Data Visualization

Plotting with matplotlib

A Basic Plot

```
import numpy as np
import matplotlib.pyplot as plt

# Create an array filled with numbers from 0 to 9
data = np.arange(10)
print(data)

# Create a plot of the data
plt.plot(data)

# show the plot
plt.show()
```

Another Basic Plot

```
# To avoid calling plt.show()
# we can use
%matplotlib
# in IPython

# Create an random data series
data = np.random.randn(50).cumsum()
print(data)

# Create a plot of the data
plt.plot(data)

# the plot is shown immediately
```

And Another Basic Plot

```
# Create an random data series
data = np.random.randn(50).cumsum()
print(data)

# Create a plot of the data
plt.plot(data)

# BUT that's in the same figure!

# So, use either
plt.clf()       # to clear a figure
# or
plt.close()     # to close a figure window.
```

Colors, Markers, and Line Styles

```
# Create a plot of the data
# as dashed line, circles and in green
plt.plot(data, linestyle='--', marker='o', color='g')

# or dotted line, triangles (up) and in red
data = np.random.randn(50).cumsum()
plt.plot(data, linestyle=':', marker='^', color='r')


plt.close()     # to close the figure window
```

Ticks, ...

```
plt.plot(np.random.randn(1000).cumsum())

# Change the x-axis ticks:
# where?
tick_loc = [0, 250, 500, 750, 1000]
# what?
tick_lab = ['one', 'two', 'three', 'four', 'five']

# set them:
plt.xticks(tick_loc, tick_lab, rotation=30,
fontsize='small')
```

..., Labels, ...

```
# Change the x-axis label:
plt.xlabel('Stages')

# Change the y-axis label:
plt.ylabel('Distance', rotation=90)

# Change the title:
plt.title('Not my first matplotlib plot!')

# Clear the figure:
plt.clf()
```

... and Legends

```
# Create three sets of random data
data = [[],[],[]]
for i in range(3):
   data[i] = np.random.randn(100).cumsum()

# define styles and labels
styles = ['ko-', 'gs--', 'b.-.']
labels = ['one', 'two', 'three']

# plot them
for i in range(3):
    plt.plot(data[i], styles[i], label=labels[i])

# Finally, the legend
plt.legend(loc='best')
```

Annotations and Drawing on a Plot

```
# some text somewhere
plt.text(10, 5, 'Just to say Hello!',
family='monospace', fontsize=10)

# more meaningful annotations
# e.g., peak of 'one'
y = data[0].max()
x = data[0].argmax()
plt.annotate('Peak One', xy=(x+0.1, y+0.1),
xytext=(x+1,y+1), arrowprops=dict(facecolor='black',
headwidth=4, width=2, headlength=4),
horizontalalignment='left', verticalalignment='top')

# see also:
# https://matplotlib.org/users/annotations.html
```

Time to Save the Plot

```
filename = 'UpDown.png'
plt.savefig(filename)

# and close it
plt.close()
```

Scatter Plot

```
xvals = np.random.randn(100).cumsum()
yvals = np.random.randn(100).cumsum()

# plot in scatter plot
plt.scatter(xvals, yvals)

# repeat
xvals = np.random.randn(100).cumsum()
yvals = np.random.randn(100).cumsum()

# plot in scatter plot
plt.scatter(xvals, yvals)
```

Scatter Plot (cont'd)

```
# random size of dots
sizes = np.random.randn(100) * 100

# repeat
xvals = np.random.randn(100).cumsum()
yvals = np.random.randn(100).cumsum()

# plot in scatter plot
plt.scatter(xvals, yvals, s=sizes)

# and close it
plt.close()
```

Histograms

```
vals = np.random.randn(100)

# Univariate Histogram
plt.hist(vals, alpha=0.5)

# with six / twenty bins
plt.hist(vals, bins=6, alpha=0.5)
plt.hist(vals, bins=20, alpha=0.5)
```

Figures and Subplots

```
# an empty figure
fig = plt.figure()

# explicitly create subplots as 2x2
# numbered starting from 1!
ax1 = fig.add_subplot(2, 2, 1)
ax2 = fig.add_subplot(2, 2, 2)
ax3 = fig.add_subplot(2, 2, 3)

# matplotlib draws on the last figure and subplot used
plt.plot(np.random.randn(50).cumsum(), 'k--')
```

Figures and Subplots (con't)

```
# draw into another subplot
# histogram
ax1.hist(np.random.randn(100), bins=20, color='k',
alpha=0.3)

# scatterplot
ax2.scatter(np.arange(30), np.arange(30) + 3 *
np.random.randn(30))

# select scatter plot
plt.subplot(2,2,2)
plt.text(5, 20, 'Positive Linear Relationship')

plt.close()
```

Grids of Subplots

```
# figure with 2x3 subplots
fig, axes = plt.subplots(2, 3)

# the axes are stored in an array
axes
# and can be easily indexed, e.g.,
axes[0,1]

# fill the subplots with histograms
for i in range(2):
    for j in range(3):
        axes[i, j].hist(np.random.randn(500), bins=50,
color='k', alpha=0.5)
```

Grids of Subplots (cont'd)

```
# Make histograms directly comparable by
# sharing same x-axis ticks and y-axis ticks.
# Have to do this WHEN creating the figure
fig, axes = plt.subplots(2, 3, sharex=True, sharey=True)

# fill the subplots with histograms
for i in range(2):
    for j in range(3):
        axes[i, j].hist(np.random.randn(500), bins=50,
color='k', alpha=0.5)

# change spacing around the subplots
plt.subplots_adjust(wspace=0, hspace=0)
```

Additional Readings

- Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython by Wes McKinney (pub. yr. 2017). Chapter 10.
- Machine Learning Mastery with Python by Jason Brownlee (pub. yr. 2017). Chapter 6.
- https://www.analyticsvidhya.com/blog/2016/01/12-pandas-techniques-python-data-manipulation/
- https://matplotlib.org/users/pyplot_tutorial.html
- https://matplotlib.org/api/markers_api.html#module-matplotlib.markers
- https://matplotlib.org/users/annotations.html

Additional Readings (cont'd)

- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html
- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.describe.html
- https://matplotlib.org/

- DataCamp:
  - Course: Intermediate Python for Data Science
    - Chapter: Matplotlib
  - Introduction to Data Visualization with Python
    - Chapter: Customizing Plots