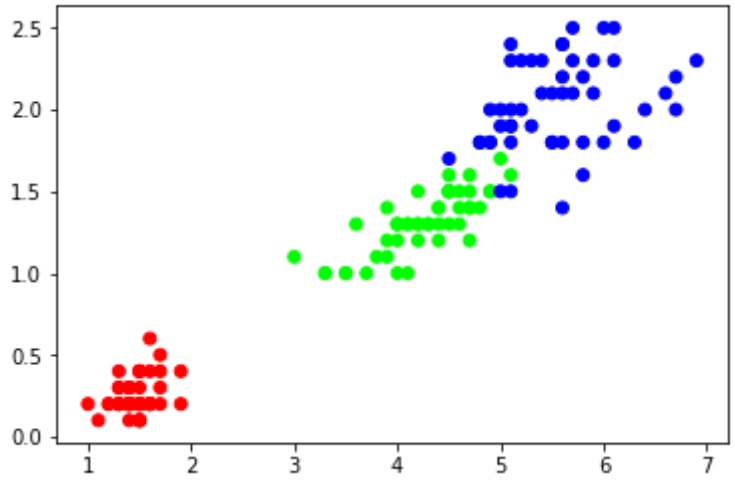
绘图

作者: 刘坤鑫

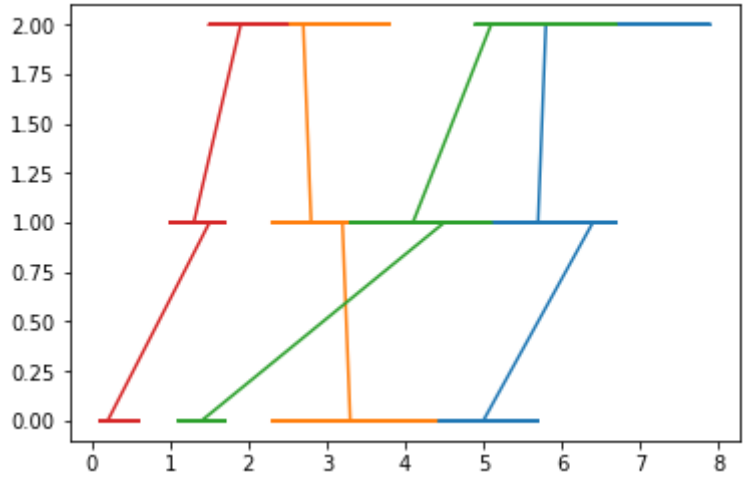
散点图

```
import  matplotlib.pyplot as plt
%matplotlib  inline
from  matplotlib.colors  import  ListedColormap #绘图引用的模块

cmap=ListedColormap(["#FF0000","#00FF00","#0000FF"])#颜色列表
plt.scatter(iris.data[:,2],iris.data[:,3],
            c=iris.target,cmap=cmap) #绘制散点图，根据颜色分类
```



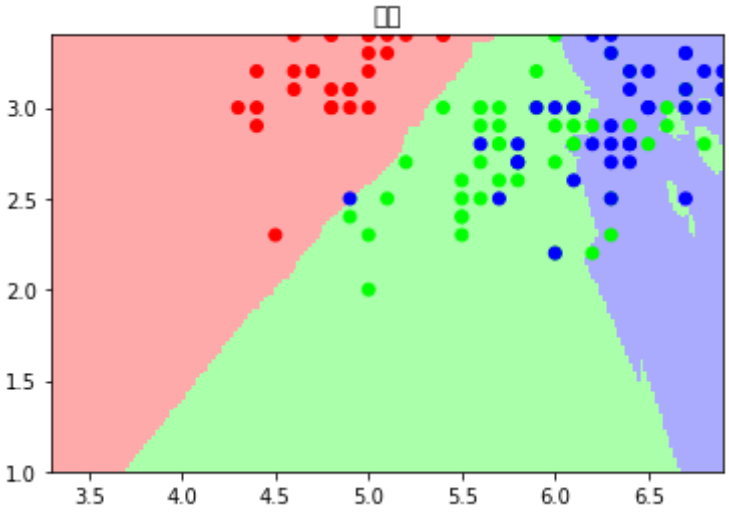
折线图:

```
plt.plot(x_test,y_test)
```

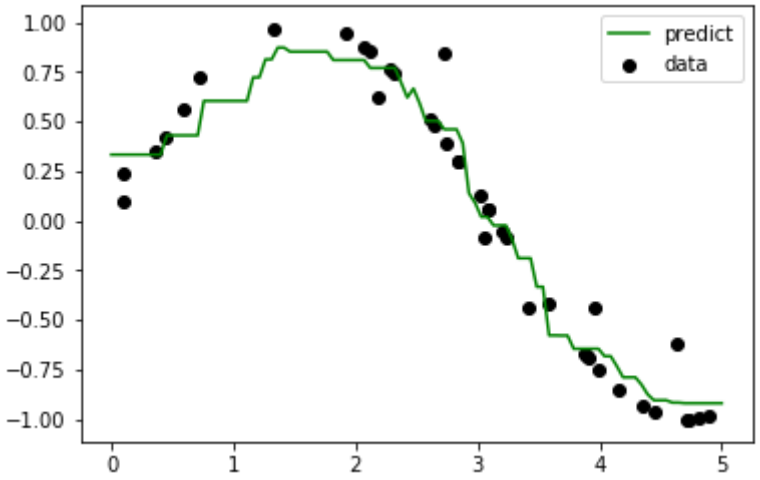



```
#两个颜色分类
cmap_light=ListedColormap(["#FFAAAA","#AAFFAA","#AAAAFF"])#颜色列
cmap_bold=ListedColormap(["#FF0000","#00FF00","#0000FF"])#颜色列
myknn=KNeighborsClassifier(n_neighbors=K) #设置访问周围15个点
myknn.fit(x,y) #训练数据
#四个数描述图片显示范围
xmin,xmax=x[:,0].min()-1,x[:,0].max()-1
ymin,ymax=x[:,1].min()-1,x[:,1].max()-1
#生成网格
```

```
h=0.02
xx,yy=np.meshgrid(np.arange(xmin,xmax,h),
                  np.arange(ymin,ymax,h))
#预测
z=myknn.predict(np.c_[xx.ravel(),yy.ravel()])
z=z.reshape(xx.shape)

#显示背景颜色
plt.pcolormesh(xx,yy,z,cmap=cmap_light)

#显示点的颜色
plt.scatter(x[:,0],x[:,1],c=y,cmap=cmap_bold)
plt.xlim(xx.min(),xx.max())
plt.ylim(yy.min(),yy.max())
plt.title("分类")
plt.show()
```

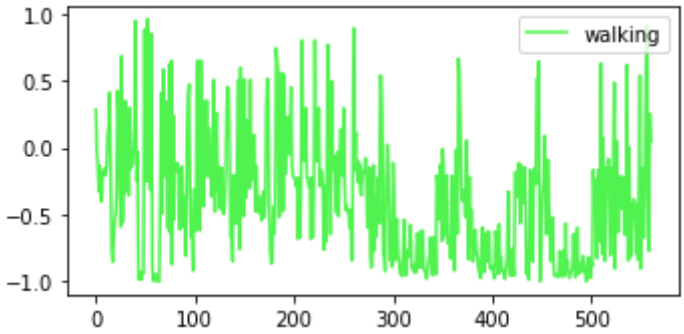


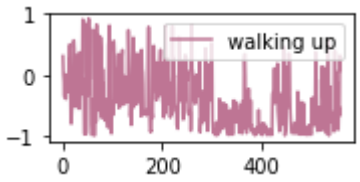

```
from sklearn.neighbors import KNeighborsRegressor #回归
T=np.linspace(0,5,100)[:,np.newaxis]
knn=KNeighborsRegressor(n_neighbors=5) #计算临近5个点

plt.scatter(x,y,c="k",label="data")
plt.plot(T,newy,c="g",label="predict")
plt.axis("tight")
plt.legend()
plt.show()
```

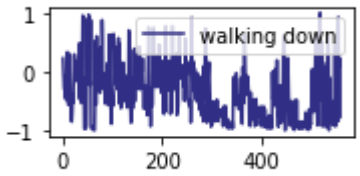


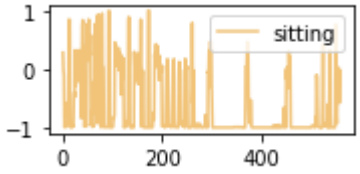

```
#  78,162,640,29,0,7260   分别对应6种行为抽样数据
x=[78,152,640,29,0,7260]
plt.figure(figsize=(12,9))
for i,r  in enumerate(x):
    plt.subplot(3,2,(i+1))
    #取出动作对应数据
```

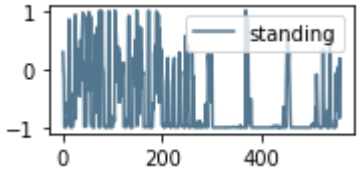
```
    data=x_train[r]
    color=np.random.rand(3)
    plt.plot(data,c=color,label=label[i+1])
    plt.legend()
    plt.show()
```

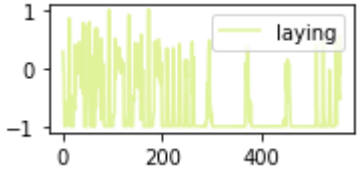




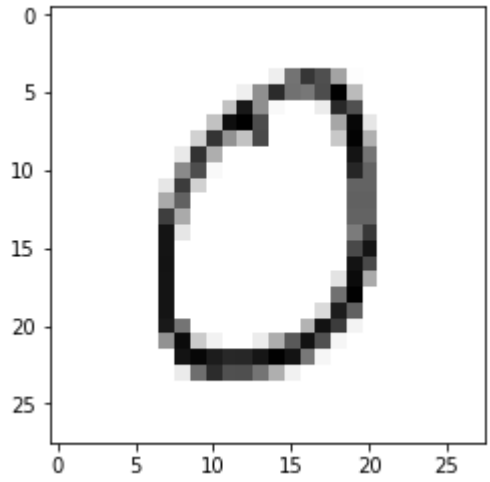




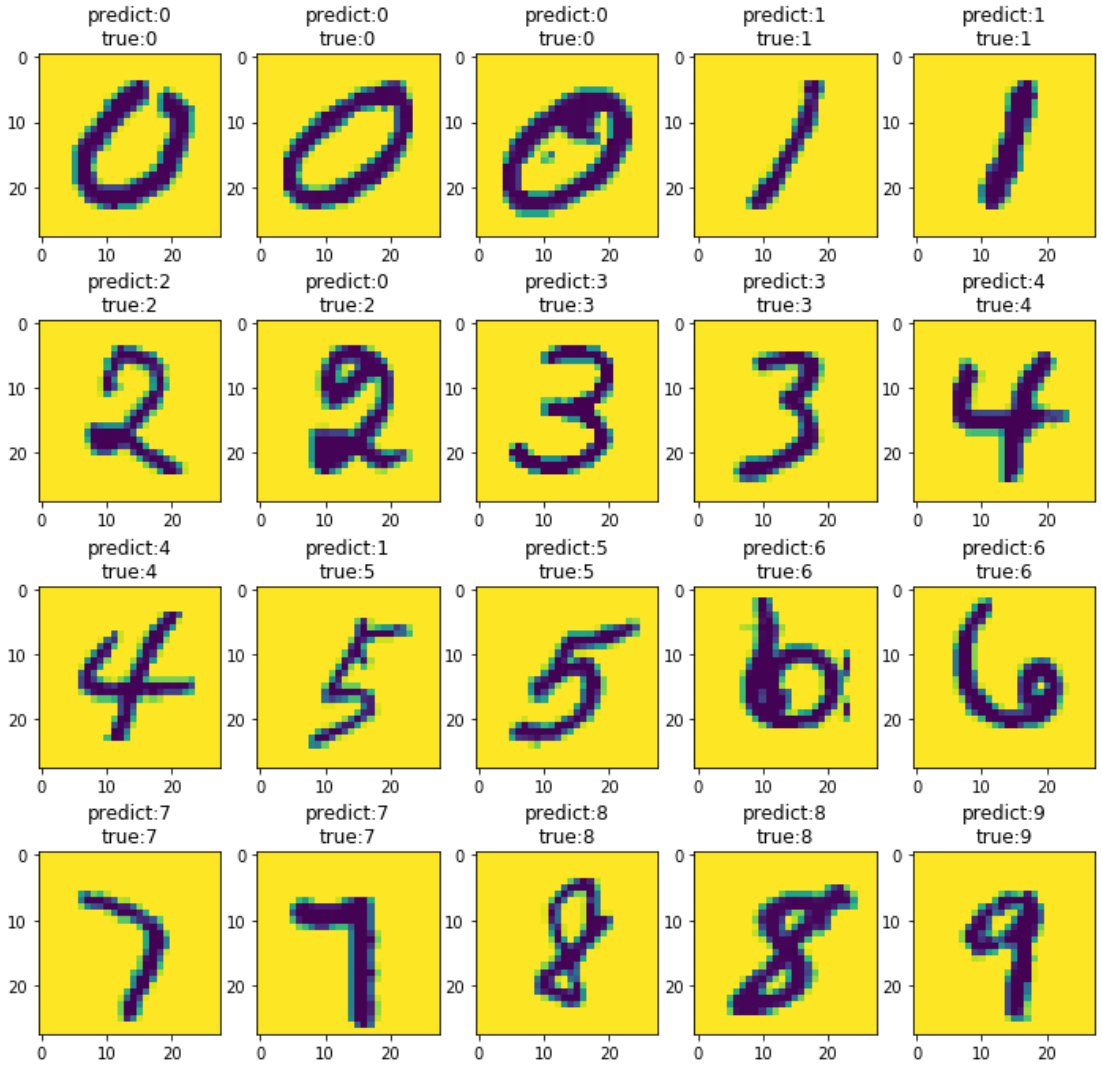





```
plt.imshow(zero1,cmap="gray")
```

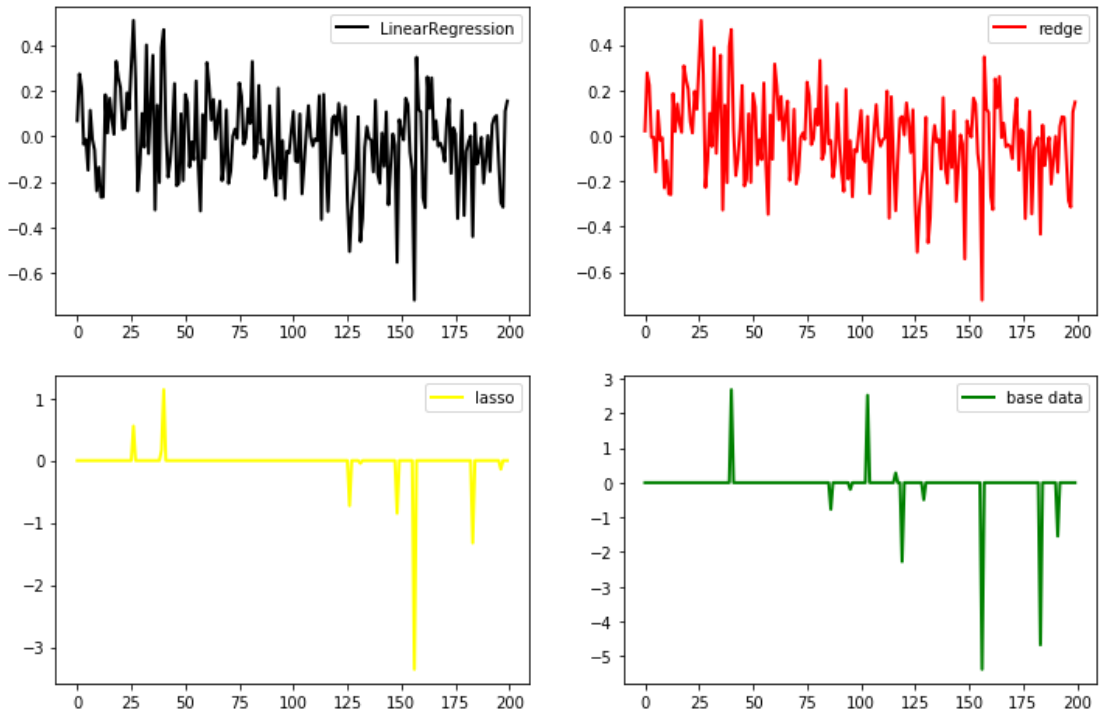


```
plt.figure(figsize=(12,15))
im_datas=X_test[::10]
im_target=Y_test[::10]
im_predict=Y_new[::10]
for  i  in range(20):
    plt.subplot(5,5,(i+1))
    plt.imshow(im_datas[i].reshape((28,28)))
    plt.title("predict:%d\n"%(im_predict[i])+"true:%d"%(im_target[i]))
```

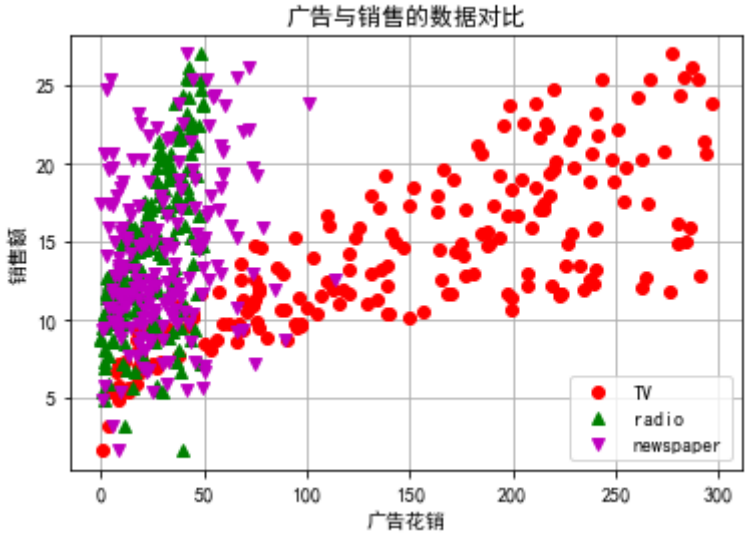



```
plt.figure(figsize=(12,8))
plt.subplot(2,2,1)
plt.plot(lnr.coef_,color="black",lw=2,
         label="LinearRegression")
```

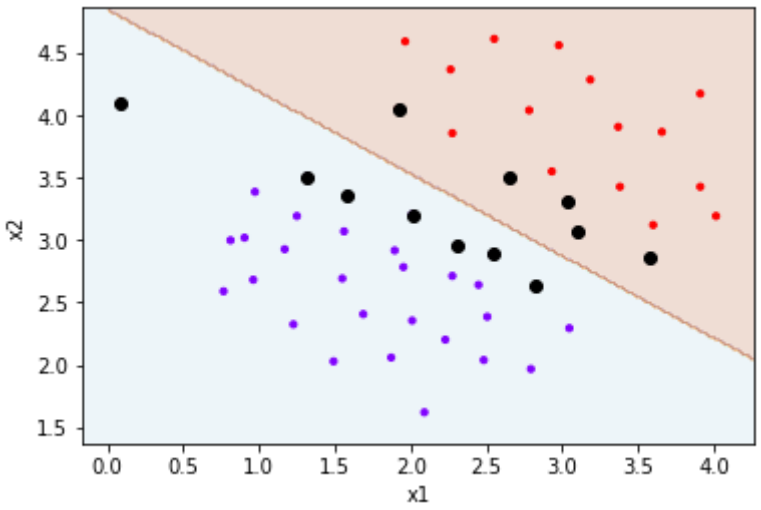
```
plt.legend()

plt.subplot(2,2,2)
plt.plot(redge.coef_,color="red",lw=2,
         label="redge")
plt.legend()

plt.subplot(2,2,3)
plt.plot(lasso.coef_,color="yellow",lw=2,
         label="lasso")
plt.legend()

plt.subplot(2,2,4)
plt.plot(coef,color="green",lw=2,
         label="base data")
plt.legend()

plt.show()
```




```
plt.figure(facecolor="w")
plt.plot(data["TV"],y,"ro",label="TV") #r 红色，o原点
plt.plot(data["Radio"],y,"g^",label="radio") #g绿色，^三角
plt.plot(data["Newspaper"],y,"mv",label="newspaper")
plt.xlabel("广告花销")
plt.ylabel("销售额")
plt.title("广告与销售的数据对比")
plt.grid()
plt.legend()
plt.show()
```




```
def  plot_svc(svc,x,y,h=0.02,pad=0.25):
    x_min,x_max=x[:,0].min()-pad,x[:,0].max()+pad #上限下限
    y_min,y_max=x[:,1].min()-pad,x[:,1].max()+pad #上限下限
    xx,yy=np.meshgrid(np.arange(x_min,x_max,h),  #矩阵表格
                      np.arange(y_min,y_max,h))
    Z=svc.predict(np.c_[xx.ravel(),yy.ravel()]) #预测结果
    Z=Z.reshape(xx.shape) #调整形状
    plt.contourf(xx,yy,Z,cmap=plt.cm.Paired,alpha=0.2)
    plt.scatter(x[:,0],x[:,1],s=10,c=y,cmap="rainbow")
    sv=svc.support_vectors_
    plt.scatter(sv[:,0],sv[:,1],c="k",linewidths="1")
    plt.xlim(x_min,x_max)
    plt.ylim(y_min,y_max)
    plt.xlabel("x1")
    plt.ylabel("x2")
    plt.show()
plot_svc(svc_linear,x,y)
```




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

# Fixing random state for reproducibility
np.random.seed(19680801)


def randrange(n, vmin, vmax):
    '''
    Helper function to make an array of random numbers having shape (n, )
    with each number distributed Uniform(vmin, vmax).
    '''
    return (vmax - vmin)*np.random.rand(n) + vmin
```

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

n = 100

# For each set of style and range settings, plot n random points in the box
# defined by x in [23, 32], y in [0, 100], z in [zlow, zhigh].
for c, m, zlow, zhigh in [('r', 'o', -50, -25), ('b', '^', -30, -5)]:
    xs = randrange(n, 23, 32)
    ys = randrange(n, 0, 100)
    zs = randrange(n, zlow, zhigh)
    ax.scatter(xs, ys, zs, c=c, marker=m)
plt.show()
```