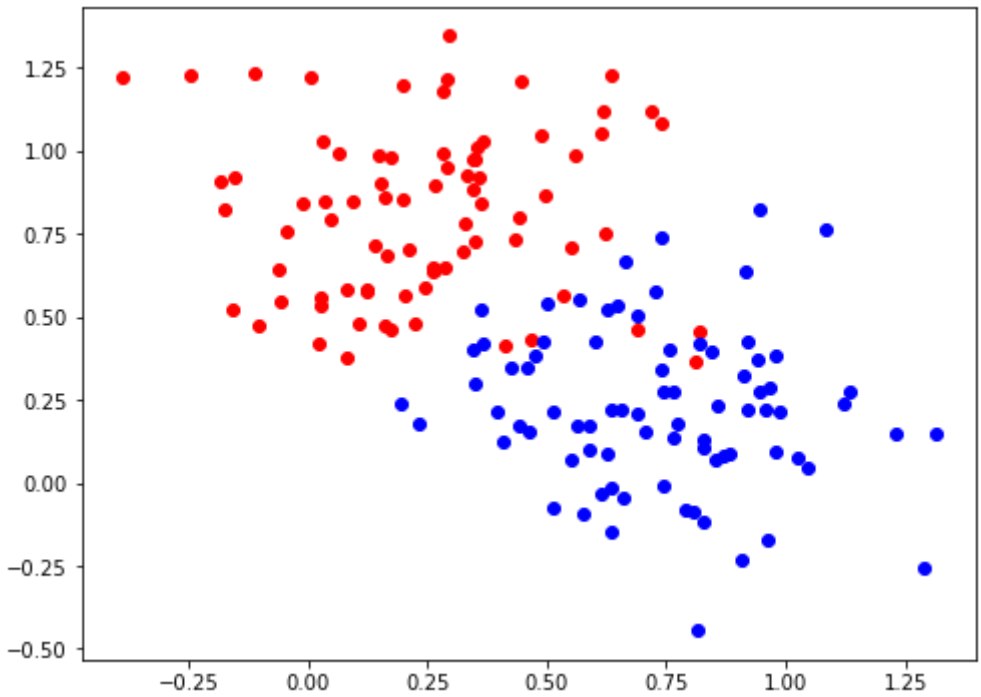
In [4]:

```
# -*- coding: utf-8 -*-
"""
Created on Sat May 29 14:20:23 2021

@author: nan25
"""

import numpy as np
import matplotlib.pyplot as plt

np.random.seed(0)
n=80
center=np.array([[0.25,0.75],[0.75,0.25]])
x0=np.random.normal(center[0,0],0.25,(n,1))
y0=np.random.normal(center[0,1],0.25,(n,1))
l0=np.zeros((n,1))
data0=np.concatenate((x0,y0,l0),axis=1)
x1=np.random.normal(center[1,0],0.25,(n,1))
y1=np.random.normal(center[1,1],0.25,(n,1))
l1=np.ones((n,1))
data1=np.concatenate((x1,y1,l1),axis=1)
data=np.concatenate((data0,data1),axis=0)
plt.figure(figsize=(8,6))
plt.scatter(data0[:,0], data0[:,1],color='r',marker='o')
plt.scatter(data1[:,0], data1[:,1],color='b',marker='o')
plt.show()
```



In [2]:

```
X0=np.concatenate((x0,y0),axis=1)
X1=np.concatenate((x1,y1),axis=1)
mu0=np.mean(X0,axis=0)  #miu 求平均值 在算每个类的中心点的位置 因为我要求w嘛所以我得先把Sw算出来#
mu1=np.mean(X1,axis=0)
Sw=(X0-mu0).T@(X0-mu0)+(X1-mu1).T@(X1-mu1)
#传说中的Sw 每一类的点关于自己类的中心点的距离的方差

w=np.linalg.inv(Sw)@(mu0-mu1).T #Sw的逆*（（miu0-miu1）.T）
#这边有转置是因为他自己之前定义mu0 mu1的时候没写转置 问题不大
print(w)
```

```
[-0.04825384  0.05749455]
```

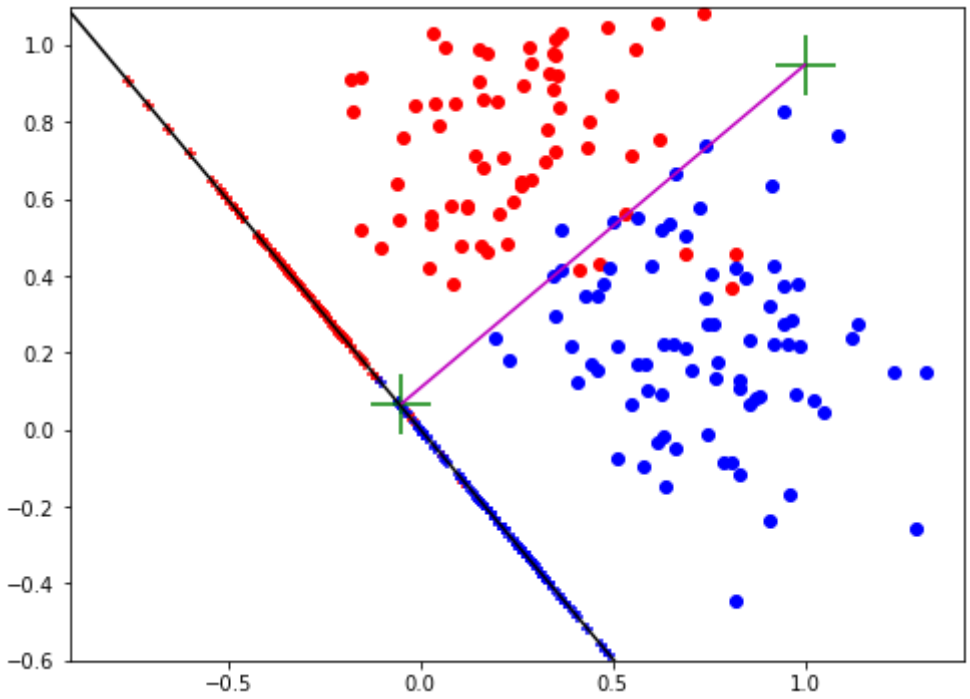
In [3]:

```
k=w[1]/w[0]      #w.T@x==0的斜率是-w[0]/w[1]
#所以这玩意儿是在算w.T@x的法线斜率

a0=(k*y0+x0)/(k*k+1) #红色点的投影点的横坐标 垂线和它的交点 自己推一下就可
b0=k*a0              #投影点的纵坐标
a1=(k*y1+x1)/(k*k+1) #蓝色同理
b1=k*a1
c0=np.mean(a0)
d0=np.mean(b0)
c1=np.mean(a1)
d1=np.mean(b1)
p=(c0+c1)/2
q=(d0+d1)/2
r=1
s=q-(r-p)/k

xx=np.linspace(-1,1,201) #黑线y=kx
yy=k*xx
plt.figure(figsize=(8,6))
plt.scatter(data0[:,0], data0[:,1],color='r',marker='o')
plt.scatter(data1[:,0], data1[:,1],color='b',marker='o')
plt.scatter(a0, b0,color='r',marker='+') #画投影点
plt.scatter(a1, b1,color='b',marker='+')

plt.scatter(r, s,s=900,color='g',marker='+')
plt.scatter(p, q,s=900,color='g',marker='+')
plt.plot([p,r],[q,s],color='m') #中间那条线 由（p,q）和（r,s）两点连接的线段
#（r,s）是右上角那个点
plt.plot(xx,yy,'k-')
plt.axis("equal")  #控制坐标轴单位长度一样 这样看起来比较像垂直
plt.xlim([-0.5,1])
plt.ylim([-0.6,1.1])
plt.show()
```



In [ ]: