Contents

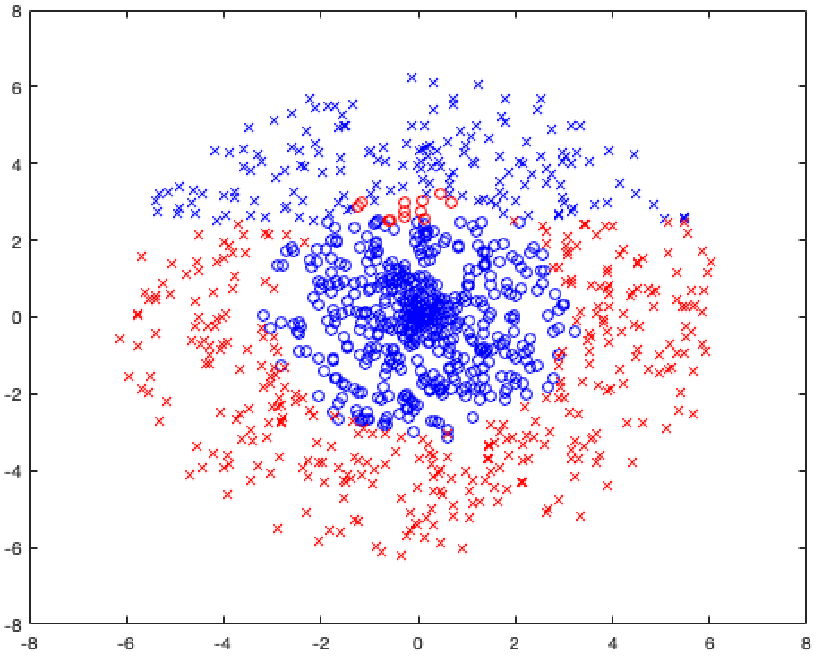
3.3 generate training data from 2 point clouds

```
%random generation for each run!
[X, t] = make_cloud();
```

3.4 weak learner performance

```
%learn parameters from data
param = weaklearn(X,t);

%make prediction using parameters
tp = weakeval(X,param);

%plotting:
figure('Name', 'weak learner');

%correct classified type A points
idx = (t < 0) & (t == tp);
plot(X(1,idx), X(2,idx), 'bx'); hold on;

%correct classified type B points
idx = (t > 0) & (t == tp);
plot(X(1,idx), X(2,idx), 'bo'); hold on;

%incorrect classified type A points
idx = (t < 0) & (t ~= tp);
plot(X(1,idx), X(2,idx), 'rx'); hold on;

%incorrect classified type B points
idx = (t > 0) & (t ~= tp);
plot(X(1,idx), X(2,idx), 'ro'); hold off;

disp(['A single weak learner gets ' num2str(100*sum(t == tp)/numel(tp)) ' % correct']);

%observation: a value of around 60-70% for a weak learner is to be expected
%since the pattern in the data set is circular with type B (circle) data in
%the center wrapped around type A (cross) points. So when the weak learner
%is trying to optimize a vertical/horizontal cut for correct classification,
%it will never be able to capture the decision boundary well which we expect is
%a ring surrounding the circular data points.
```

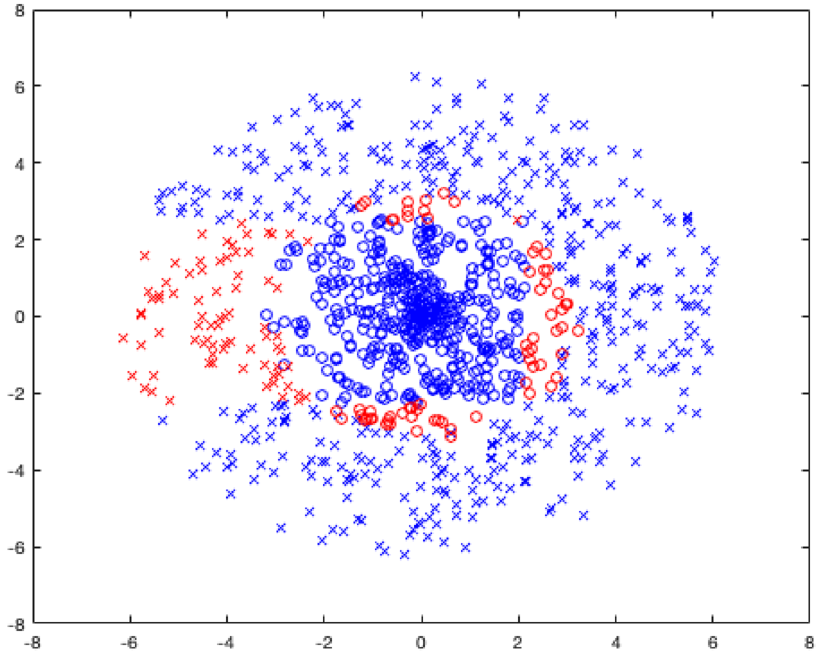
```
A single weak learner gets 65.1 % correct
```



3.5 M = 5 adaptive boosting

```
M = 5;

[params, weight] = boostlearn(X, t, M);
tp = boosteval(X, params, weight);

%same plotting as previously
figure('Name', 'M = 5 adaboost classifier');
idx = (t < 0) & (t == tp);
plot(X(1,idx), X(2,idx), 'bx'); hold on;
idx = (t > 0) & (t == tp);
plot(X(1,idx), X(2,idx), 'bo'); hold on;
idx = (t < 0) & (t ~= tp);
plot(X(1,idx), X(2,idx), 'rx'); hold on;
idx = (t > 0) & (t ~= tp);
plot(X(1,idx), X(2,idx), 'ro'); hold off;

disp(['A M = 5 adaboost classifier gets ' num2str(100*sum(t == tp)/numel(tp)) ' % correct']);
```

```
A M = 5 adaboost classifier gets 85.6 % correct
```

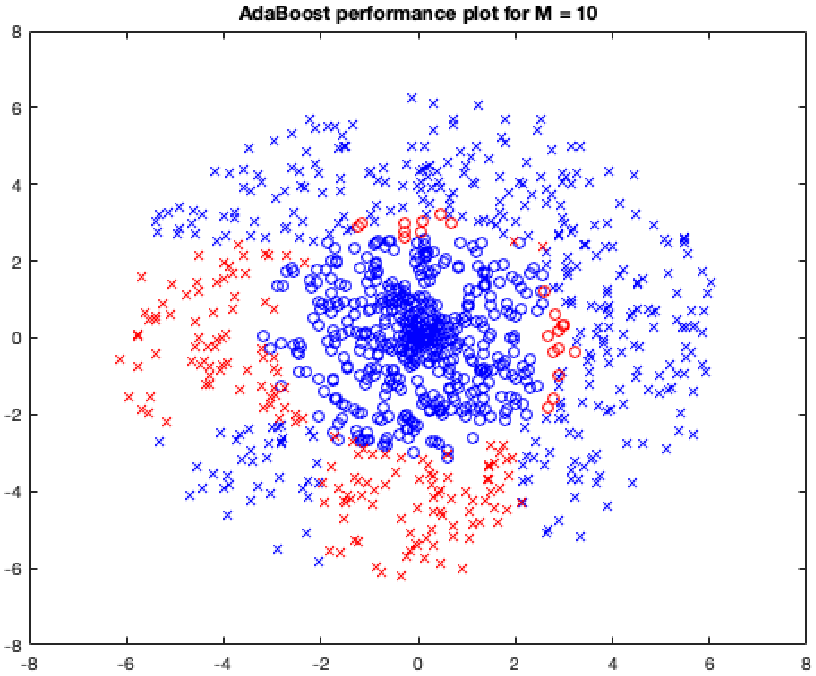


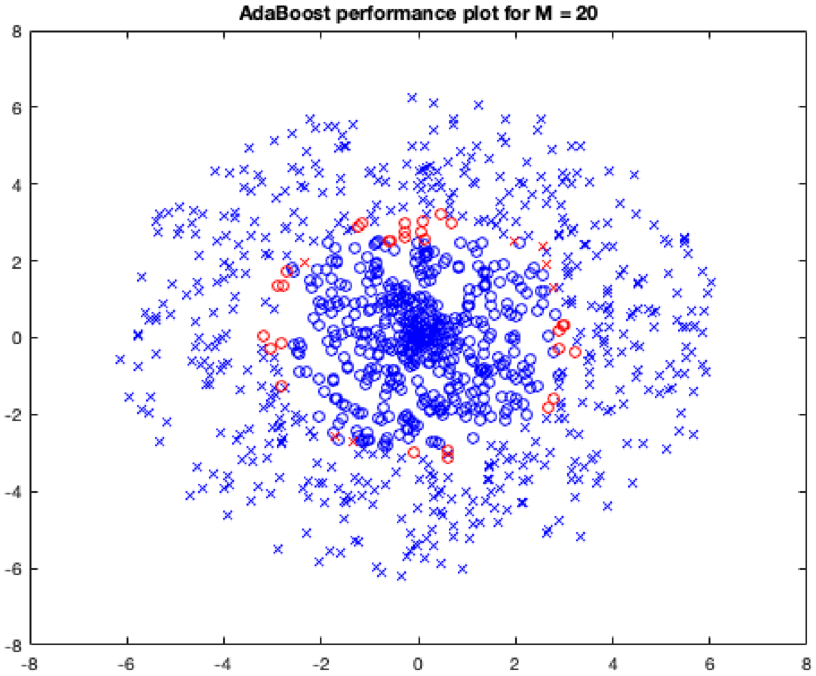
3.6 test larger values of M on training data

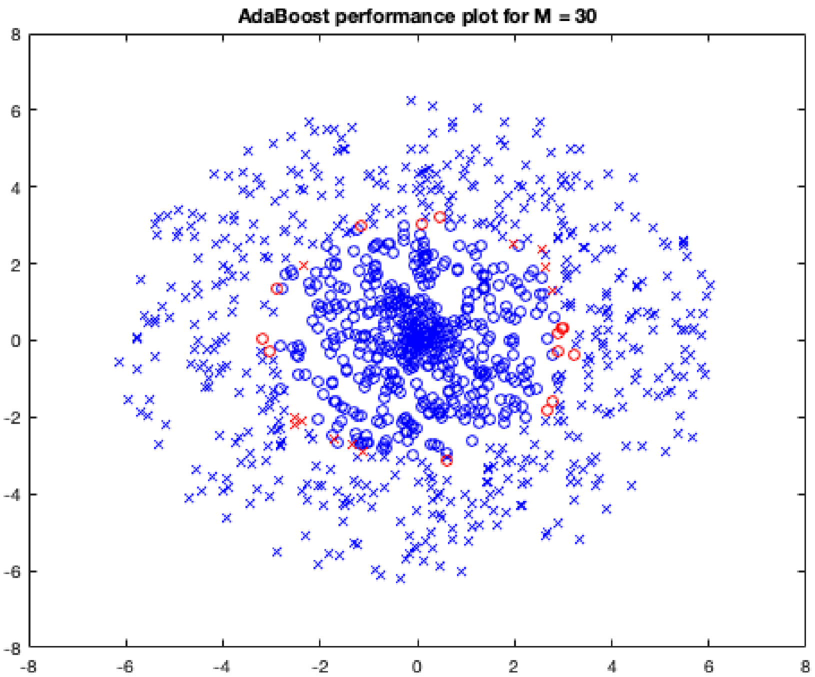
```
for M = 10:10:100
    %learn AdaBoost parameters given number of learners M
    [params, weight] = boostlearn(X, t, M);
    tp = boosteval(X, params, weight);

    %plot training result for each M
    figure;
    idx = (t < 0) & (t == tp);
    plot(X(1,idx), X(2,idx), 'bx'); hold on;
    idx = (t > 0) & (t == tp);
    plot(X(1,idx), X(2,idx), 'bo'); hold on;
    idx = (t < 0) & (t ~= tp);
    plot(X(1,idx), X(2,idx), 'rx'); hold on;
    idx = (t > 0) & (t ~= tp);
    plot(X(1,idx), X(2,idx), 'ro'); hold off;
    title(['AdaBoost performance plot for M = ' num2str(M)]);

    disp(['A M  = ' num2str(M) ' adaboost classifier gets ' num2str(100*sum(t == tp)/numel(tp)) ' % correct']);
end

%observation: Judging from the resulting scatterplots of the data based on
%their types ('x' vs 'o') and if they are classified correctly (red vs blue),
%we notice that while M = 10 still recognizes a strip incorrectly, for M = 20
%the aggregate scheme has already learned the circular pattern of data very well
%to even sometimes outperform M = 30. This means that overfitting most likely
%happens in the M = 20 to M = 30 range.

%In other words, M = 20 looks like the point at which the adaboost classifier
%is sufficiently represents the circular pattern of data such that increasing
%M further will mean overfitting (fitting noise specific to the sample)
```

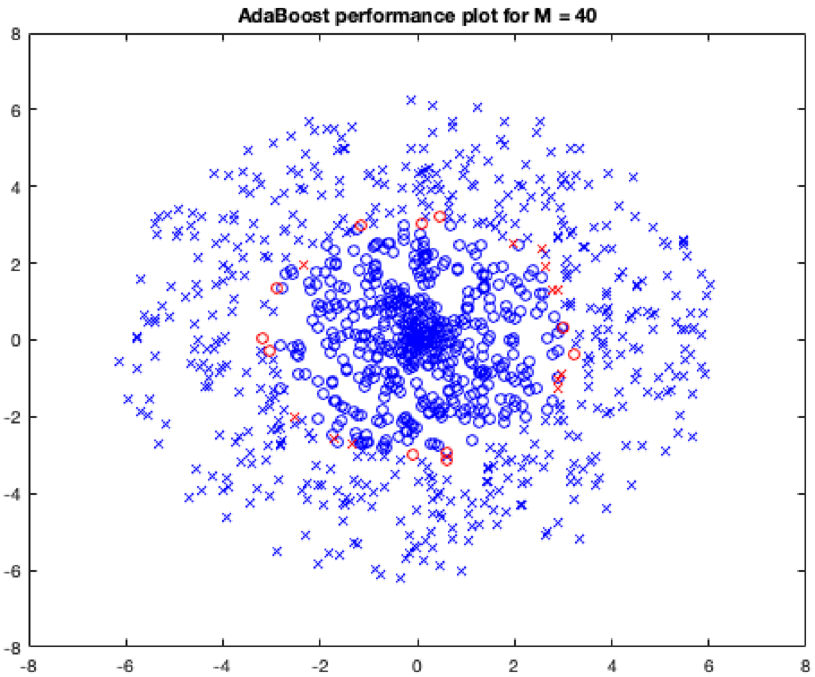
```
A M  = 10 adaboost classifier gets 81.6 % correct
A M  = 20 adaboost classifier gets 96.4 % correct
A M  = 30 adaboost classifier gets 97.5 % correct
A M  = 40 adaboost classifier gets 97.7 % correct
A M  = 50 adaboost classifier gets 97.7 % correct
A M  = 60 adaboost classifier gets 97.5 % correct
```

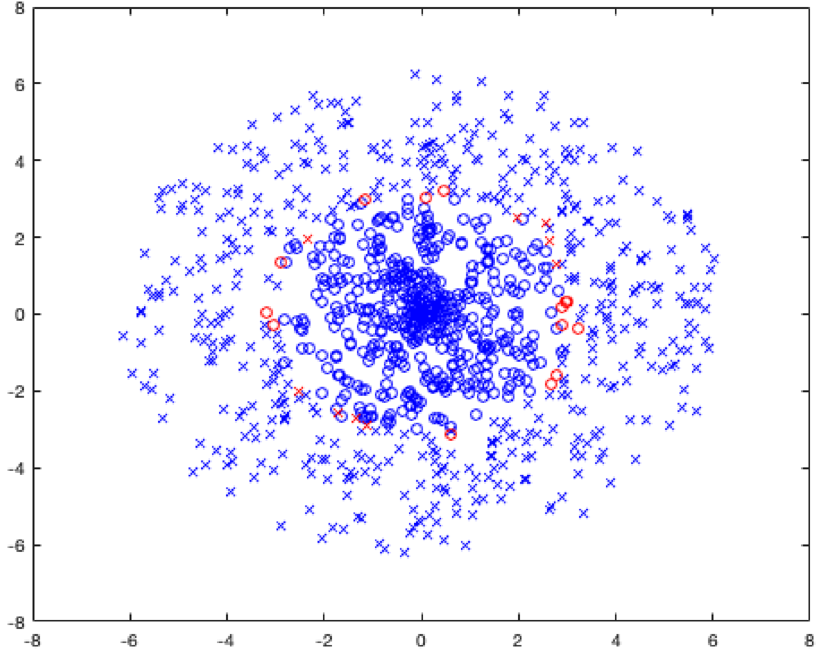
```
A M  = 70 adaboost classifier gets 97.4 % correct
A M  = 80 adaboost classifier gets 98.2 % correct
A M  = 90 adaboost classifier gets 98.4 % correct
A M  = 100 adaboost classifier gets 98.3 % correct
```

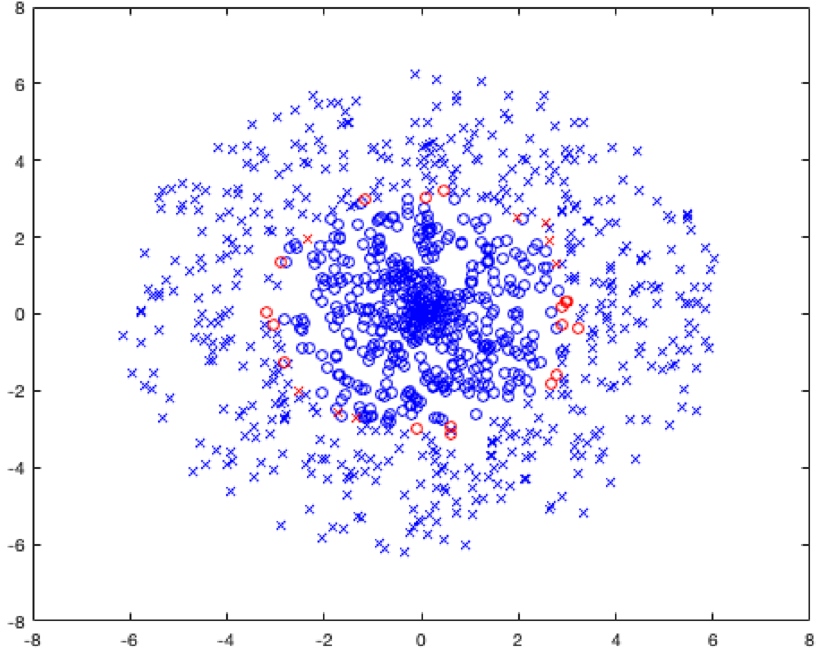


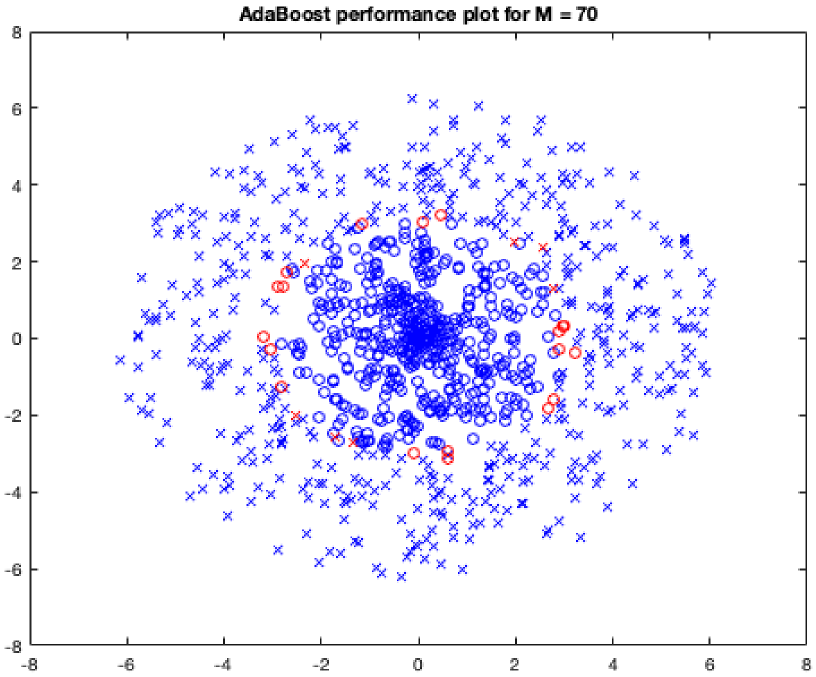


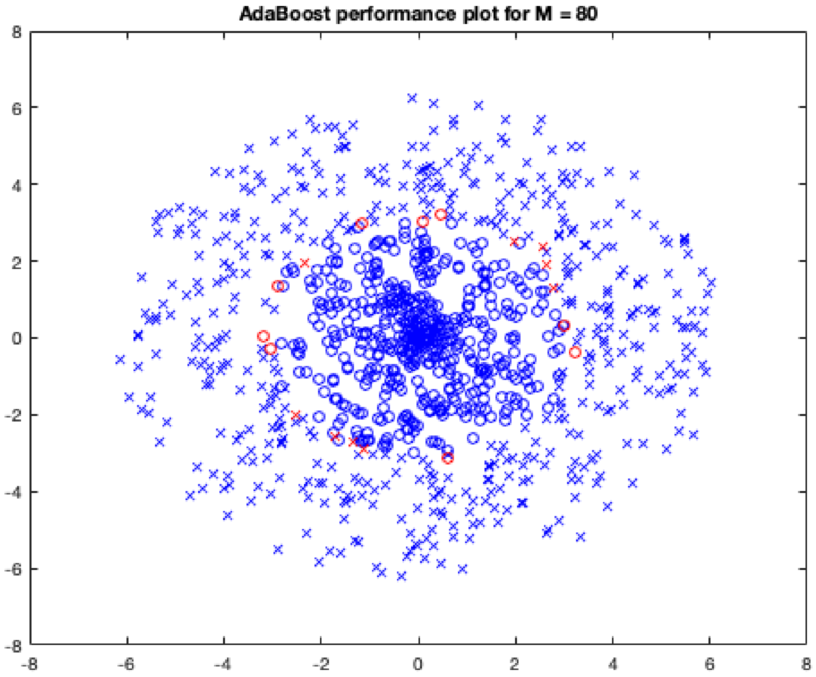

AdaBoost performance plot for M = 30
8
6
4
2
0
-2
-4
-6
-8
-8
-6
-4
-2
0
2
4
6
8

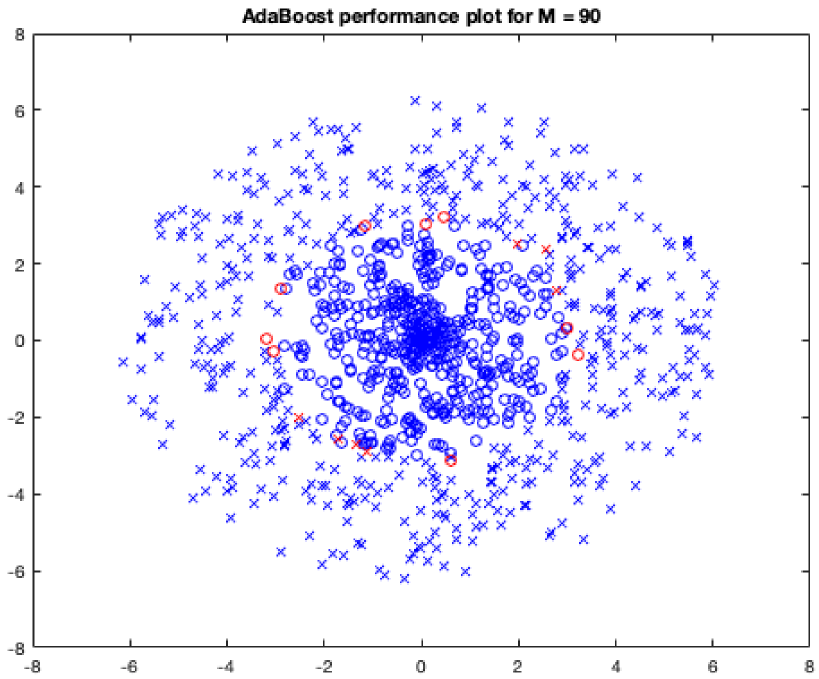

AdaBoost performance plot for M = 40
8
6
4
2
0
-2
-4
-6
-8
-8
-6
-4
-2
0
2
4
6
8

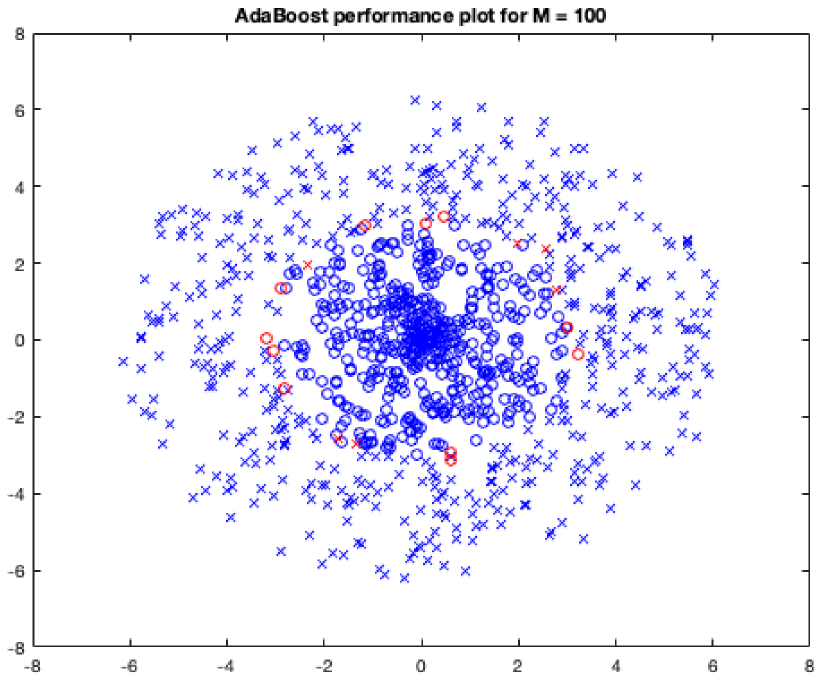

AdaBoost performance plot for M = 50


AdaBoost performance plot for M = 60


AdaBoost performance plot for M = 70
8
6
4
2
0
-2
-4
-6
-8
-8
-6
-4
-2
0
2
4
6
8

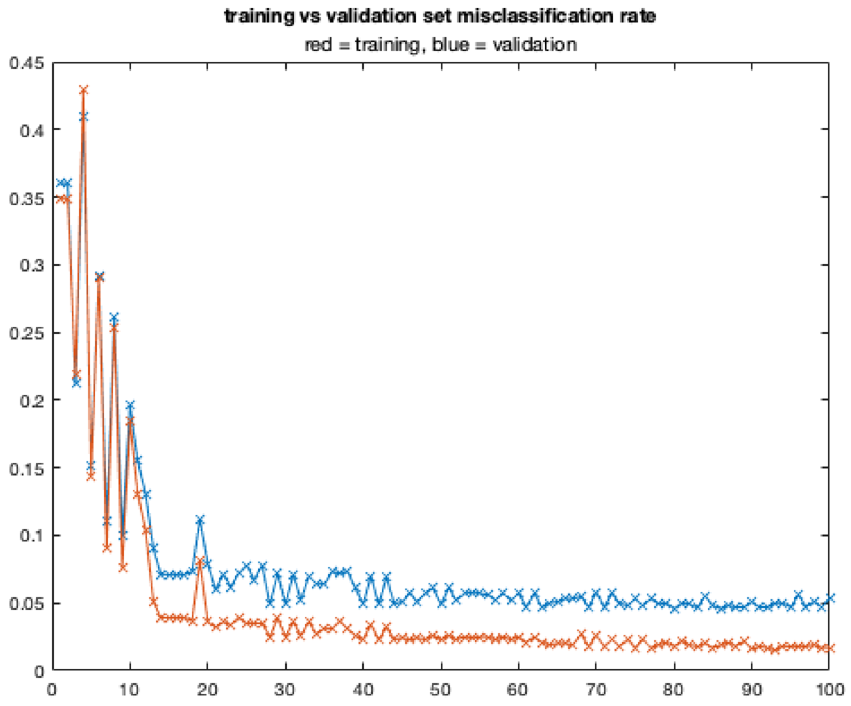

AdaBoost performance plot for M = 80
8
6
4
2
0
-2
-4
-6
-8
-8
-6
-4
-2
0
2
4
6
8





3.7 test accuracy on validation set, with M = 1:100

```
%generate validation data
[vX, vt] = make_cloud();

t_acc_rate = zeros(100,1);
v_acc_rate = zeros(100,1);

for M = 1:100
    %learn w parameters, alpha using original point cloud
    [params, weight] = boostlearn(X, t, M);
    %compute performance for both training and validation sets
    t_tp = boosteval(X, params, weight);
    v_tp = boosteval(vX, params, weight);

    %store training/validation performance acc in 2 distinct arrays
    t_acc_rate(M) = sum(t == t_tp)/numel(t_tp);
    v_acc_rate(M) = sum(vt == v_tp)/numel(v_tp);
end

figure;
plot(1:100,1-v_acc_rate,'-x'); hold on;

plot(1:100,1-t_acc_rate,'-x'); hold off;
title('training vs validation set misclassification rate', 'red = training, blue = validation');

%Observation: we see that both the graph for training and validation set
%misclassification rate follow the same shape as they all decrease sharply
%for small M but very quickly the rate of decrease stops to 0 when M = 20.
%Beyond M = 20, both graphs stays flat but fluctuate in the same way with the
%training misclassification decreasing very slightly, but validation set
%misclassification rate is clearly stuck at around 0.05 for large M. This is
%due to the subsequent learners after M = 20 mostly just fitting noise.
```



Published with MATLAB® R2020b