The function `showdata` displays the data points (the u_i and the v_j) and the function `showSVMs2` displays the separating line and the two margin lines.

```
function  showSVMs2(w,b,eta,ll,mm,nw)
%
%  Function to display the result of running SVM
%  on p blue points u_1, ..., u_p in u
%  and q red points v_1, ..., v_q in v

l = makeline(w,b,ll,mm,nw);        %  makes separating line
lm1 = makeline(w,b+eta,ll,mm,nw);  %  makes blue margin line
lm2 = makeline(w,b-eta,ll,mm,nw);  %  makes red margin line

plot(l(1,:),l(2,:),'-m','LineWidth',1.2)            %  plots separating line
plot(lm1(1,:),lm1(2,:),'-b','LineWidth',1.2)        %  plots blue margin line
plot(lm2(1,:),lm2(2,:),'-r','LineWidth',1.2)        %  plots red margin line
hold off
end
```

Actually, implementing the above function is not entirely trivial. It is necessary to write a function `makeline` to plot the line segment which is part of the line of equation $w_1 x + w_2 y = b$ inside a box containing the data points. We leave the details an exercises.

B.2 Soft Margin SVM (SVM$_{s2'}$)

The following `Matlab` programs implement the method described in Section 54.8.

The function `doSVMs2pbv3` calls the function `solve1` given in Section 52.7.

```
function [lamb,mu,alpha,beta,lambnz,munz,numsvl1,numsvm1,badnu,w,nw,b,eta]
    = doSVMs2pbv3(nu,rho,u,v,K)
%
%   Best version
%   Uses the duality gap to compute eta
%   In principle, needs a single support vector of type 1
%
%   Soft margin nu-SVM version s2'
%   with the constraint
%   \sum_{i = 1}^p + \sum_{j = 1}^q mu_j  = K_m
%   (without the variable gamma)
%
%   p green vectors u_1, ..., u_p in n x p array u
%   q red   vectors v_1, ..., v_q in n x q array v
```