基本用法

本章知识点归纳如下：

1. 导入模块：import matplotlib.pyplot as plt
2. 定义图像窗口：plt.figure()
3. 画图：plt.plot(x, y)
4. 定义坐标轴范围：plt.xlim()/plt.ylim()
5. 定义坐标轴名称：plt.xlabel()/plt.ylabel()
6. 定义坐标轴刻度及名称：plt.xticks()/plt.yticks()
7. 设置图像边框颜色：ax = plt.gca() ax.spines[].set_color()
8. 调整刻度位置：ax.xaxis.set_ticks_position()/ax.yaxis.set_ticks_position()
9. 调整边框（坐标轴）位置：ax.spines[].set_position()

导入模块

使用import导入模块matplotlib.pyplot，并简写成plt；使用import导入模块numpy，并简写成np

In [1]:

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

然后创建两组数据，使用np.linspace定义x：范围是(-3,3)，个数是50，将产生一组（-3，3）内均匀分布的50个数；(x,y1)表示曲线1，(x,y2)表示曲线2。

In [2]:

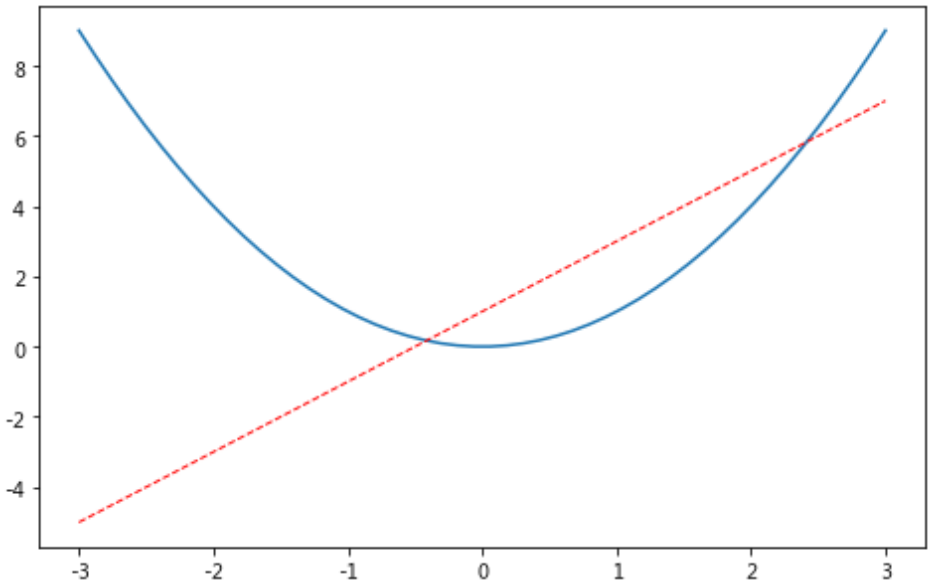
```
x = np.linspace(-3, 3, 50)
y1 = 2*x + 1
y2 = x**2
```

定义图像窗口并画图

在画图前使用plt.figure()定义一个图像窗口：编号为3；大小为(8, 5)；这两项参数可缺省。其中，num参数决定了程序运行后弹出的图像窗口名字，但在klab平台下不会显示。接着，我们使用plt.plot画出(x ,y2)曲线；使用plt.plot画(x ,y1)曲线，曲线的颜色属性(color)为红色；曲线的宽度(linewidth)为1.0；曲线的类型(linestyle)为虚线，除了虚线外，大家还可使用以下线性：'-'、'--'、'-.'、':' 。接着，我们使用plt.show()显示图像。

In [3]:

```
plt.figure(num=3, figsize=(8, 5))
plt.plot(x, y2)
plt.plot(x, y1, color='red', linewidth=1.0, linestyle='--')
plt.show()
```

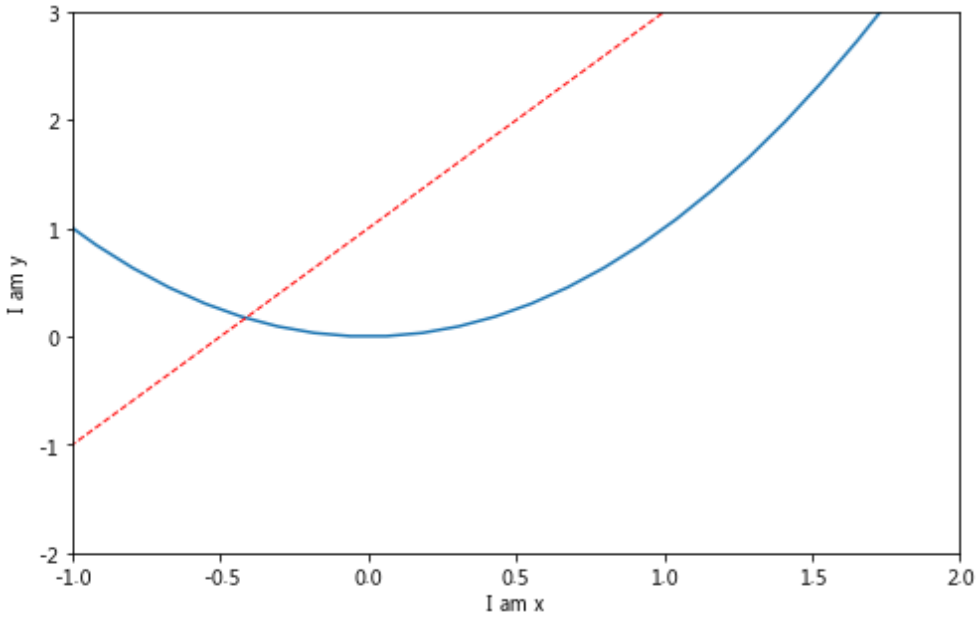


定义坐标轴名称及范围

使用plt.xlim设置x坐标轴范围：(-1, 2)； 使用plt.ylim设置y坐标轴范围：(-2, 3)； 使用plt.xlabel设置x坐标轴名称：'I am x'； 使用plt.ylabel设置y坐标轴名称：'I am y'；

In [6]:

```
plt.figure(num=3, figsize=(8, 5),)
plt.plot(x, y2)
plt.plot(x, y1, color='red', linewidth=1.0, linestyle='--')
plt.xlim((-1, 2))
plt.ylim((-2, 3))
plt.xlabel('I am x')
plt.ylabel('I am y')
plt.show()
```



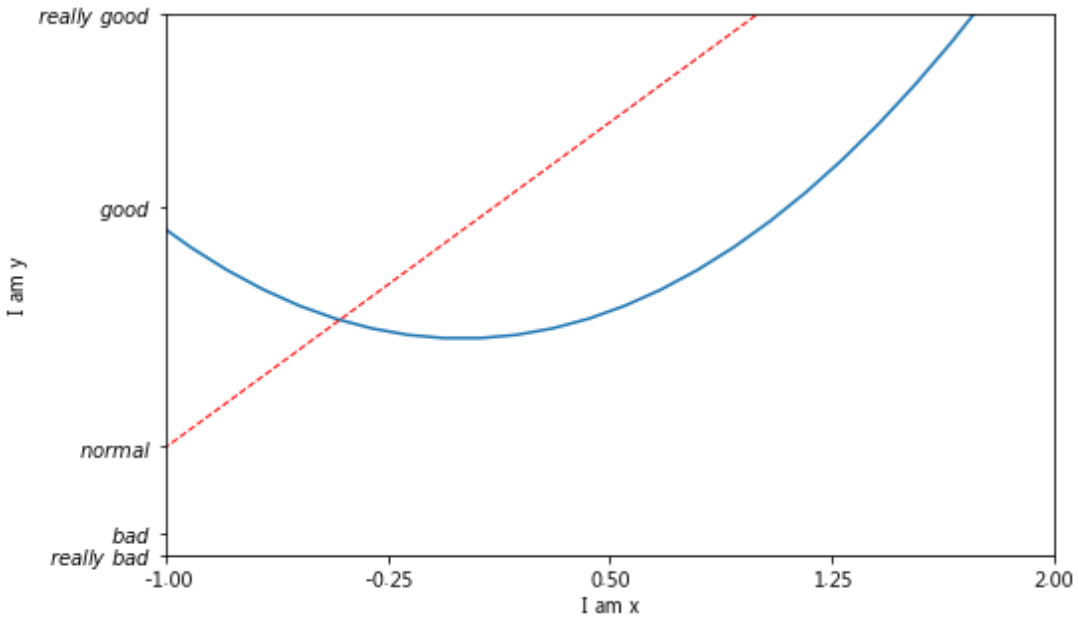
定义坐标轴刻度及名称

有时候，我们的坐标轴刻度可能并不是一连串的数字，而是一些文字，或者我们想要调整坐标轴的刻度的稀疏，这时，就需要使用plt.xticks()或者plt.yticks()来进行调整：首先，使用np.linspace定义新刻度范围以及个数：范围是(-1,2);个数是5。使用plt.xticks设置x轴刻度：范围是(-1,2);个数是5。使用plt.yticks设置y轴刻度以及名称：刻度为[-2, -1.8, -1, 1.22, 3]；对应刻度的名称为['really bad','bad','normal','good', 'really good']。使用plt.show()显示图像。

In [8]:

```
plt.figure(num=3, figsize=(8, 5))
plt.plot(x, y2)
plt.plot(x, y1, color='red', linewidth=1.0, linestyle='--')
plt.xlim((-1, 2))
plt.ylim((-2, 3))
plt.xlabel('I am x')
plt.ylabel('I am y')
new_ticks = np.linspace(-1, 2, 5)
print(new_ticks)
plt.xticks(new_ticks)
plt.yticks([-2, -1.8, -1, 1.22, 3],[r'$really\ bad$', r'$bad$', r'$normal$', r'$good$', r'$really\ good$'])
plt.show()
```

```
[-1.   -0.25  0.5   1.25  2.  ]
```

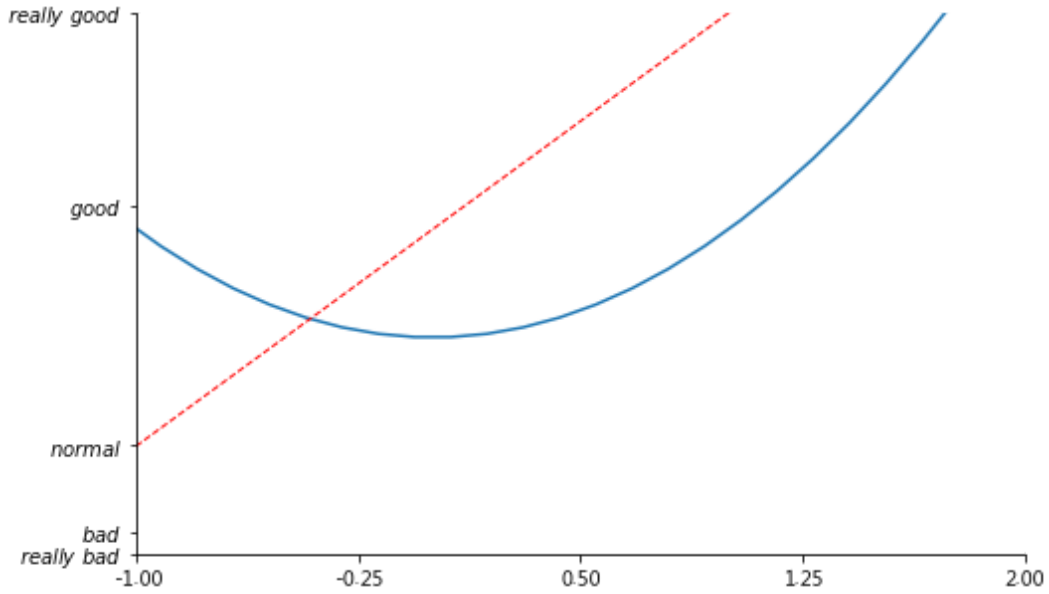


设置图像边框颜色

细心的小伙伴可能会注意到，我们的图像坐标轴总是由上下左右四条线组成，我们也可以对它们进行修改：首先，使用plt.gca()获取当前坐标轴信息。使用.spines设置边框：右侧边框；使用.set_color设置边框颜色：默认白色； 使用.spines设置边框：上边框；使用.set_color设置边框颜色：默认白色；

In [21]:

```
plt.figure(num=3, figsize=(8, 5))
plt.plot(x, y2)
plt.plot(x, y1, color='red', linewidth=1.0, linestyle='--')
plt.xlim((-1, 2))
plt.ylim((-2, 3))
plt.xticks(new_ticks)
plt.yticks([-2, -1.8, -1, 1.22, 3],[r'$really\ bad$', r'$bad$', r'$normal$', r'$good$', r'$really\ good$'])
ax = plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
plt.show()
```

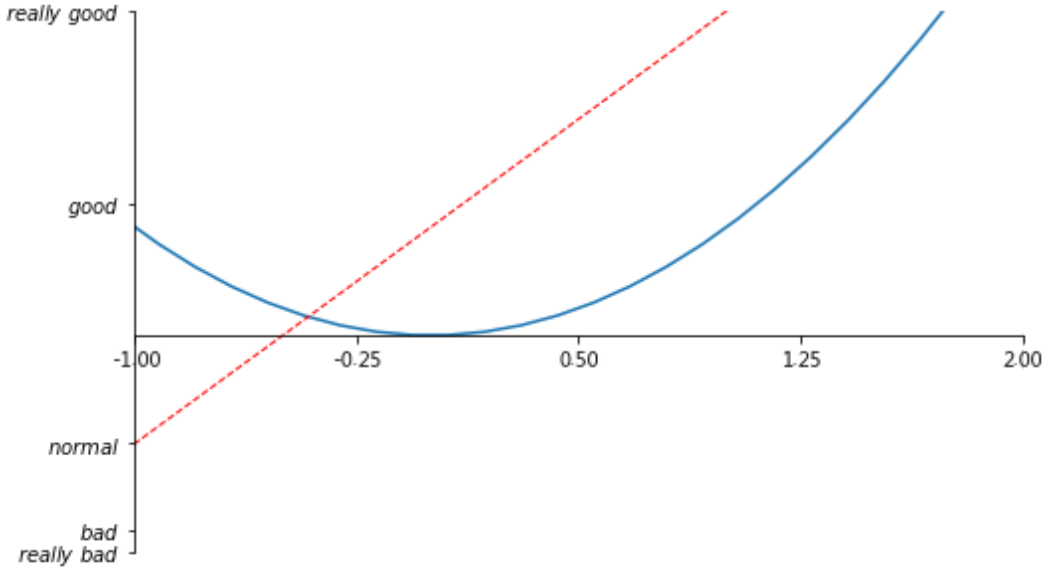


调整刻度及边框位置

使用.xaxis.set_ticks_position设置x坐标刻度数字或名称的位置：bottom.（所有位置：top，bottom，both，default，none）；使用.spines设置边框：x轴；使用.set_position设置边框位置：y=0的位置；（位置所有属性：outward，axes，data）

In [22]:

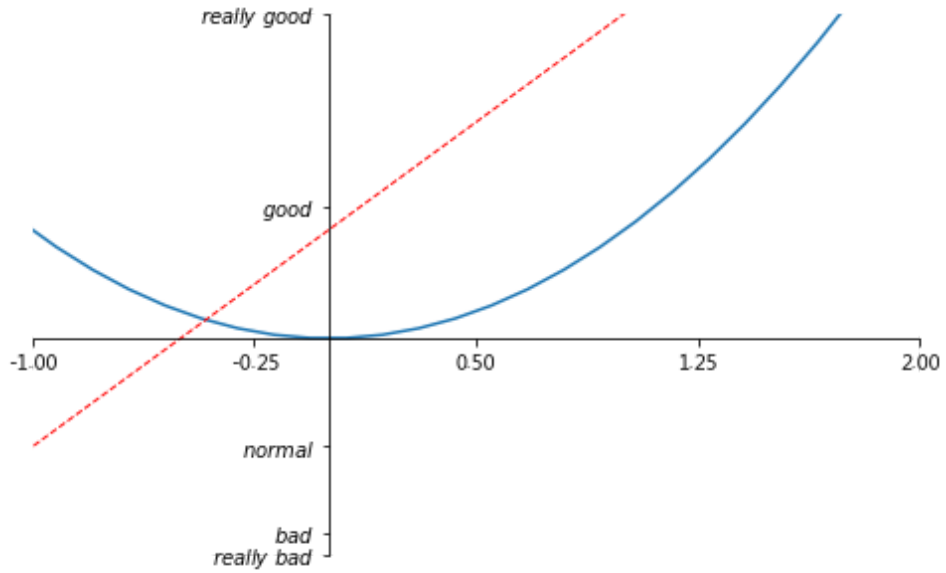
```
plt.figure(num=3, figsize=(8, 5))
plt.plot(x, y2)
plt.plot(x, y1, color='red', linewidth=1.0, linestyle='--')
plt.xlim((-1, 2))
plt.ylim((-2, 3))
plt.xticks(new_ticks)
plt.yticks([-2, -1.8, -1, 1.22, 3],[r'$really\ bad$', r'$bad$', r'$normal$', r'$good$', r'$really\ good$'])
ax = plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data', 0))
plt.show()
```



使用.yaxis.set_ticks_position设置y坐标刻度数字或名称的位置：left.（所有位置：left，right，both，default，none） 使用.spines设置边框：y轴；使用.set_position设置边框位置：x=0的位置；（位置所有属性：outward，axes，data） 使用plt.show显示图像.

In [24]:

```
plt.figure(num=3, figsize=(8, 5))
plt.plot(x, y2)
plt.plot(x, y1, color='red', linewidth=1.0, linestyle='--')
plt.xlim((-1, 2))
plt.ylim((-2, 3))
plt.xticks(new_ticks)
plt.yticks([-2, -1.8, -1, 1.22, 3],[r'$really\ bad$', r'$bad$', r'$normal$', r'$good$', r'$really\ good$'])
ax = plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data', 0))
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))
plt.show()
```



练一练

小伙伴们，以上就是matplotlib的基本用法，是不是比较简单呢？现在，请根据上述所学内容，画出直线 y = x-1, 线型为虚线，线宽为1，纵坐标范围（-2，1），横坐标范围（-1，2），横纵坐标在（0，0）坐标点相交。横坐标的 [-1,-0.5,1] 分别对应 [bad, normal, good]。请一定自己尝试一番再看下面的答案噢~

In [39]:

```
#答案
x = np.linspace(-1, 2, 50)
y = x - 1
plt.figure()
plt.plot(x,y, linewidth=1.0, linestyle='--')
plt.xlim((-1,2))
plt.ylim((-2,2))
plt.xticks([-1,-0.5,1],['bad', 'normal', 'good'])
ax = plt.gca()
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()
```