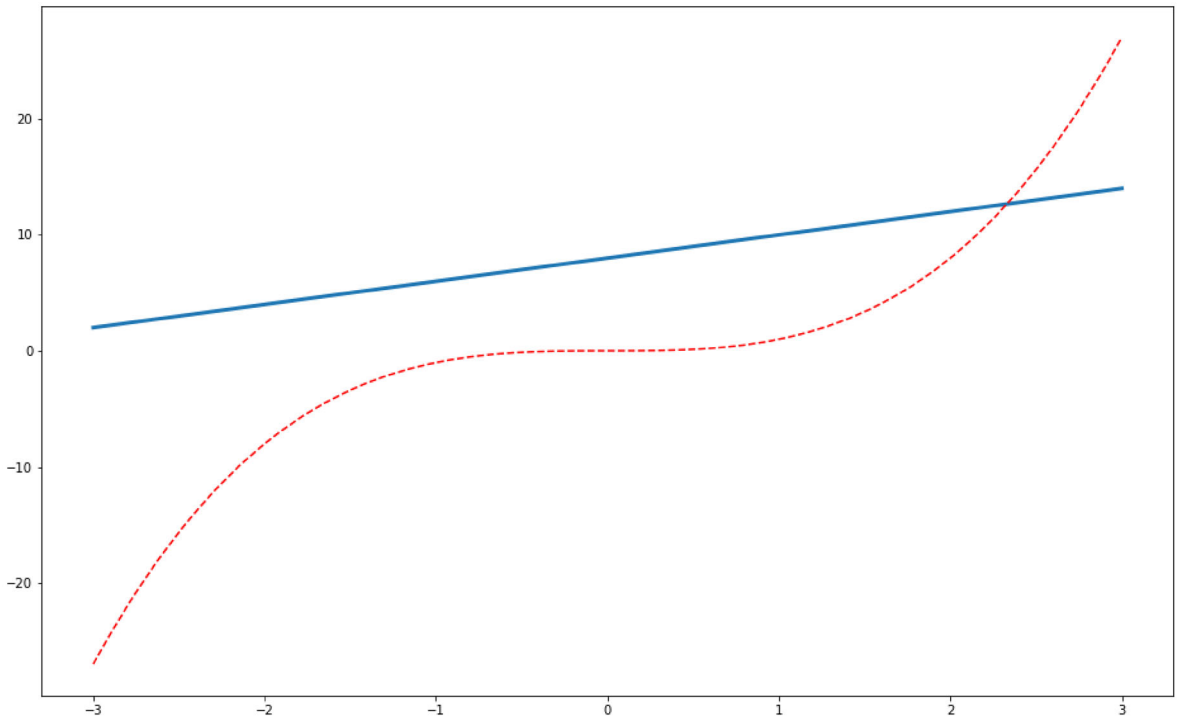
0708 Python / AI Programming Practice

Tutor: Nanyang Ye
Note Taker: Y. Qiu

Matplotlib

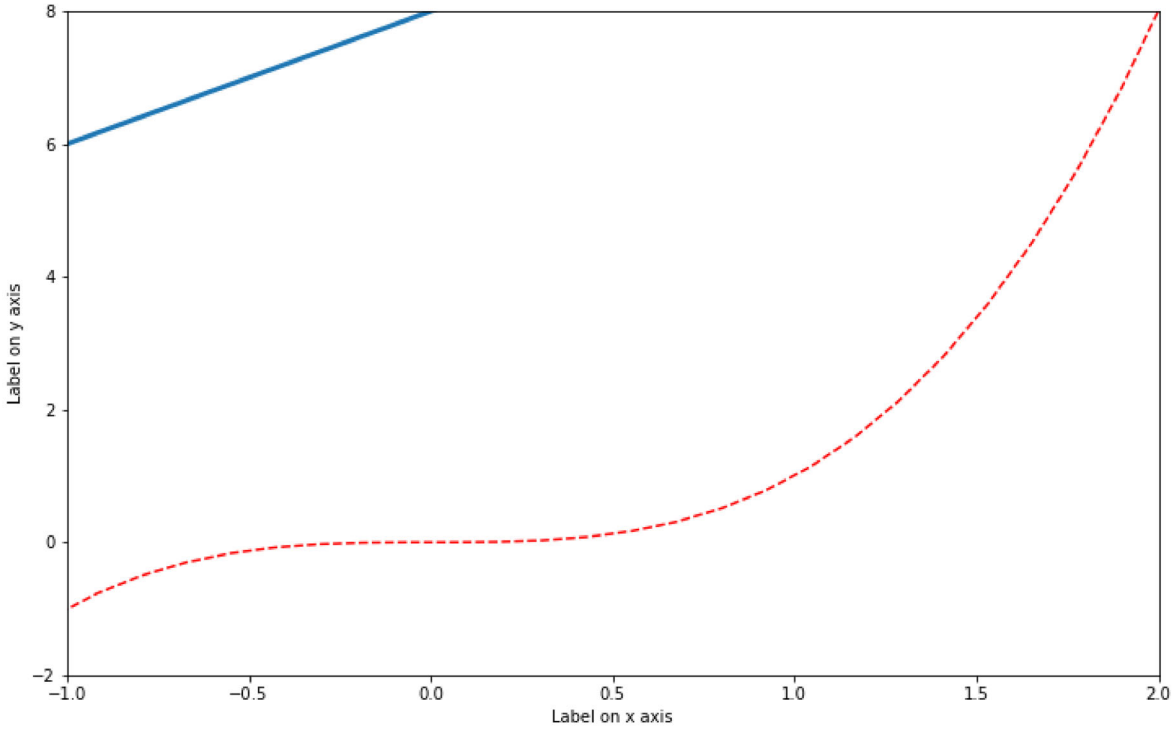
In [23]:

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

x = np.linspace(-3,3,50)
y1 = 2*x+8
y2 = x**3
y3 = np.exp(x)

plt.figure(num=1, figsize=(16,10))                    # num: the order of the graph
plt.plot(x, y1, linewidth = 3)
plt.plot(x, y2, color = 'red', linestyle= '--')
#plt.plot(x, y3)
plt.show()
```



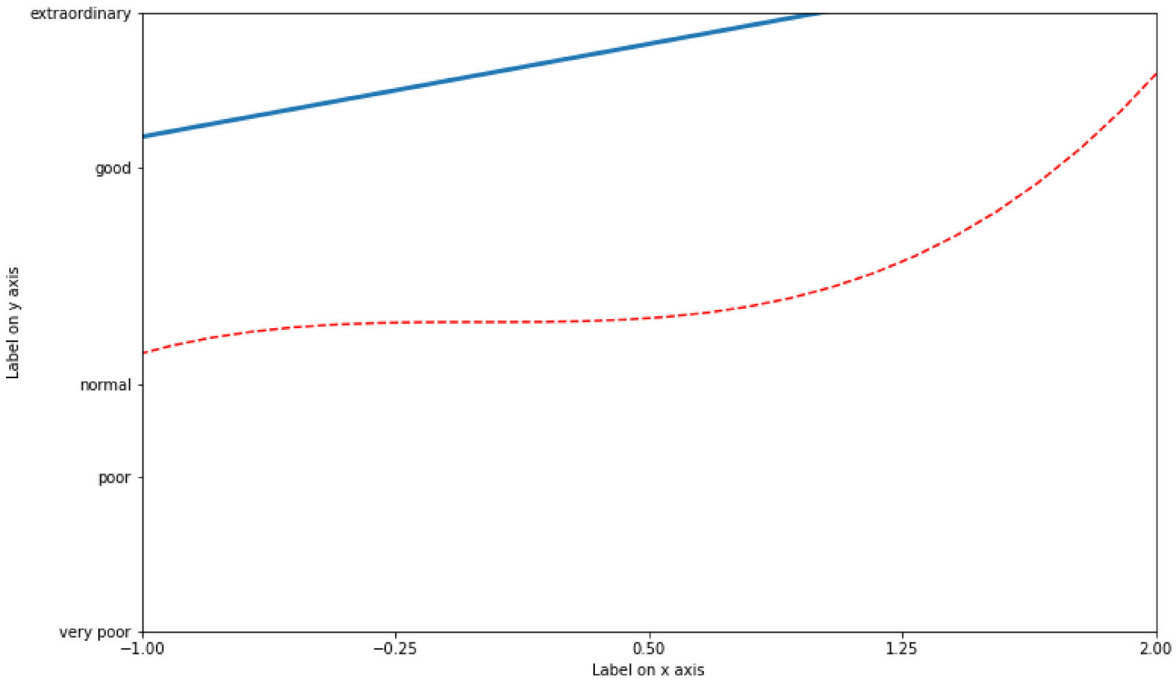
In [27]:

```
plt.figure(num=2, figsize=(12,7.5))                # num: the order of the graph
plt.plot(x, y1, linewidth = 3)
plt.plot(x, y2, color = 'red', linestyle= '--')

plt.xlim((-1,2))                                   # limit x range
plt.ylim((-2,8))
plt.xlabel("Label on x axis")
plt.ylabel("Label on y axis")
plt.show()
```



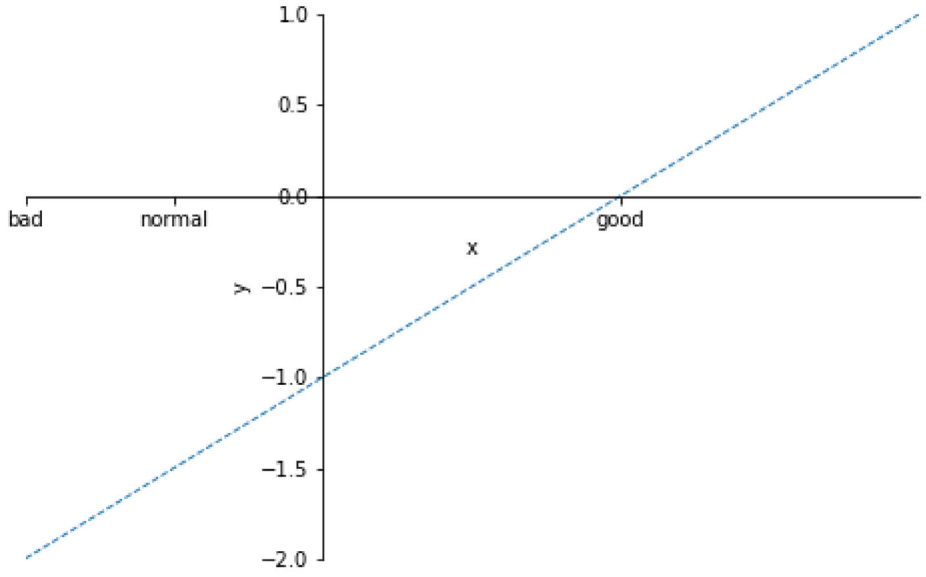
In [28]:

```
plt.figure(num=3, figsize=(12,7.5))                # num: the order of the graph
plt.plot(x, y1, linewidth = 3)
plt.plot(x, y2, color = 'red', linestyle= '--')

plt.xlim((-1,2))                                   # limit x range
plt.ylim((-2,8))
plt.xticks([-1,-0.25,0.5,1.25,2])
plt.yticks([-10,-5,-2,5,10],["very poor","poor","normal","good","extraordinary"])

plt.xlabel("Label on x axis")
plt.ylabel("Label on y axis")
plt.show()
```



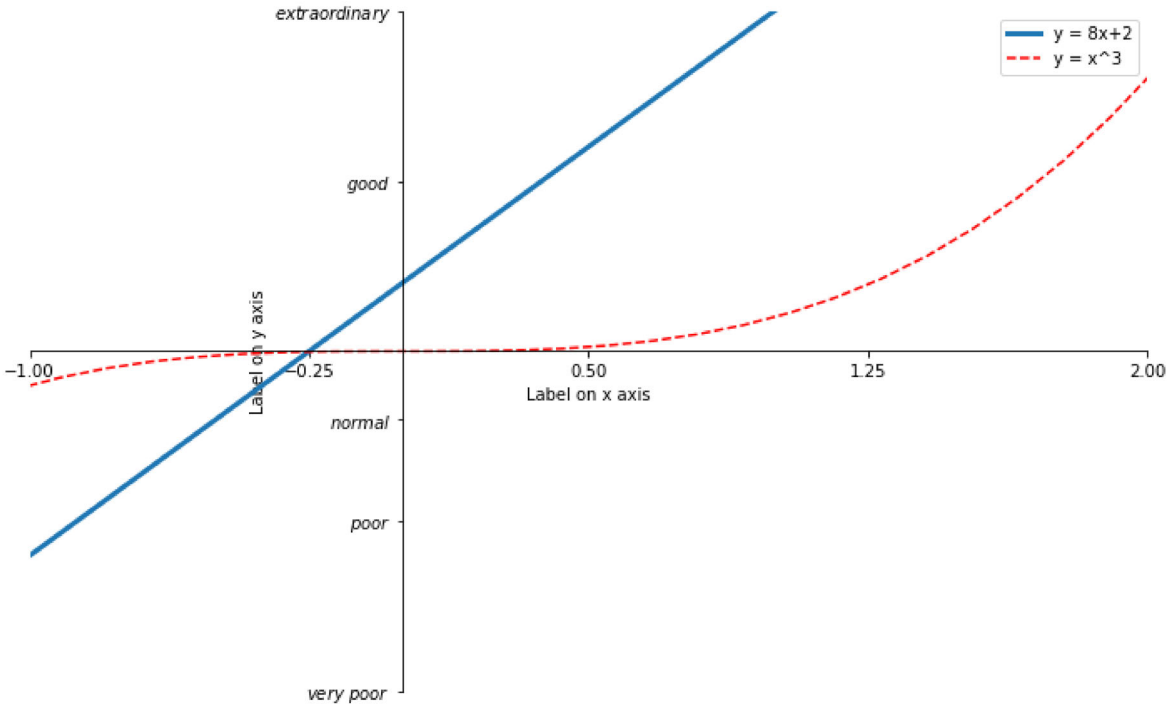
In [32]:

```
# Example 01

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(-1,2,50)
y = x-1

plt.figure(num=4, figsize = (8,5))
plt.plot(x, y, linewidth = 1, linestyle = '--')
plt.xlabel("x")
plt.ylabel("y")
plt.xlim((-1,2))
plt.ylim((-2,1))
plt.xticks([-1,-0.5,1],["bad","normal","good"])

ax = plt.gca()                                  # move the axis line
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')
ax.spines['left'].set_position(('data',0))
ax.spines['bottom'].set_position(('data',0))

plt.show()
```



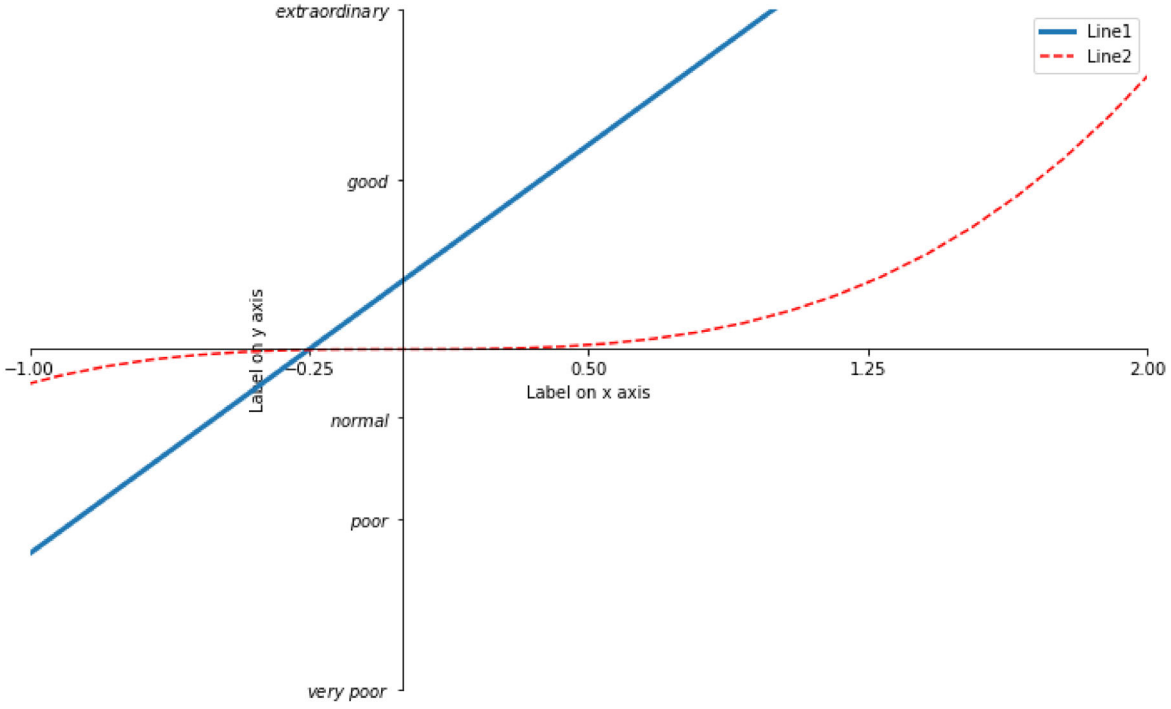
In [44]:

```
plt.figure(num=3, figsize=(12,7.5))                # num: the order of the graph

x = np.linspace(-3,3,50)
y1 = 8*x+2
y2 = x**3
y3 = np.exp(x)

l1 = plt.plot(x, y1, linewidth = 3, label = 'y = 8x+2')
l2 = plt.plot(x, y2, color = 'red', linestyle= '--', label = 'y = x^3')

plt.xlim((-1,2))                                   # limit x range
plt.ylim((-2,8))
plt.xticks([-1,-0.25,0.5,1.25,2])
plt.yticks([-10,-5,-2,5,10],[r'$very\ poor$',r'$poor$',r'$normal$',r'$good$',r'$extraordinary$'])

plt.xlabel("Label on x axis")
plt.ylabel("Label on y axis")

ax = plt.gca()                                     # move the axis line
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')
ax.spines['left'].set_position(('data',0))
ax.spines['bottom'].set_position(('data',0))

plt.legend()
plt.show()
```



In [93]:

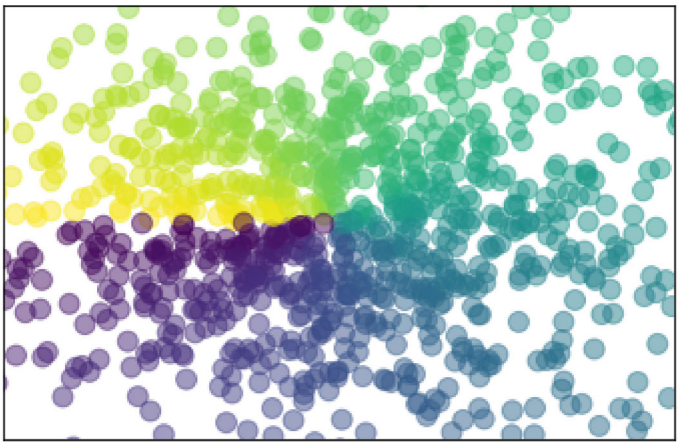
```
plt.figure(num=4, figsize=(12,7.5))                # num: the order of the graph

x = np.linspace(-3,3,50)
y1 = 8*x+2
y2 = x**3
y3 = np.exp(x)

l1, = plt.plot(x, y1, linewidth = 3, label = 'y = 8x+2')
                                    # plt.plot() returns two elements (as a list or as a tuple), ac
                                    # The comma is Python syntax that denotes either a single-eleme
                                    # i.e. l1, is a tuple

l2, = plt.plot(x, y2, color = 'red', linestyle= '--', label = 'y = x^3')

plt.xlim((-1,2))                            # limit x range
plt.ylim((-2,8))
plt.xticks([-1,-0.25,0.5,1.25,2])
plt.yticks([-10,-5,-2,5,10],[r'$very\ poor$',r'$poor$',r'$normal$',r'$good$',r'$extraordinary$'])

plt.xlabel("Label on x axis")
plt.ylabel("Label on y axis")

ax = plt.gca()                                     # move the axis line
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')
ax.spines['left'].set_position(('data',0))
ax.spines['bottom'].set_position(('data',0))

plt.legend(handles=[l1,l2], labels=['Line1','Line2'], loc='best')
plt.show()
```



Scatter Graph

In [61]:

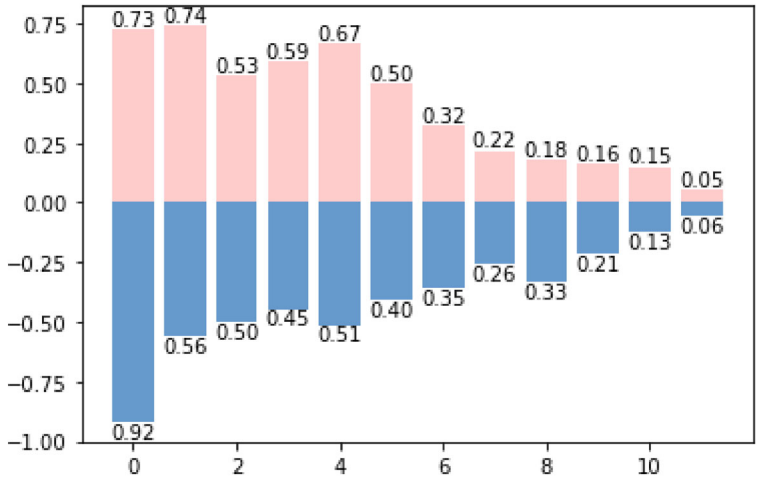
```
import numpy as np
import matplotlib.pyplot as plt

n = 1024
X = np.random.normal(0,1,n)
Y = np.random.normal(0,1,n)
T = np.arctan2(Y,X)           # color value

plt.scatter(X, Y, s = 100, c = T, alpha=.5)      # alpha: transformation
plt.xlim((-2,2))
plt.ylim((-2,2))
plt.xticks(())
plt.yticks(())        # ignore ticks

plt.show()
```



Bar Chart

In [64]:

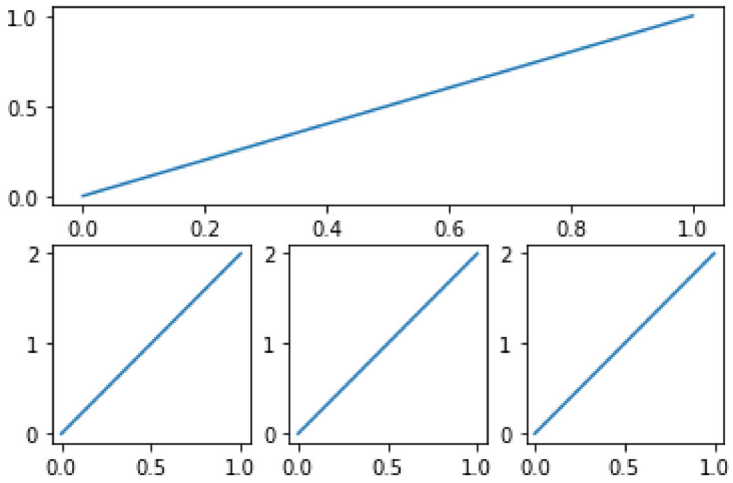
```
import numpy as np
import matplotlib.pyplot as plt

n = 12
X = np.arange(n)
Y1 = (1-X/float(n)) * np.random.uniform(0.5, 1.0, n)
Y2 = (1-X/float(n)) * np.random.uniform(0.5, 1.0, n)

plt.bar(X,Y1, facecolor = '#FFCCCC', edgecolor='none')
plt.bar(X,-Y2, facecolor = '#6699CC', edgecolor='none')

# Add label to each bar
for x,y in zip(X,Y1):              # zipping multiple iterable data structures as a pack ((x,y) as a
    plt.text(x, y, "%.2f" % y, ha = 'center', va = 'bottom')

for x,y in zip(X,Y2):
    plt.text(x, -y, "%.2f" % y, ha = 'center', va = 'top')

plt.show()
```



Subplots

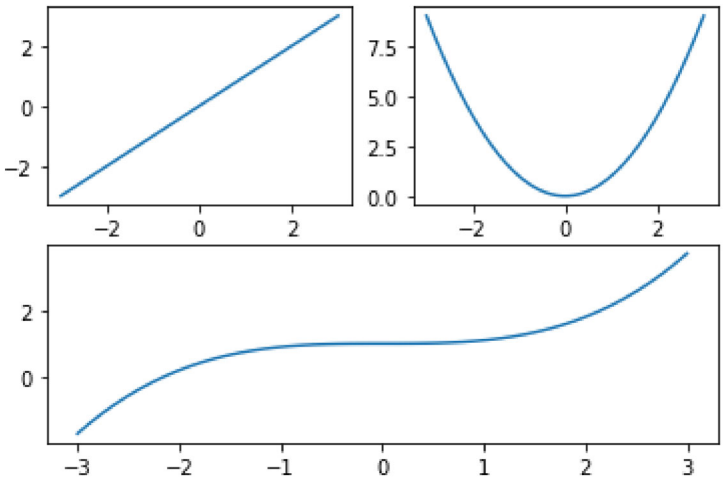
In [68]:

```
plt.subplot(2, 1, 1)
plt.plot([0, 1], [0, 1])

plt.subplot(234)                    # means "2, 3, 4"
plt.plot([0, 1], [0, 2])

plt.subplot(235)
plt.plot([0, 1], [0, 2])

plt.subplot(236)
plt.plot([0, 1], [0, 2])

plt.show()
```



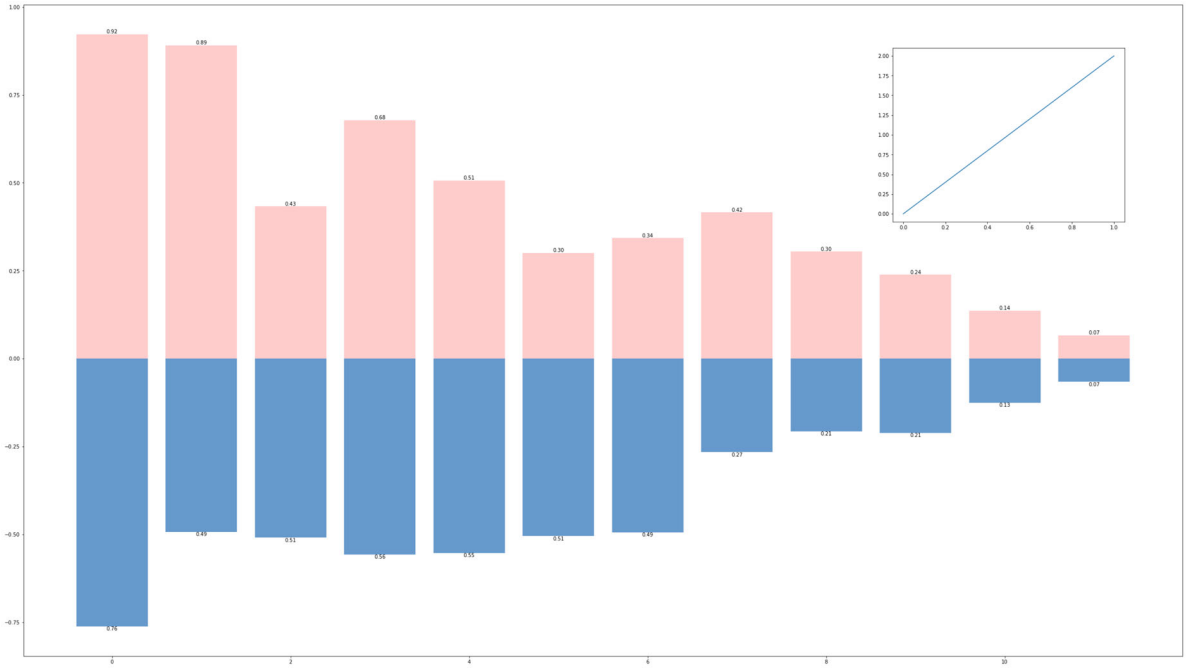
In [91]:

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

x = np.linspace(-3, 3, 50)
y1 = x
y2 = x**2
y3 = 0.1*x**3 + 1

plt.subplot(2,2,1)
plt.plot(x,y1)

plt.subplot(2,2,2)
plt.plot(x,y2)

plt.subplot(2,1,2)
plt.plot(x,y3)

plt.show()
```



In [89]:

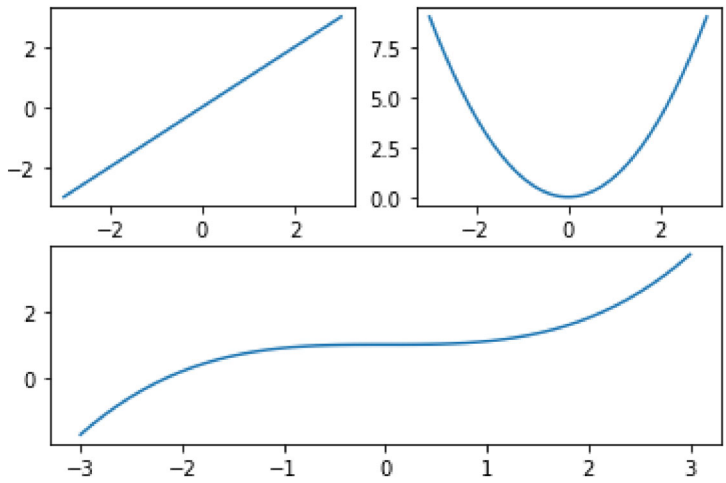
```
import numpy as np
import matplotlib.pyplot as plt

fig = plt.figure(figsize = (8,6))

n = 12
X = np.arange(n)
Y1 = (1-X/float(n)) * np.random.uniform(0.5, 1.0, n)
Y2 = (1-X/float(n)) * np.random.uniform(0.5, 1.0, n)


left, bottom, width, height = 0, 0, 4, 3
fig01 = fig.add_axes([left, bottom, width, height])
fig01.bar(X,Y1, facecolor = '#FFCCCC', edgecolor='none')
fig01.bar(X,-Y2, facecolor = '#6699CC', edgecolor='none')

# Add label to each bar
for x,y in zip(X,Y1):              # zipping multiple iterable data structures as a pack ((x,y) as a
    plt.text(x, y, "%.2f" % y, ha = 'center', va = 'bottom')

for x,y in zip(X,Y2):
    plt.text(x, -y, "%.2f" % y, ha = 'center', va = 'top')


left, bottom, width, height = 3, 2, 0.8, 0.8
fig01 = fig.add_axes([left, bottom, width, height])
fig01.plot([0,1],[0,2])

plt.show()
```



twinx

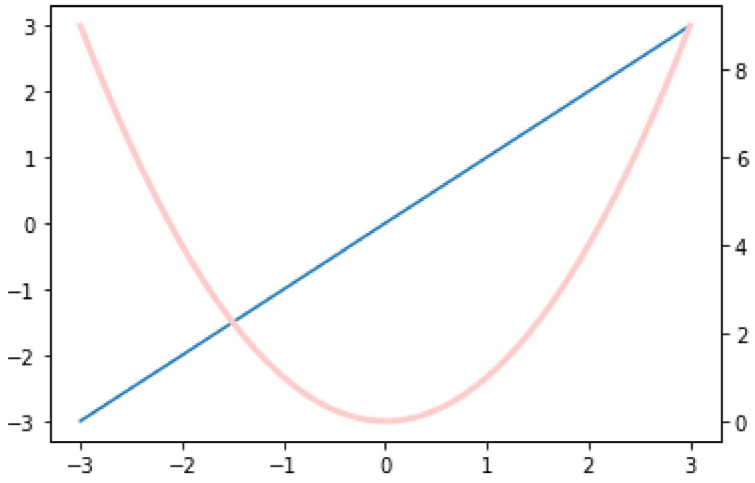
In [91]:

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

x = np.linspace(-3, 3, 50)
y1 = x
y2 = x**2
y3 = 0.1*x**3 + 1

plt.subplot(2,2,1)
plt.plot(x,y1)

plt.subplot(2,2,2)
plt.plot(x,y2)

plt.subplot(2,1,2)
plt.plot(x,y3)

plt.show()
```



In [96]:

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

fig = plt.figure()

ax1 = plt.subplot()
ax2 = ax1.twinx()


x = np.linspace(-3, 3, 50)
y1 = x
y2 = x**2
y3 = 0.1*x**3 + 1

ax1.plot(x,y1)
ax2.plot(x,y2, color = '#FFCCCC',linewidth=3)

plt.show()
```

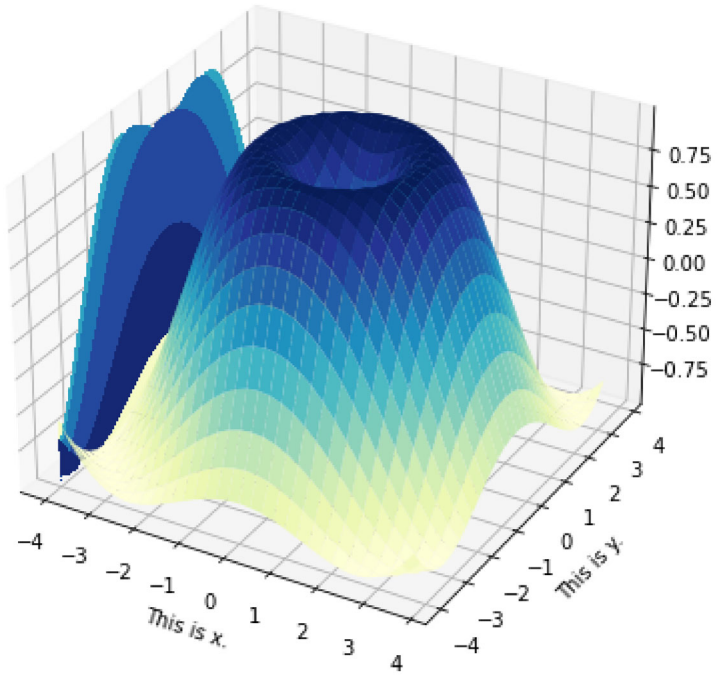


3D

Notice: X,Y must be a matrix (should be grided)

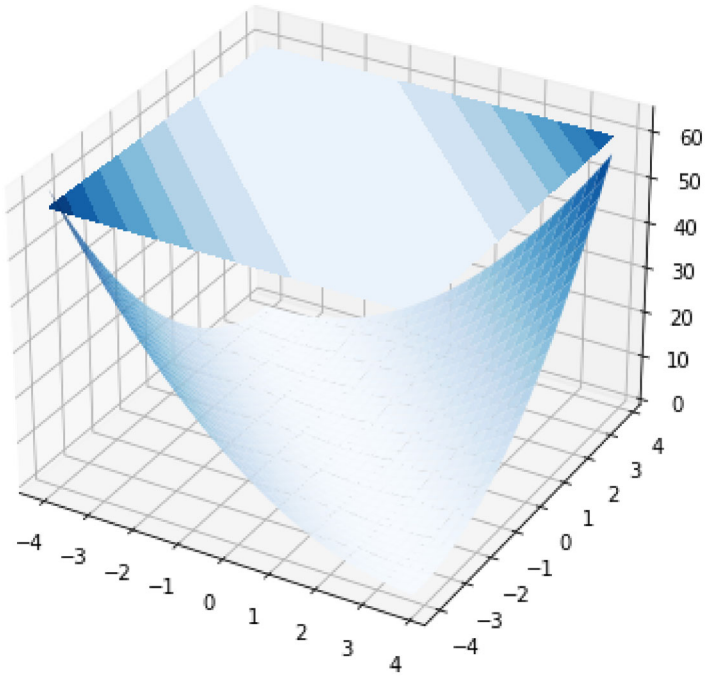
In [112]:

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize = (8,5))
ax = Axes3D(fig)

X = np.arange(-4, 4, .25)
Y = np.arange(-4, 4, .25)
X, Y = np.meshgrid(X, Y)            # IMPORTANT!
Z = np.sqrt(X**2 + Y**2)
Z = np.sin(Z)

ax.plot_surface(X, Y, Z, rstride = 1, cstride = 1, cmap = plt.get_cmap('YlGnBu'))
ax.contourf(X, Y, Z, zdir='x', offset=-4, cmap=plt.get_cmap('YlGnBu'))                    # projected
plt.xlabel("This is x.")
plt.ylabel("This is y.")

plt.show()
```



In [109]:

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize = (8,5))
ax = Axes3D(fig)

X = np.arange(-4, 4, .25)
Y = np.arange(-4, 4, .25)
X, Y = np.meshgrid(X, Y)            # IMPORTANT!
Z = (X+Y) ** 2

ax.plot_surface(X, Y, Z, rstride = 1, cstride = 1, cmap = plt.get_cmap('Blues'))
ax.contourf(X, Y, Z, zdir='z', offset=60, cmap=plt.get_cmap('Blues'))                    # projected to

plt.show()
```



Credits:

参考：https://www.bilibili.com/read/cv8555776/ 出处：bilibili

Note: In fact, plotly is making websites through plotly.js, thus it's JavaScript which is making these kind-of-stunning websites.