv('Data/customers.csv')
data.iloc[0:4,4] # first is row, second is column
```
0 Berlin
1 Mxico D.F.
2 Mxico D.F.
3 London
Name: City, dtype: object

```
data=pd.read_csv('Data/customers.csv')
data.iloc[0:4,4:6] # first is row, second is column
```

	City	PostalCode
0	Berlin	12209
1	Mxico D.F.	5021
2	Mxico D.F.	5023
3	London	WA1 1DP

```
data=pd.read_csv('Data/customers.csv')
data.iloc[0:4][['City', 'PostalCode']]
```

	City	PostalCode
0	Berlin	12209
1	Mxico D.F.	5021
2	Mxico D.F.	5023
3	London	WA1 1DP

```
data=pd.read_csv('Data/customers.csv')
data.loc[0:4, 'CustomerName':'City'] # first is row, second is column  NOTE THIS IS LOC NOT ILOC
```

	CustomerName	ContactName	Address	City
0	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin
1	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.
2	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.
3	Around the Horn	Thomas Hardy	120 Hanover Sq.	London
4	Berglunds snabbkp	Christina Berglund	Berguvsvgen 8	Lule

Select Rows by Condition

```
data=pd.read_csv('Data/customers.csv')
data.loc[data.Country == 'Germany']
```

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
5	6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
16	17	Drachenblut Delikatessend	Sven Ottlieb	Walserweg 21	Aachen	52066	Germany
24	25	Frankenversand	Peter Franken	Berliner Platz 43	Mnchen	80805	Germany

```
data=pd.read_csv('Data/customers.csv')
data.loc[data.Country.isin(['Germany', 'Mexico'])]
```

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico
5	6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany

Manipulation

Rename Columns

```
data=pd.read_csv('Data/customers.csv')
data=data.rename(columns={'ContactName': 'SalesPerson'})
```

data

	CustomerID	CustomerName	SalesPerson	Address	City	PostalCode	Country
0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico
3	4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK

Re-arrange Columns

```
data=pd.read_csv('Data/customers.csv')
cols=data.columns.tolist()
cols=cols[::-1] # reverse the list
data[cols] # or data.loc[:, cols]
```

	Country	PostalCode	City	Address	ContactName	CustomerName	CustomerID
0	Germany	12209	Berlin	Obere Str. 57	Maria Anders	Alfreds Futterkiste	1
1	Mexico	5021	Mxico D.F.	Avda. de la Constitucin 2222	Ana Trujillo	Ana Trujillo Emparedados y helados	2
2	Mexico	5023	Mxico D.F.	Mataderos 2312	Antonio Moreno	Antonio Moreno Taquera	3
3	UK	WA1 1DP	London	120 Hanover Sq.	Thomas Hardy	Around the Horn	4

Change Index

```
data=pd.read_csv('Data/customers.csv')
data=data.set_index('CustomerID', verify_integrity=True)
data
```

NOTE: Notice how the original index has been abandoned… CustomerID can be used as the index now. You can also set this to a non-int column (e.g. CustomerName).

NOTE: Index should be of unique values. If it isn't unique, you may get back multiple rows when you access a single index. This is wrong because people usually associate an index with a single value, not multiple values. If you want duplicates, set verify_integrity=False.

	CustomerName	ContactName	Address	City	PostalCode	Country
CustomerID						
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK

Delete Columns

```
data=pd.read_csv('Data/customers.csv')
data=data.drop('Address', 1)  # The 1 here means you're dropping a row
data
```

	CustomerID	CustomerName	ContactName	City	PostalCode	Country

0	1	Alfreds Futterkiste	Maria Anders	Berlin	12209	Germany
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mxico D.F.	5023	Mexico
3	4	Around the Horn	Thomas Hardy	London	WA1 1DP	UK
4	5	Berglunds snabbkp	Christina Berglund	Lule	S-958 22	Sweden

```
data=pd.read_csv('Data/customers.csv')
data=data.drop(['Address', 'Country', 'PostalCode'], 1)
data
```

	CustomerID	CustomerName	ContactName	City
0	1	Alfreds Futterkiste	Maria Anders	Berlin
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mxico D.F.
2	3	Antonio Moreno Taquera	Antonio Moreno	Mxico D.F.
3	4	Around the Horn	Thomas Hardy	London

Delete Rows

```
data=pd.read_csv('Data/customers.csv')
data=data.drop(2, 0)  # 0 means row
data
```

NOTE: Notice how row 2 is gone

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
0	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
3	4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
4	5	Berglunds snabbkp	Christina Berglund	Berguvsvgen 8	Lule	S-958 22	Sweden

```
data=pd.read_csv('Data/customers.csv')
data=data.drop([0,1,2,3,4], 0) # 0 means row
data
```

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country	
5	6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
6	7	Blondel pre et fils	Frdrique Citeaux	24, place Klber	Strasbourg	67000	France
7	8	Blido Comidas preparadas	Martn Sommer	C/ Araquil, 67	Madrid	28023	Spain
8	9	Bon app'	Laurence Lebihans	12, rue des Bouchers	Marseille	13008	France

Delete Rows by Condition

```
data=pd.read_csv('Data/customers.csv')
data=data[data.Country!='Germany']
```

data

	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico
3	4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
4	5	Berglunds snabbkp	Christina Berglund	Berguvsvgen 8	Lule	S-958 22	Sweden

Insert Columns

data=pd.read_csv('Data/customers.csv')
data.insert(0, 'New column', data.City) # INPLACE -- DOES NOT RETURN NEW DATAFRAME
data

	New column	CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
0	Berlin	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
1	Mxico D.F.	2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitucin 2222	Mxico D.F.	5021	Mexico
2	Mxico D.F.	3	Antonio Moreno Taquera	Antonio Moreno	Mataderos 2312	Mxico D.F.	5023	Mexico
3	London	4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK

Insert Rows

```
data=pd.read_csv('Data/customers.csv')

# Create row as dict
info={'CustomerID': 420,
      'CustomerName': 'b',
      'ContactName': 'c',
      'Address': 'd',
      'City': 'e',
      'PostalCode': 'f',
      'Country': 'g'}

# Turn the dict into a DataFrame with a single row.
# We'll be inserting at index 3 so give this row an index of 3.
line=pd.DataFrame(info, index=[3]) # create at index=3

# Concatenate the original DataFrame with the new DataFrame
data=pd.concat([
    data.iloc[:2],
    line,
    data.iloc[3:]
])

# Display
data
```

NOTE: Notice in the output how there are now 2 rows with index 3. To fix this, call data.reset_index(drop=True). It'll re-number the indexes from 0 onward.

	Address	City	ContactName	Country	CustomerID	CustomerName	PostalCode
0	Obere Str. 57	Berlin	Maria Anders	Germany	1	Alfreds Futterkiste	12209
1	Avda. de la Constitucin 2222	Mxico D.F.	Ana Trujillo	Mexico	2	Ana Trujillo Emparedados y helados	5021
3	d	e	c	g	420	b	f

3	120 Hanover Sq.	Londo n	Thomas Hardy	UK	4	Around the Horn	WA1 1DP
4	Berguvsvgen 8	Lule	Christina Berglund	Sweden	5	Berglunds snabbkp	S-958 22

Visualization

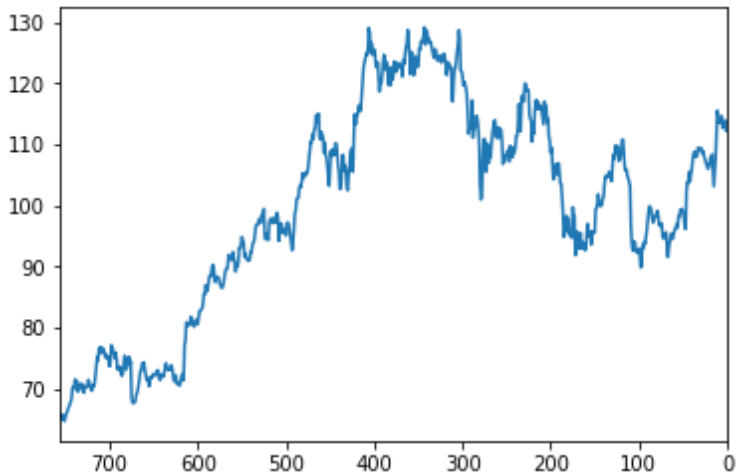
NOTE: The dataset being used here is apple financial data.

NOTE: Before doing any of this in jupyter notebook, you need to include matplotlib's magic jupyter notebook thing to display matplotlib plots…

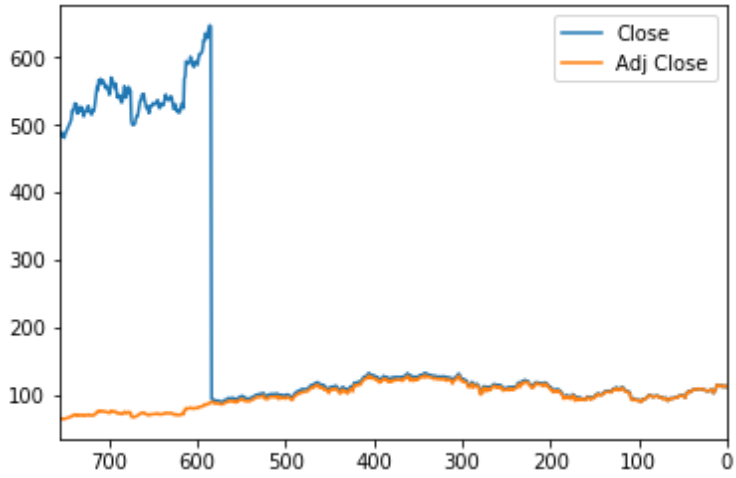
```
import numpy as np
import pandas as pd
%matplotlib inline

# Uncomment the following lines for prettier plots
# import matplotlib
#
# matplotlib.style.use('ggplot')
```

Series Plot

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] # reverse rows so that rows are earliest to latest
data['Adj Close'].plot()
```



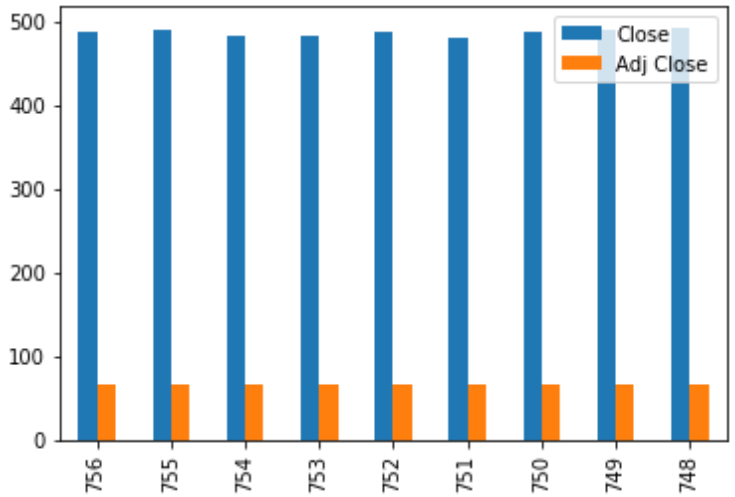
```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] # reverse rows so that rows are earliest to latest
data[['Close', 'Adj Close']].plot()
```



Bar Plot

NOTE: Never make a bar plot on more than 10 observations -- results will be useless.

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] # reverse rows so that rows are earliest to latest
data=data[0:9]
data[['Close', 'Adj Close']].plot.bar()
```

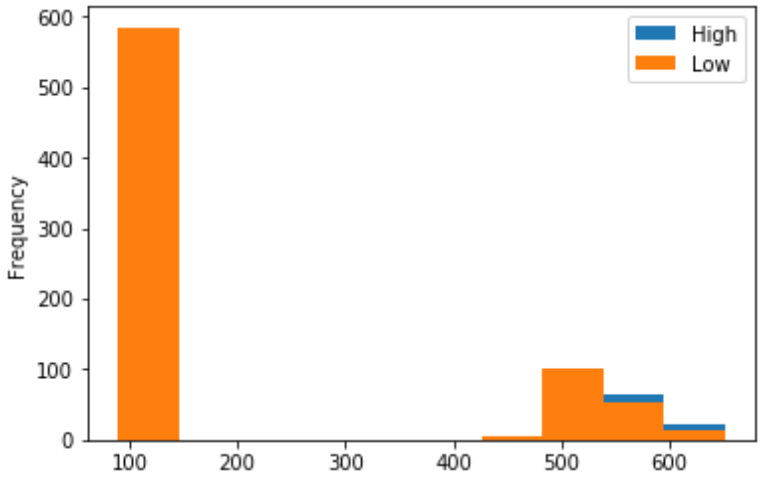


Histogram

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
```

```
data[['High', 'Low']].plot.hist()
```

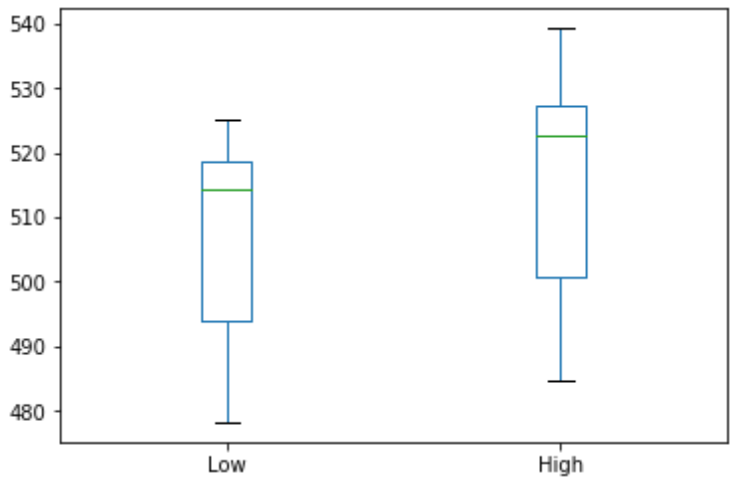
NOTE: Notice how Low covers up high. You can handle this by either having these 2 as separate plots, setting alpha=0.5 in hist() for some transparency (although this still looks not good), or setting stacked=True in hist() (to get them to stack on top of each other instead of overlapping).




```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data[['Low', 'High']].plot.hist(stacked=True)
```

Box

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data[:40]
data[['Low', 'High']].plot.box()
```

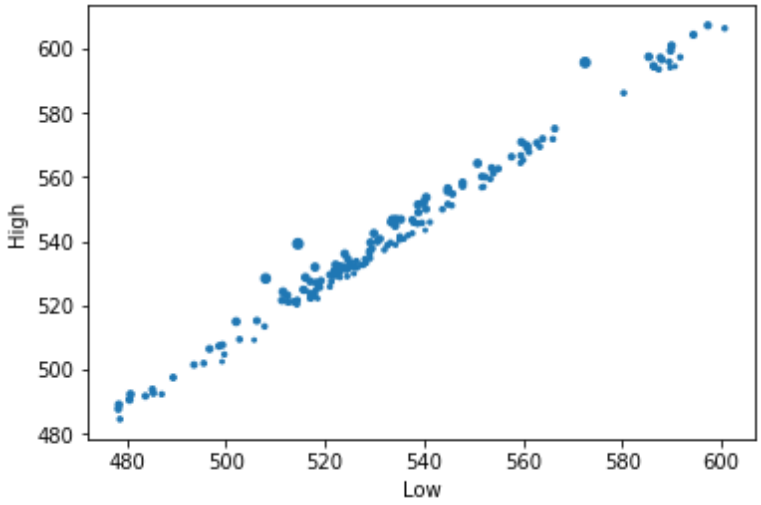


Scatter

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data[:40]
data[['Low', 'High']].plot.scatter(x='Low', y='High')
```

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data[:160]
data[['Low', 'High', 'Volume']].plot.scatter(x='Low', y='High', s=data.High-data.Low)
```

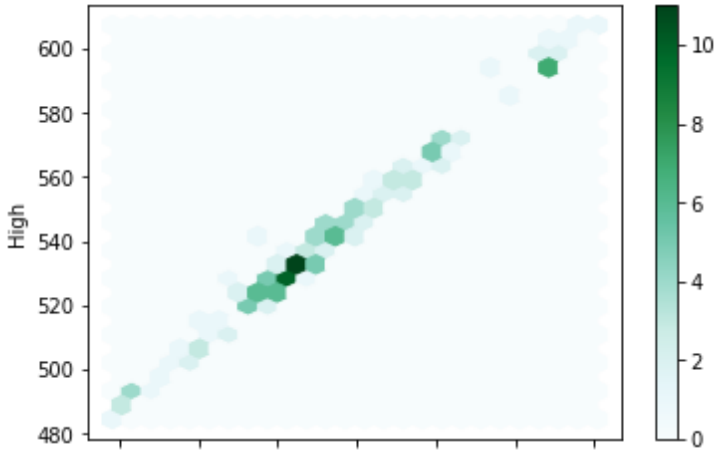
NOTE: s controls the size of the dot being plotted



Heat Map (HexBin)

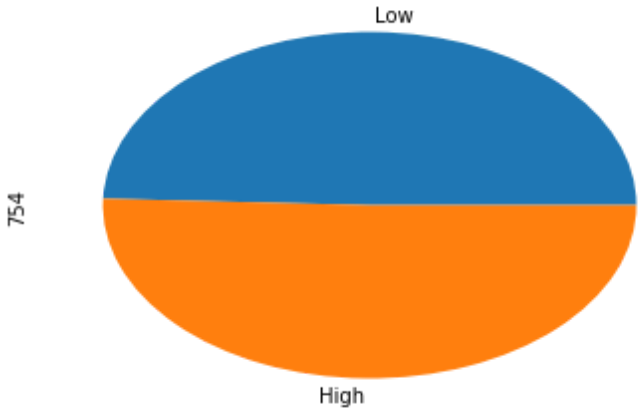
```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data[:160]
data[['Low', 'High']].plot.hexbin(x='Low', y='High', gridsize=25)
```

NOTE: This is showing you the concentration of observations at some point. The deeper the color, the more the area is concentrated with observations.

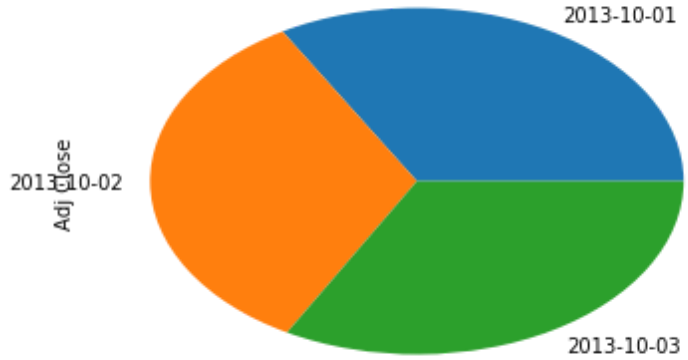


Pie

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data[:160]
data[['Low', 'High']].iloc[2].plot.pie()
```

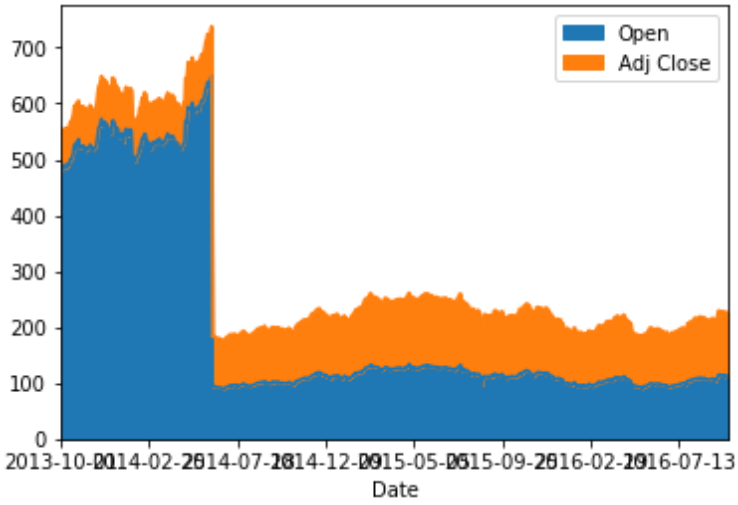



```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data[:3]
data=data.set_index('Date', verify_integrity=True)
data['Adj Close'].plot.pie()
```



Area

```
data=pd.read_csv('Data/aapl.csv')
data=data[::-1] #  reverse rows so that rows are earliest to latest
data=data.set_index('Date', verify_integrity=True)
data[['Open', 'Adj Close']].plot.area()
```



Array Transformations

Transpose

```
import numpy as np
a=np.arange(28).reshape(4,7)
a.T  # transpose
array([[ 0,  7, 14, 21],
       [ 1,  8, 15, 22],
       [ 2,  9, 16, 23],
       [ 3, 10, 17, 24],
       [ 4, 11, 18, 25],
       [ 5, 12, 19, 26],
       [ 6, 13, 20, 27]])
```

```
import numpy as np
a=np.arange(28).reshape(4,7)
a.ravel()  # unravels multidimensional array to 1 dimensional array
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27])
```

```
import numpy as np
a=np.arange(28).reshape(4,7)
a.reshape(2,2,7) # new shape must contain same number of elements, e.g. 4*7=2*2*7
array([[[ 0,  1,  2,  3,  4,  5,  6],
        [ 7,  8,  9, 10, 11, 12, 13]],
```

```
[[14, 15, 16, 17, 18, 19, 20],
 [21, 22, 23, 24, 25, 26, 27]]])
```

Vertical Concatenation (normal)

```
import numpy as np
a=np.array([[1,2], [3,4]])
b=np.array([[5,6], [7,8]])
np.vstack((a,b))
array([[1, 2], [3, 4], [5, 6], [7, 8]])
```

Horizontal Concatenation

```
import numpy as np
a=np.array([[1,2], [3,4]])
b=np.array([[5,6], [7,8]])
np.hstack((a,b))
array([[1, 2, 5, 6], [3, 4, 7, 8]])
```

Split Horizontally

```
import numpy as np
a=np.array([[1,2], [3,4]])
b=np.array([[5,6], [7,8]])
h = np.hstack((a,b))
h_split = np.hsplit(h, 2)
print(h)
print(h_split)
[[1 2 5 6]
 [3 4 7 8]]
[array([[1, 2], [3, 4]]), array([[5, 6], [7, 8]])]
```

Split Vertically

```
import numpy as np
a=np.array([[1,2], [3,4]])
b=np.array([[5,6], [7,8]])
v = np.vstack((a,b))
v_split = np.vsplit(v, 2)
print(v)
print(v_split)
[[1 2]
 [3 4]
 [5 6]
```

```
 [7 8]]
[array([[1, 2], [3, 4]]), array([[5, 6], [7, 8]])]
```