T-61.6040 Assignment-08/2011

Santosh Tirunagari (245577)
santosh.tirunagari@aalto.fi

December 18, 2011

AIM

Implementation group Lasso MKL algorithm discussed on Iris data using p=1 and p=2 as parameters.

$$k_{GAU} = (x_i, x_j) = exp(-\|x_i x_j\|_2^2/s^2),\ C = 1,\ and\ s = 1. \quad (1)$$

Datasets

The datasets given are Iris training and test data with 4 features Training set consisting of 90 and test set consisting of 60 samples each.

Implementation

Load and zscore normalize both the train and test data using zscore matlab command. Zscore normalization can also be done using the method described in the lecture. As the distribution is same the statistics of the training set and the test set would be same.

```
   X = zscore(dlmread('iris_train_X.txt'));
X_t = zscore(dlmread('iris_test_X.txt'));
Y = (dlmread('iris_train_y.txt'));
Y_t = (dlmread('iris_test_y.txt'));
```

Create a label list for one versus all aproach method. The true values are given -1s for one class and the rest are denoted as 1s

```
y1 = ones(90,1);
y2 = ones(90,1);
y3 = ones(90,1);
y1t = ones(60,1);
```

```
y2t = ones(60,1);
y3t = ones(60,1);
y1(1:30,1) =-1;
y2(31:60,1) =-1;
y3(61:90,1) =-1;
y1t(1:20,1) =-1;
y2t(21:40,1) =-1;
y3t(41:60,1) =-1;
```

Calculate kernel function for each feature.

```
for i = 1:1:4
  KF{i} = kro(X(:,i),X(:,i));

end
```

Calculate Keta value

```
P = 1;
eta = ones(1,4)./4;
ceta = [];
x = 1/(P+1);
y = (P/(P+1));
C =1;

Keta = zeros(90,90);
for k = 1:1:4
    Keta = Keta + eta(k) .* KF{k};
end
yv = {y1 y2 y3};
yvt = {y1t y2t y3t};
```

Solve SVM optimization problem in standard QP form using quadprog function in matlab and find alpha values.

```
H = (yv{i}*yv{i}').*Keta;
        A = [];
        Aeq = yv{i}';
        l = zeros(90,1);
        c = -1*ones(90,1);
        b = [];
        beq = 0;
        u = 1*ones(90, 1);
        warning off;
        options = optimset('Algorithm','interior-point-convex','MaxIter',1500);
        % solve quadprog to find alpha values
        alpha = quadprog(H, c, A, b, Aeq, beq, l, u,[],options);
        % calculate near boundary coefficients

        alpha(alpha < C * 0.001) = 0;
        alpha(alpha > C*0.999) = C;
```

Calculate the new weight norm and update eta values till the threshold or the maximum iterations till 100.

```
iter = 0;
obj = 1000;
old = 500;
    while  abs(old - obj) > 0.001 && iter < 100;
           old = obj;
%----------------> calculate alpha values using above code ----------------> %

 cw = zeros(4,1);
        for it = 1:1:4
        cw(it,1) = eta(it).^2 * ((alpha.*yv{i})'*KF{it})*(alpha.*yv{i});
        end

        eta = (cw.^x)./((sum((cw.^y))).^(1/P));
        Keta = zeros(90,90);
        for k = 1:1:4
        Keta = Keta + eta(k) .* KF{k};
        end
        iter = iter+1;
        obj = objective(Keta,KF,yv{i},alpha);
```

Calculate Keta values and solve svm problem for finding alpha values

```
   Q = yv{i}*yv{i}';
Keta = zeros(90,90);
for k = 1:1:4
    Keta = Keta + eta(k) .* KF{k};
end
H = Q.*Keta;
A = [];
Aeq = yv{i}';
l = zeros(90,1);
c = -1*ones(90,1);
b = [];
beq = 0;
u = 1*ones(90, 1);
options = optimset('Algorithm','interior-point-convex','MaxIter',1500);
% solve quadprog to find alpha values
alpha = quadprog(H, c, A, b, Aeq, beq, l, u,[],options);
```

Compute the near boundary coefficients and move near boundary alpha values to boundaries

```
    alpha(alpha < C * 0.001) = 0;
    alpha(alpha > C*0.999) = C;
```

Compute value for bias parameter b

```
    sv = find(alpha >0 & alpha<C(k));
```

```
    sv_one = zeros(200,1);
    sv_one(sv,1) = 1;
   b = sv_one'*(yv{i}-((alpha.*yv{i})'*Keta'))'/sum(sv_one)
```

Compute the decision function on training set and calculate the confusion matrix

```
  temp = bsxfun(@plus,Keta(sv,:)'*(alpha(sv,:).*yv{i}(sv,:)),b);
res =temp;
res(res>=0) = 1;
res(res<0) = -1;
conf{i} = confusionmat(yv{i},res);
```

Similarly compute the Keta value for the test data

```
for k = 1:1:4
  KFt1{k} = kro(X_t(:,k),X(sv,k));

end
Ketat1 = zeros(length(sv),60);
for k = 1:1:4
    Ketat1 = Ketat1 + eta(k) .* KFt1{k};
end
```

Compute the decision function on test set and calculate the confusion matrix.

```
   temp = bsxfun(@plus,Ketat1'*(alpha(sv,:).*yv{i}(sv,:)),b);
rest = temp;
% thresholding
rest(rest>=0) = 1;
rest(rest<0) = -1;
conft{i} = confusionmat(yvt{i},rest);
```

Threshold the decisions to get proper $+1$ and -1

```
    res(res>=0) = 1;
    res(res<0) = -1;
```

Plot the bar graphs showing the eta values.

```
    bar(eta);
xlabel('Kernel');
ylabel('Weight');
axis([1 4 0.0 1.0])
title([num2str((i)), ' versus others(p=1)' ]);
  %--print figure--
  fnout = ['p1_',num2str((i)), '.jpg'];
  print('-djpeg','-r150',fnout);
```

function for calculating the objective function

```
 function val = objective(Keta,KF,yv,alpha)
    val1 = zeros(4,1);
```

```
    for x = 1:1:4
        val1(x) = (alpha .* yv)'*KF{x}*(alpha .* yv);
    end
    val1 = max(val1);
    val2 = (alpha .* yv)'*Keta*(alpha .* yv);
    val = (abs(val1 - val2));
end
```

Similarly compute the same steps for parameter P =2; Running of code for this implementation in matlab: ex8_p1 and ex8_p2 respectively.

Results

Tables below show the confusion matrices computed.

Table 1: Confusion matrix for 1 vs rest for training set p=1

30	0
0	60

Table 2: Confusion matrix for 2 vs rest for training set p=1

28	2
2	58

Table 3: Confusion matrix for 3 vs rest for training set p=1

28	2
2	58

Table 4: Confusion matrix for 1 vs rest for test set p=1

20	0
0	40

Table 5: Confusion matrix for 2 vs rest for test set p=1

20	0
2	38

Table 6: Confusion matrix for 3 vs rest for test set p=1

18	2
0	40

Figure 1 shows the bar plot of eta values generated for p=1 and Figure 2 shows the bar plot of eta values generated for p=2.

Table 7: Confusion matrix for 1 vs rest for training set p=1

30	0
0	60

Table 8: Confusion matrix for 2 vs rest for training set p=1

28	2
2	58

Table 9: Confusion matrix for 3 vs rest for training set p=1

28	2
2	58

Table 10: Confusion matrix for 1 vs rest for test set p=1

20	0
0	40

Table 11: Confusion matrix for 2 vs rest for test set p=1

20	0
2	38

References

[2] http://en.wikipedia.org/wiki/Support_vector_machine
[3] http://www.mathworks.se/help/toolbox/optim/ug/quadprog.html

Table 12: Confusion matrix for 3 vs rest for test set p=1

18	2
0	40

Figure 1: Bar plot of eta values utilized in implementing kernel with p =1.

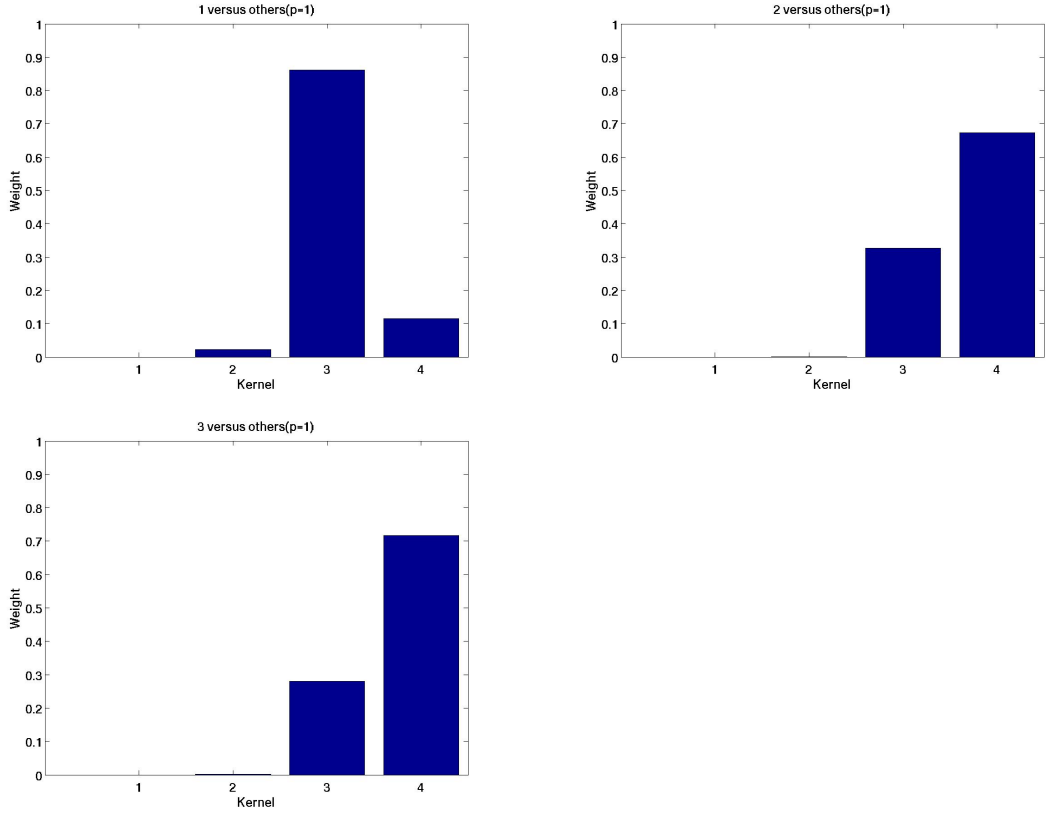

Table 13: Confusion matrix for 1 vs rest for training set p=2

30	0
0	60

Table 14: Confusion matrix for 2 vs rest for training set p=2

28	2
2	58

Figure 2: Bar plot of eta values utilized in implementing kernel with p = 2.

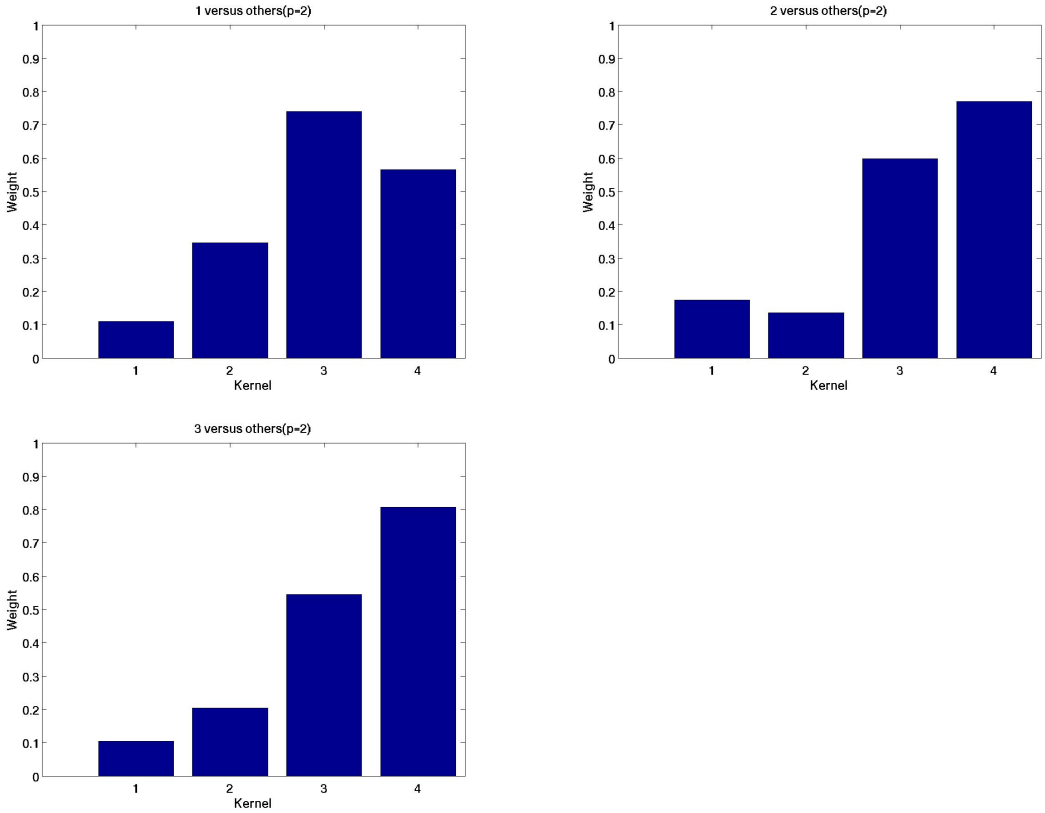

Table 15: Confusion matrix for 3 vs rest for training set p=2

28	2
2	58

Table 16: Confusion matrix for 1 vs rest for test set p=2

20	0
0	40

Table 17: Confusion matrix for 2 vs rest for test set p=2

20	0
2	38

Table 18: Confusion matrix for 3 vs rest for test set p=2

18	2
0	40

Attachments

Matlab code ex8_p1.m,ex8_p2 and datasets have been archived along with this report.