In [47]:
```
import numpy as np
def pontos_2d(w,P):
    np.random.seed(30)
    # Gera um conjunto de pontos aleatorios no plano
    X = [np.ones(P),np.random.random_sample(P,),
         np.random.random_sample((P,))]
    X = np.array(X)
    Y = []
    # w representa os coeficientes de uma reta escolhida ao acaso.
    # Esta reta separa os pontos em dois conjuntos, acima e abaixo da reta.
    for i in range(P):
        if X[2][i]*w[2] + X[1][i]*w[1] + X[0][i]* w[0] < 0:
                Y.append(1)
        else:
                Y.append(0)
    return X,Y
```

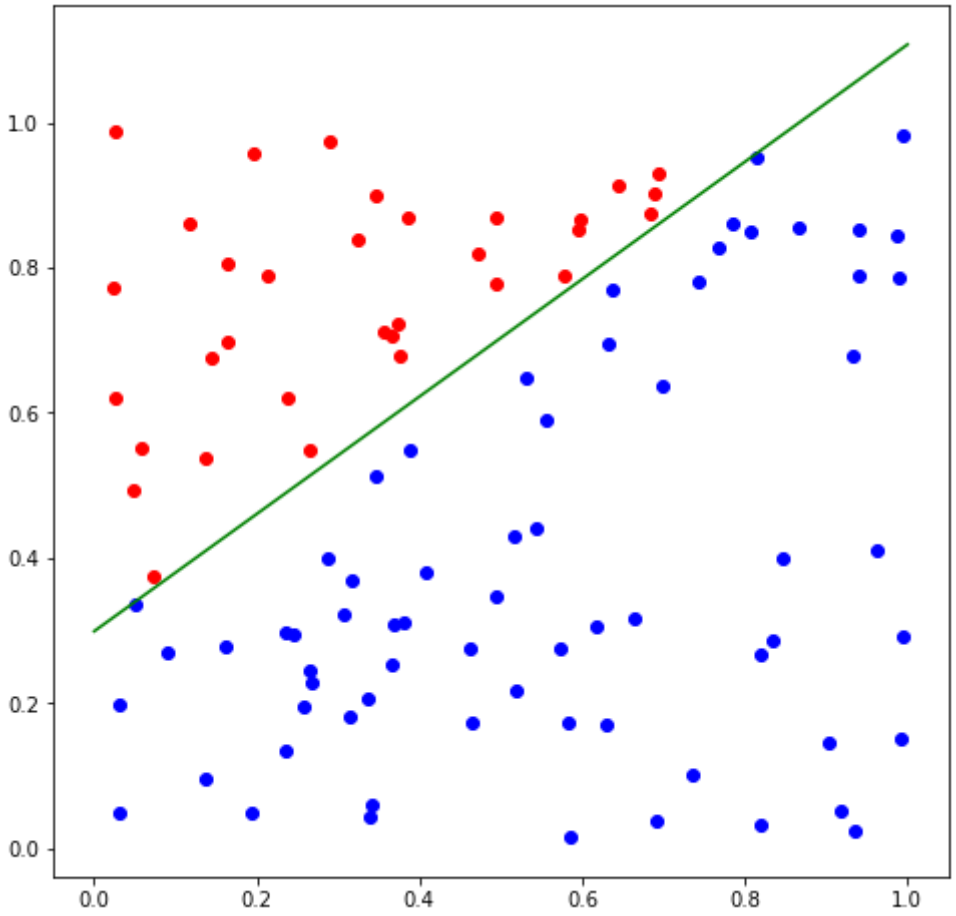
In [48]:
```
def sigma(w,x):
    return 1/(1+np.exp(-np.dot(x,w)))

# Funcao Custo da regressao Logistica
def g_lr(w,x,y):
    P = len(x[0,:])
    N = len(x[:,0])
    cost = 0
    for p in range(P):
        cost = cost + (sigma(w,x[:,p])-y[p])**2  # 1-y[p] pata SVM
    cost = cost/P
    return cost

# Gradiente da funcao custo
def grad_lr(w,x,y):
    P = len(x[0,:])
    N = len(x[:,0])
    grad = np.zeros(N)
    for p in range(P):
        k = (sigma(w,x[:,p])-y[p])*(sigma(w,x[:,p])**2)*np.exp(-np.dot(x[:,p
        grad = grad + k * x[:,p]
    grad = grad/P
    return 2*grad

# Gradiente Descendente
def gradient_descent(w,x,y,alpha,max_its):
    for k in range(max_its):
        w = w - alpha * grad_lr(w,x,y)
        cost = g_lr(w,x,y)
        print('cost = ',cost)
    return w
```

In [49]:
```
import matplotlib.pyplot as plt

# Numero de pontos da amostra
P = 100

# Reta escolhida para classificar os pontos iniciais
w = np.array([-0.3,-0.8,1])

# Geracao dos pontos
[X,Y] = pontos_2d(w,P)
```

```
N = 3
w = np.ones(N)

# Maximo de iteracoes
max_its = 100

# Tamanho do passo
alpha = 10

# Aplica o gradiente descendente
w = gradient_descent(w,X,Y,alpha,max_its)

# Plota os pontos e a reta obtida pela regressao
plt.figure(figsize = (8,8))
for i in range(P):
    if Y[i] > 0:
        plt.scatter(X[1,i],X[2,i],c='b')
    else:
        plt.scatter(X[1,i],X[2,i],c='r')
x = np.linspace(0,1,100)
y = - (w[1]*x+w[0])/w[2]
plt.plot(x,y,'g')
plt.show()
```

```
cost =  0.24381843780897824
cost =  0.2102605230219109
cost =  0.1914468340065291
cost =  0.17600883387983732
cost =  0.16319333123242832
cost =  0.15248922475736862
cost =  0.14351617525299193
cost =  0.13589808472061635
cost =  0.12937301284663602
cost =  0.12371633472062687
cost =  0.11876877607660674
cost =  0.11439831267689632
cost =  0.11050759992046749
cost =  0.10701636182353451
cost =  0.10386302986780395
cost =  0.10099675284518113
cost =  0.09837718756187572
cost =  0.09597084749028871
cost =  0.09375041364609839
cost =  0.0916930064448803
cost =  0.08977953542127697
cost =  0.08799382747483374
cost =  0.0863221469543933
cost =  0.08475271746671821
cost =  0.0832753963221399
cost =  0.08188138958168956
cost =  0.08056303559487236
cost =  0.07931362397715361
cost =  0.07812725068432241
cost =  0.07699869787227721
cost =  0.07592333518450409
cost =  0.07489703752339123
cost =  0.07391611644380948
cost =  0.07297726249894054
cost =  0.07207749659117202
cost =  0.07121412870847037
cost =  0.0703847227723076
cost =  0.06958706655983321
cost =  0.06881914586224708
cost =  0.06807912219486403
cost =  0.06736531349837355
cost =  0.06667617736970706
cost =  0.06601029644079426
cost =  0.06536636558811332
cost =  0.06474318070852765
cost =  0.06413962883987581
cost =  0.06355467944004696
cost =  0.06298737666734912
cost =  0.062436832529025994
cost =  0.0619022207847654
cost =  0.06138277150870217
cost =  0.06087776622736185
cost =  0.06038653356270192
cost =  0.05990844531927217
cost =  0.05944291296285686 4
cost =  0.05898938444503373
cost =  0.05854734133410039
cost =  0.0581162962179509
cost =  0.057695790348875814
cost =  0.057285391504026076
cost =  0.05688469203851964
cost =  0.056493307110966305
cost =  0.05611087306359889
cost =  0.05573704594129587
```

```
cost =  0.055371500135596385
cost =  0.05501392714139491
cost =  0.054664034415382104
cost =  0.054321544326509465
cost =  0.05398619318981313
cost =  0.053657730375862175
cost =  0.05333591748891581
cost =  0.05302052760759464
cost =  0.05271134458250886
cost =  0.05240816238584928
cost =  0.05211078450844723
cost =  0.051819023400253406
cost =  0.051532699950578784
cost =  0.05125164300479434
cost =  0.05097568891449785
cost =  0.05070468111843793
cost =  0.050438469751734376
cost =  0.0501769112811613
cost =  0.04991986816445789
cost =  0.04966720853181575
cost =  0.04941880588785349
cost =  0.0491745388325367
cost =  0.04893429079963323
cost =  0.04869794981141569
cost =  0.04846540824842822
cost =  0.04823656263323615
cost =  0.04801131342716348
cost =  0.04778956483910557
cost =  0.047571224645577004
cost =  0.04735620402122181
cost =  0.04714441737907344
cost =  0.04693578221990839
cost =  0.04673021899008671
cost =  0.04652765094731996
cost =  0.046328004033848284
cost =  0.04613120675654837
```



In [ ]:

In [ ]: