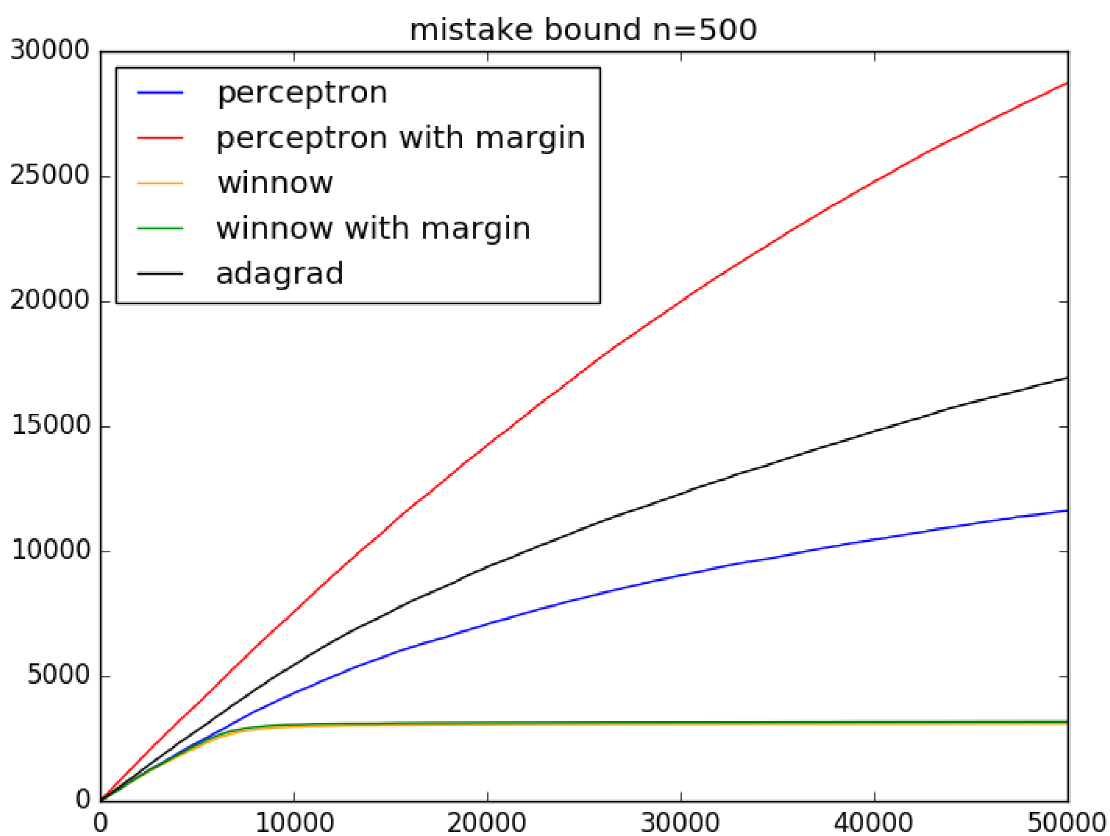
CS446: Machine Learning	**Fall 2016**

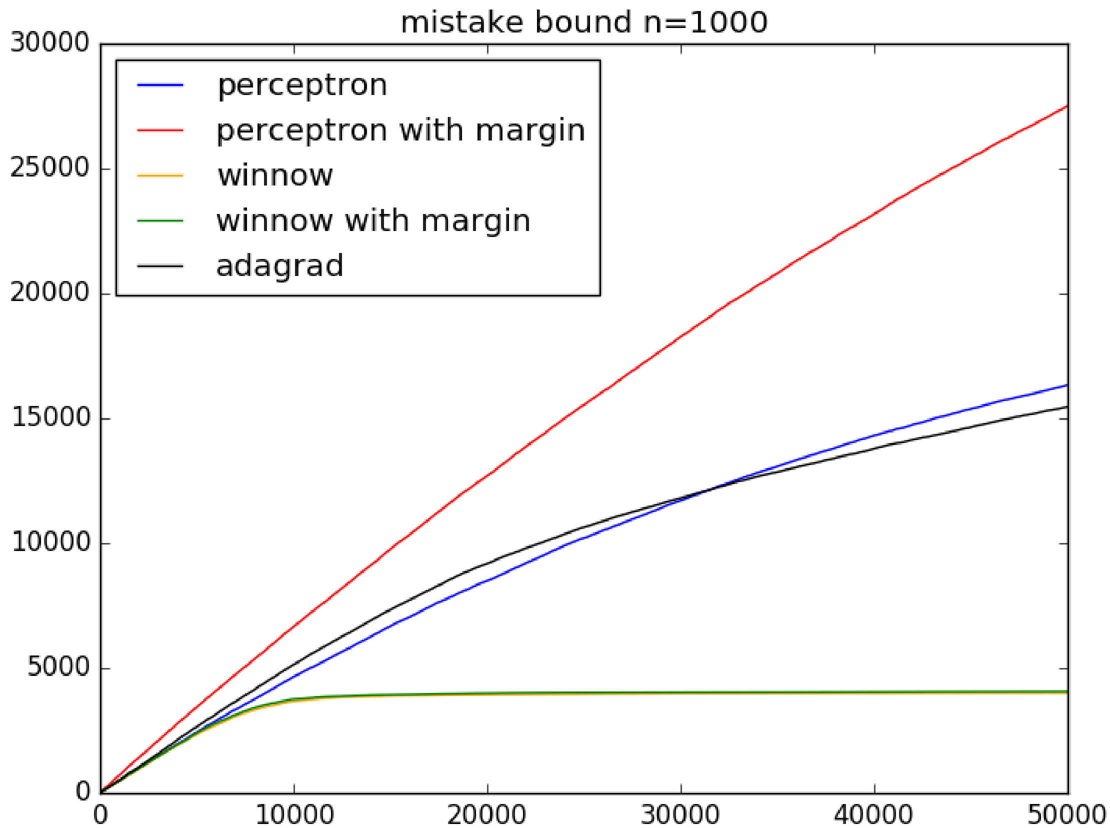
Problem Set 3

Yihao Zhang yzhng127 *Handed In: October 8, 2016*

1. Answer to problem 1

Algorithm	parameters	Dataset n = 500	Dataset n = 1000
Perceptron	NO	NO	NO
Perceptron w/ margin	η	0.005	0.001
Winnow	α	1.1	1.1
Winnow w/ margin	α, γ	1.1 ,2.0	1.1, 2.0
AdaGrad	η	0.25	0.25

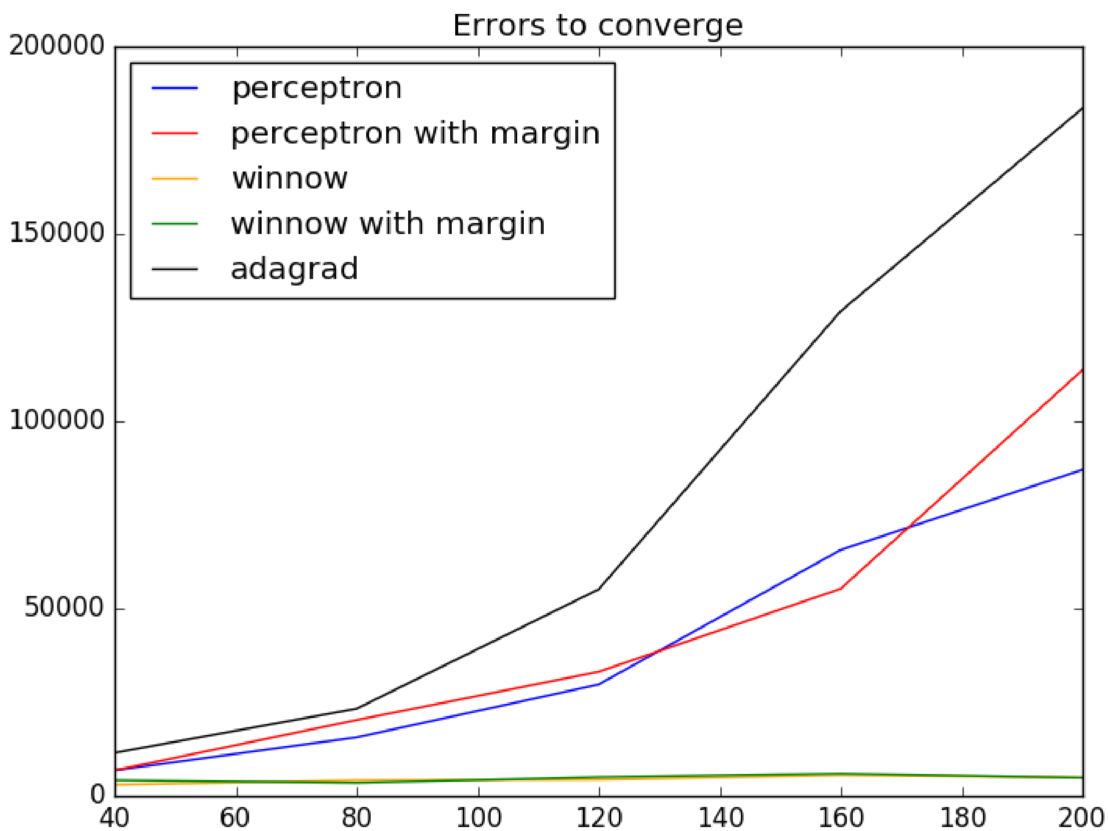




In these 5 algorithms, Winnow and Winnow with margin have the best mistake bound. On the graph they are extremely close. The yellow line and green line are almost together. Since the theoretical mistake bound for Winnow is O(klogn). The performance is expected. Three perceptron algorithm performed a linear mistake bound, which is also as expected,because the theoretical mistake bound is O(n). Perceptron with margin performs the worst. I think the reason is it has a margin, and that will increase the mistake since some correctly predicted examples in perceptron will be treated as mispredicted. Perceptron is more generous and it performs better than perceptron with margin. Adagrad is pretty close to the other two perceotron algorithms. It has a slightly higher mistake bound than perceptron. Probably because it tunes the learning rate. It learns the frequent features slowly and new features fast.

2. Answer to problem 2

Algorithm	parameters	n = 40	n = 80	n = 120	n = 160	n = 200
Perceptron	NO	NO	NO	NO	NO	NO
Perceptron w/ margin	η	1.5	0.25	0.25	0.25	0.03
Winnow	α	1.1	1.1	1.1	1.1	1.1
Winnow w/ margin	α, γ	1.1 ,2.0	1.1, 2.0	1.1 ,2.0	1.1, 2.0	1.1, 2.0
AdaGrad	η	1.5	1.5	1.5	1.5	1.5



In these 5 algorithms, Winnow and Winnow with margin have the best convergence rate. On the graph they are again extremely close. The yellow line and green line are almost together. Since the theoretical mistake bound for Winnow is O(klogn). The performance is expected. Three perceptron algorithm performed a linear mistake bound, which is also as expected,because the theoretical mistake bound is O(n). So their convergence rates are similar. Adagrad performs the worst when the feature space is bigger. Maybe the reason is the learning speed tuning. But the three algorithms all need above 10000 mistakes and below 20000 mistakes to converge. The speeds are actually pretty close.

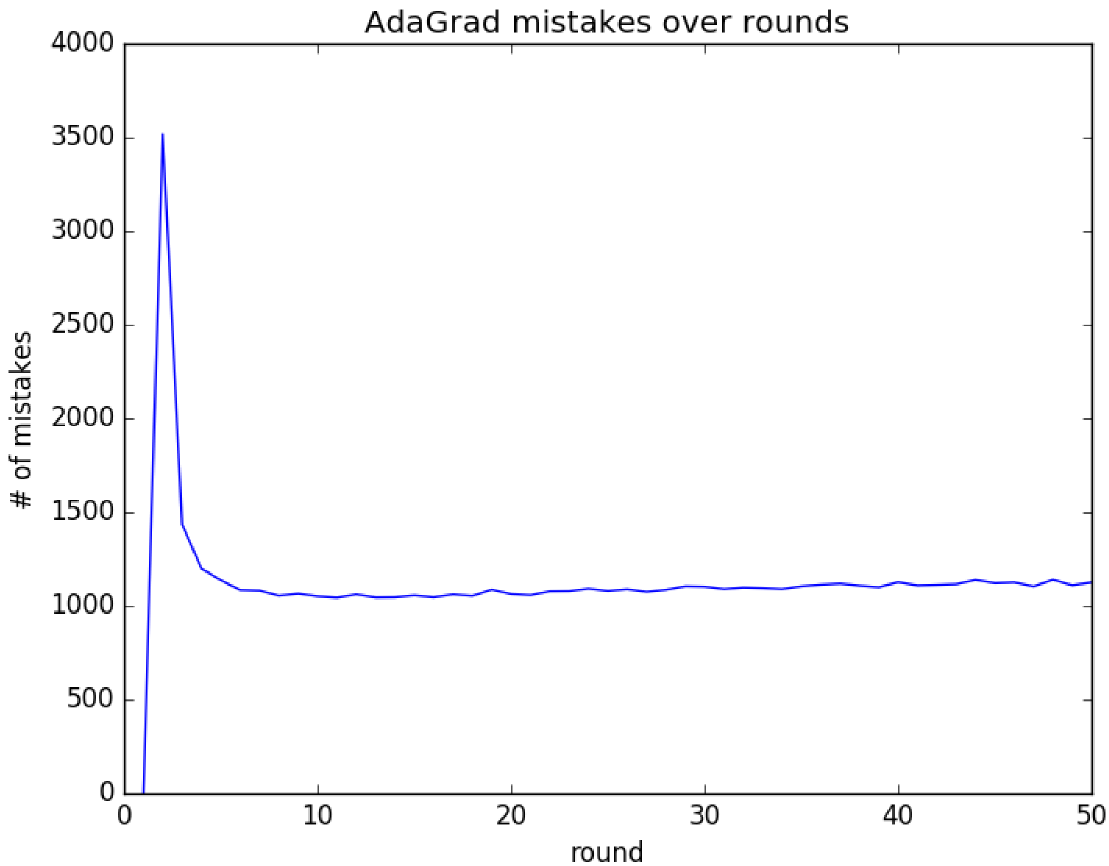
3. Answer to problem 3

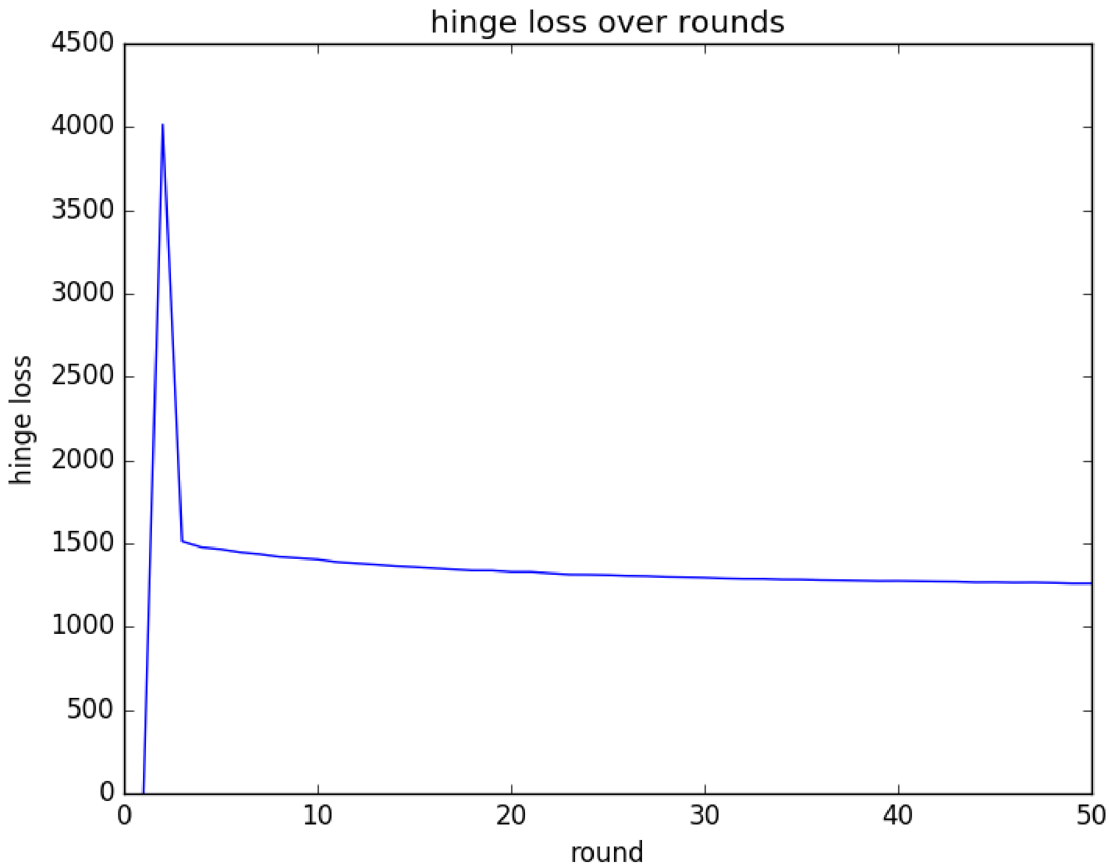
Algorithm	m=100		m=500		m=1000	
	acc.	params.	acc.	params.	acc.	params.
Perceptron	0.966		0.9175		0.7278	
Perceptron w/margin(η)	0.9935	0.03	0.9488	0.25	0.7843	0.03
Winnow(α)	0.9967	1.01	0.911	1.1	0.7695	1.1
Winnow w/margin (α, γ)	0.998	1.01,2.0	0.9088	1.1,0.006	0.7579	1.1,0.001
AdaGrad(η)	0.9996	0.25	0.937	1.5	0.7767	1.5

As we can see in the table. The overall accuracy is lower than previous experiments. The reason is the introduction of noise in training data. Noisy training data will reduce the accuracy of training result. And when applying the result w and theta to the test data, it will yield more errors. This is expected.

If compare the data horizontally, when the number m increases, the accuracy goes down. The reason is that m is the related feature in n total features. Since the total number of examples are the same for all 3 scenario, when m is larger, there's more noise in the data. If m is small, the pattern for correct prediction will be pretty easy to calculate. But when m is large and there's noise, more features are uncertain and the prediction result will be less accurate. In general the result meets my expectation.

4. Answer to problem 3


AdaGrad mistakes over rounds
# of mistakes
round
0
500
1000
1500
2000
2500
3000
3500
4000
0
10
20
30
40
50



As we can see in the graph, hinge loss and mistake count drops increases first and drops and finally come to a steady state. Hinge loss and mistake acts very similar over round. The graph shows how Adagrad algorithm learn the examples and converge. We can tell from the graph after less than 10 rounds, the result comes to a steady state. There's still a certain amount of error, because the data is noisy. As a result, no matter how many rounds we run the code, there will still be some mistakes.

I have appended all the results in the report.

Part1 result

Perceptron:
perceptron with n = 500
0.9931
perceptron with n = 1000
0.9664
bestresult: correct_500 = 0.9931 correct_1000 = 0.9664

Perceptron with margin:
perceptron(margin) with n = 500, r = 1.5
0.9931
perceptron(margin) with n = 1000, r = 1.5
0.9664
perceptron(margin) with n = 500, r = 0.25
0.9953
perceptron(margin) with n = 1000, r = 0.25
0.9822
perceptron(margin) with n = 500, r = 0.03
0.9975
perceptron(margin) with n = 1000, r = 0.03
0.9839
perceptron(margin) with n = 500, r = 0.005
0.9994
perceptron(margin) with n = 1000, r = 0.005
0.9844
perceptron(margin) with n = 500, r = 0.001
0.9958
perceptron(margin) with n = 1000, r = 0.001
0.9953
bestresult: correct_500 = 0.9958 correct_1000 = 0.9953 learning rate = 0.001

Winnow:
winnow with n = 500, alpha = 1.1
0.9998
winnow with n = 1000, alpha = 1.1
0.9994
winnow with n = 500, alpha = 1.01
0.9799
winnow with n = 1000, alpha = 1.01
0.967
winnow with n = 500, alpha = 1.005
0.9602

winnow with n = 1000, alpha = 1.005
0.8998
winnow with n = 500, alpha = 1.0005
0.5376
winnow with n = 1000, alpha = 1.0005
0.5255
winnow with n = 500, alpha = 1.0001
0.525
winnow with n = 1000, alpha = 1.0001
0.5197
bestresult: correct_500 = 0.9998 correct_1000 = 0.9994 alpha = 1.1

Winnow with margin:
winnow(margin) with n = 500, alpha = 1.1 gamma = 2.0
1.0
winnow(margin) with n = 1000, alpha = 1.1 gamma = 2.0
0.9992
winnow(margin) with n = 500, alpha = 1.1 gamma = 0.3
0.9981
winnow(margin) with n = 1000, alpha = 1.1 gamma = 0.3
0.9992
winnow(margin) with n = 500, alpha = 1.1 gamma = 0.04
0.9992
winnow(margin) with n = 1000, alpha = 1.1 gamma = 0.04
0.9996
winnow(margin) with n = 500, alpha = 1.1 gamma = 0.006
0.9998
winnow(margin) with n = 1000, alpha = 1.1 gamma = 0.006
0.9994
winnow(margin) with n = 500, alpha = 1.1 gamma = 0.001
0.9998
winnow(margin) with n = 1000, alpha = 1.1 gamma = 0.001
0.9994
winnow(margin) with n = 500, alpha = 1.01 gamma = 2.0
0.9866
winnow(margin) with n = 1000, alpha = 1.01 gamma = 2.0
0.9721
winnow(margin) with n = 500, alpha = 1.01 gamma = 0.3
0.9797
winnow(margin) with n = 1000, alpha = 1.01 gamma = 0.3
0.9688
winnow(margin) with n = 500, alpha = 1.01 gamma = 0.04
0.9804

winnow(margin) with n = 1000, alpha = 1.01 gamma = 0.04
0.9664
winnow(margin) with n = 500, alpha = 1.01 gamma = 0.006
0.9789
winnow(margin) with n = 1000, alpha = 1.01 gamma = 0.006
0.9673
winnow(margin) with n = 500, alpha = 1.01 gamma = 0.001
0.9796
winnow(margin) with n = 1000, alpha = 1.01 gamma = 0.001
0.9668
winnow(margin) with n = 500, alpha = 1.005 gamma = 2.0
0.9717
winnow(margin) with n = 1000, alpha = 1.005 gamma = 2.0
0.9144
winnow(margin) with n = 500, alpha = 1.005 gamma = 0.3
0.9618
winnow(margin) with n = 1000, alpha = 1.005 gamma = 0.3
0.8933
winnow(margin) with n = 500, alpha = 1.005 gamma = 0.04
0.9617
winnow(margin) with n = 1000, alpha = 1.005 gamma = 0.04
0.8913
winnow(margin) with n = 500, alpha = 1.005 gamma = 0.006
0.9606
winnow(margin) with n = 1000, alpha = 1.005 gamma = 0.006
0.8929
winnow(margin) with n = 500, alpha = 1.005 gamma = 0.001
0.9595
winnow(margin) with n = 1000, alpha = 1.005 gamma = 0.001
0.8932
winnow(margin) with n = 500, alpha = 1.0005 gamma = 2.0
0.5387
winnow(margin) with n = 1000, alpha = 1.0005 gamma = 2.0
0.525
winnow(margin) with n = 500, alpha = 1.0005 gamma = 0.3
0.5373
winnow(margin) with n = 1000, alpha = 1.0005 gamma = 0.3
0.5256
winnow(margin) with n = 500, alpha = 1.0005 gamma = 0.04
0.5376
winnow(margin) with n = 1000, alpha = 1.0005 gamma = 0.04
0.5244
winnow(margin) with n = 500, alpha = 1.0005 gamma = 0.006
0.5374

winnow(margin) with n = 1000, alpha = 1.0005 gamma = 0.006
0.5252
winnow(margin) with n = 500, alpha = 1.0005 gamma = 0.001
0.5377
winnow(margin) with n = 1000, alpha = 1.0005 gamma = 0.001
0.5252
winnow(margin) with n = 500, alpha = 1.0001 gamma = 2.0
0.5257
winnow(margin) with n = 1000, alpha = 1.0001 gamma = 2.0
0.5204
winnow(margin) with n = 500, alpha = 1.0001 gamma = 0.3
0.5254
winnow(margin) with n = 1000, alpha = 1.0001 gamma = 0.3
0.5195
winnow(margin) with n = 500, alpha = 1.0001 gamma = 0.04
0.5253
winnow(margin) with n = 1000, alpha = 1.0001 gamma = 0.04
0.5197
winnow(margin) with n = 500, alpha = 1.0001 gamma = 0.006
0.5253
winnow(margin) with n = 1000, alpha = 1.0001 gamma = 0.006
0.5197
winnow(margin) with n = 500, alpha = 1.0001 gamma = 0.001
0.525
winnow(margin) with n = 1000, alpha = 1.0001 gamma = 0.001
0.5198
bestresult: correct_500 = 1.0 correct_1000 = 0.9992 alpha = 1.1 gamma = 2.0

Adagrad:
adagrad with n = 500, alpha = 1.5
0.9827
adagrad with n = 1000, alpha = 1.5
0.9946
adagrad with n = 500, alpha = 0.25
0.9906
adagrad with n = 1000, alpha = 0.25
0.9947
adagrad with n = 500, alpha = 0.03
0.9581
adagrad with n = 1000, alpha = 0.03
0.9428
adagrad with n = 500, alpha = 0.005
0.6657
adagrad with n = 1000, alpha = 0.005

0.6202
adagrad with n = 500, alpha = 0.001
0.4977
adagrad with n = 1000, alpha = 0.001
0.5
bestresult: correct_500 = 0.9906 correct_1000 = 0.9947 learning rate = 0.25

Part2 Results

Perceptron:
perceptron with n = 40
1.0
perceptron with n = 80
1.0
perceptron with n = 120
0.9999
perceptron with n = 160
0.9999
perceptron with n = 200
0.9994
bestresult: correct1 = 1.0 n = 40

Perceptron with margin:
perceptron_margin with n = 40, r = 1.5
1.0
perceptron_margin with n = 40, r = 0.25
1.0
perceptron_margin with n = 40, r = 0.03
1.0
perceptron_margin with n = 40, r = 0.005
1.0
perceptron_margin with n = 40, r = 0.001
1.0
bestresult for n = 40 : correct1 = 1.0 learning rate = 1.5
perceptron_margin with n = 80, r = 1.5
0.9998
perceptron_margin with n = 80, r = 0.25
1.0
perceptron_margin with n = 80, r = 0.03
1.0
perceptron_margin with n = 80, r = 0.005
1.0
perceptron_margin with n = 80, r = 0.001
1.0
bestresult for n = 80 : correct1 = 1.0 learning rate = 0.25
perceptron_margin with n = 120, r = 1.5
0.9998
perceptron_margin with n = 120, r = 0.25
1.0
perceptron_margin with n = 120, r = 0.03

1.0
perceptron_margin with n = 120, r = 0.005
1.0
perceptron_margin with n = 120, r = 0.001
0.9959
bestresult for n = 120 : correct1 = 1.0 learning rate = 0.25
perceptron_margin with n = 160, r = 1.5
0.9997
perceptron_margin with n = 160, r = 0.25
1.0
perceptron_margin with n = 160, r = 0.03
1.0
perceptron_margin with n = 160, r = 0.005
1.0
perceptron_margin with n = 160, r = 0.001
0.9867
bestresult for n = 160 : correct1 = 1.0 learning rate = 0.25
perceptron_margin with n = 200, r = 1.5
0.9999
perceptron_margin with n = 200, r = 0.25
0.9996
perceptron_margin with n = 200, r = 0.03
1.0
perceptron_margin with n = 200, r = 0.005
1.0
perceptron_margin with n = 200, r = 0.001
0.9842
bestresult for n = 200 : correct1 = 1.0 learning rate = 0.03

Winnow:
winnow with n = 40, alpha = 1.1
0.9999
winnow with n = 40, alpha = 1.01
0.9991
winnow with n = 40, alpha = 1.005
0.9987
winnow with n = 40, alpha = 1.0005
0.9917
winnow with n = 40, alpha = 1.0001
0.8588
bestresult n = 40: correct1 = 0.9999 alpha = 1.1
winnow with n = 80, alpha = 1.1
1.0

winnow with n = 80, alpha = 1.01
0.9983
winnow with n = 80, alpha = 1.005
0.9981
winnow with n = 80, alpha = 1.0005
0.9743
winnow with n = 80, alpha = 1.0001
0.6979
bestresult n = 80: correct1 = 1.0 alpha = 1.1
winnow with n = 120, alpha = 1.1
1.0
winnow with n = 120, alpha = 1.01
0.9973
winnow with n = 120, alpha = 1.005
0.9971
winnow with n = 120, alpha = 1.0005
0.9496
winnow with n = 120, alpha = 1.0001
0.6431
bestresult n = 120: correct1 = 1.0 alpha = 1.1
winnow with n = 160, alpha = 1.1
0.9993
winnow with n = 160, alpha = 1.01
0.9971
winnow with n = 160, alpha = 1.005
0.9968
winnow with n = 160, alpha = 1.0005
0.9362
winnow with n = 160, alpha = 1.0001
0.6188
bestresult n = 160: correct1 = 0.9993 alpha = 1.1
winnow with n = 200, alpha = 1.1
0.9998
winnow with n = 200, alpha = 1.01
0.9983
winnow with n = 200, alpha = 1.005
0.998
winnow with n = 200, alpha = 1.0005
0.922
winnow with n = 200, alpha = 1.0001
0.6081
bestresult n = 200: correct1 = 0.9998 alpha = 1.1

Winnow with margin:
winnow_margin with n = 40, alpha = 1.1 gamma = 2.0
1.0
winnow_margin with n = 40, alpha = 1.1 gamma = 0.3
0.9998
winnow_margin with n = 40, alpha = 1.1 gamma = 0.04
0.9998
winnow_margin with n = 40, alpha = 1.1 gamma = 0.006
0.9999
winnow_margin with n = 40, alpha = 1.1 gamma = 0.001
0.9999
winnow_margin with n = 40, alpha = 1.01 gamma = 2.0
1.0
winnow_margin with n = 40, alpha = 1.01 gamma = 0.3
0.9995
winnow_margin with n = 40, alpha = 1.01 gamma = 0.04
0.9991
winnow_margin with n = 40, alpha = 1.01 gamma = 0.006
0.999
winnow_margin with n = 40, alpha = 1.01 gamma = 0.001
0.9988
winnow_margin with n = 40, alpha = 1.005 gamma = 2.0
1.0
winnow_margin with n = 40, alpha = 1.005 gamma = 0.3
0.9994
winnow_margin with n = 40, alpha = 1.005 gamma = 0.04
0.9987
winnow_margin with n = 40, alpha = 1.005 gamma = 0.006
0.9987
winnow_margin with n = 40, alpha = 1.005 gamma = 0.001
0.9987
winnow_margin with n = 40, alpha = 1.0005 gamma = 2.0
1.0
winnow_margin with n = 40, alpha = 1.0005 gamma = 0.3
0.9939
winnow_margin with n = 40, alpha = 1.0005 gamma = 0.04
0.9927
winnow_margin with n = 40, alpha = 1.0005 gamma = 0.006
0.9918
winnow_margin with n = 40, alpha = 1.0005 gamma = 0.001
0.9917
winnow_margin with n = 40, alpha = 1.0001 gamma = 2.0
0.8955
winnow_margin with n = 40, alpha = 1.0001 gamma = 0.3

0.8619
winnow_margin with n = 40, alpha = 1.0001 gamma = 0.04
0.861
winnow_margin with n = 40, alpha = 1.0001 gamma = 0.006
0.8594
winnow_margin with n = 40, alpha = 1.0001 gamma = 0.001
0.8591
bestresult for n = 40: correct1 = 1.0 alpha = 1.1 gamma = 2.0
winnow_margin with n = 80, alpha = 1.1 gamma = 2.0
1.0
winnow_margin with n = 80, alpha = 1.1 gamma = 0.3
1.0
winnow_margin with n = 80, alpha = 1.1 gamma = 0.04
1.0
winnow_margin with n = 80, alpha = 1.1 gamma = 0.006
1.0
winnow_margin with n = 80, alpha = 1.1 gamma = 0.001
1.0
winnow_margin with n = 80, alpha = 1.01 gamma = 2.0
1.0
winnow_margin with n = 80, alpha = 1.01 gamma = 0.3
0.9988
winnow_margin with n = 80, alpha = 1.01 gamma = 0.04
0.9988
winnow_margin with n = 80, alpha = 1.01 gamma = 0.006
0.9982
winnow_margin with n = 80, alpha = 1.01 gamma = 0.001
0.9983
winnow_margin with n = 80, alpha = 1.005 gamma = 2.0
0.9999
winnow_margin with n = 80, alpha = 1.005 gamma = 0.3
0.9988
winnow_margin with n = 80, alpha = 1.005 gamma = 0.04
0.9985
winnow_margin with n = 80, alpha = 1.005 gamma = 0.006
0.9983
winnow_margin with n = 80, alpha = 1.005 gamma = 0.001
0.9982
winnow_margin with n = 80, alpha = 1.0005 gamma = 2.0
0.9967
winnow_margin with n = 80, alpha = 1.0005 gamma = 0.3
0.9783
winnow_margin with n = 80, alpha = 1.0005 gamma = 0.04
0.9753

winnow_margin with n = 80, alpha = 1.0005 gamma = 0.006
0.9744
winnow_margin with n = 80, alpha = 1.0005 gamma = 0.001
0.9742
winnow_margin with n = 80, alpha = 1.0001 gamma = 2.0
0.73
winnow_margin with n = 80, alpha = 1.0001 gamma = 0.3
0.7022
winnow_margin with n = 80, alpha = 1.0001 gamma = 0.04
0.6983
winnow_margin with n = 80, alpha = 1.0001 gamma = 0.006
0.6979
winnow_margin with n = 80, alpha = 1.0001 gamma = 0.001
0.698
bestresult for n = 80: correct1 = 1.0 alpha = 1.1 gamma = 2.0
winnow_margin with n = 120, alpha = 1.1 gamma = 2.0
1.0
winnow_margin with n = 120, alpha = 1.1 gamma = 0.3
0.9997
winnow_margin with n = 120, alpha = 1.1 gamma = 0.04
1.0
winnow_margin with n = 120, alpha = 1.1 gamma = 0.006
1.0
winnow_margin with n = 120, alpha = 1.1 gamma = 0.001
1.0
winnow_margin with n = 120, alpha = 1.01 gamma = 2.0
0.9998
winnow_margin with n = 120, alpha = 1.01 gamma = 0.3
0.9982
winnow_margin with n = 120, alpha = 1.01 gamma = 0.04
0.9971
winnow_margin with n = 120, alpha = 1.01 gamma = 0.006
0.9975
winnow_margin with n = 120, alpha = 1.01 gamma = 0.001
0.9975
winnow_margin with n = 120, alpha = 1.005 gamma = 2.0
0.9996
winnow_margin with n = 120, alpha = 1.005 gamma = 0.3
0.9977
winnow_margin with n = 120, alpha = 1.005 gamma = 0.04
0.997
winnow_margin with n = 120, alpha = 1.005 gamma = 0.006
0.9971
winnow_margin with n = 120, alpha = 1.005 gamma = 0.001

0.9972
winnow_margin with n = 120, alpha = 1.0005 gamma = 2.0
0.9853
winnow_margin with n = 120, alpha = 1.0005 gamma = 0.3
0.9568
winnow_margin with n = 120, alpha = 1.0005 gamma = 0.04
0.9508
winnow_margin with n = 120, alpha = 1.0005 gamma = 0.006
0.9498
winnow_margin with n = 120, alpha = 1.0005 gamma = 0.001
0.9495
winnow_margin with n = 120, alpha = 1.0001 gamma = 2.0
0.6574
winnow_margin with n = 120, alpha = 1.0001 gamma = 0.3
0.6465
winnow_margin with n = 120, alpha = 1.0001 gamma = 0.04
0.6433
winnow_margin with n = 120, alpha = 1.0001 gamma = 0.006
0.643
winnow_margin with n = 120, alpha = 1.0001 gamma = 0.001
0.643
bestresult for n = 120: correct1 = 1.0 alpha = 1.1 gamma = 2.0
winnow_margin with n = 160, alpha = 1.1 gamma = 2.0
1.0
winnow_margin with n = 160, alpha = 1.1 gamma = 0.3
1.0
winnow_margin with n = 160, alpha = 1.1 gamma = 0.04
0.9996
winnow_margin with n = 160, alpha = 1.1 gamma = 0.006
0.9993
winnow_margin with n = 160, alpha = 1.1 gamma = 0.001
0.9993
winnow_margin with n = 160, alpha = 1.01 gamma = 2.0
0.9996
winnow_margin with n = 160, alpha = 1.01 gamma = 0.3
0.9981
winnow_margin with n = 160, alpha = 1.01 gamma = 0.04
0.9974
winnow_margin with n = 160, alpha = 1.01 gamma = 0.006
0.9974
winnow_margin with n = 160, alpha = 1.01 gamma = 0.001
0.9971
winnow_margin with n = 160, alpha = 1.005 gamma = 2.0
0.9992

winnow_margin with n = 160, alpha = 1.005 gamma = 0.3
0.997
winnow_margin with n = 160, alpha = 1.005 gamma = 0.04
0.9965
winnow_margin with n = 160, alpha = 1.005 gamma = 0.006
0.9969
winnow_margin with n = 160, alpha = 1.005 gamma = 0.001
0.9968
winnow_margin with n = 160, alpha = 1.0005 gamma = 2.0
0.9709
winnow_margin with n = 160, alpha = 1.0005 gamma = 0.3
0.943
winnow_margin with n = 160, alpha = 1.0005 gamma = 0.04
0.9374
winnow_margin with n = 160, alpha = 1.0005 gamma = 0.006
0.9363
winnow_margin with n = 160, alpha = 1.0005 gamma = 0.001
0.9365
winnow_margin with n = 160, alpha = 1.0001 gamma = 2.0
0.6259
winnow_margin with n = 160, alpha = 1.0001 gamma = 0.3
0.6203
winnow_margin with n = 160, alpha = 1.0001 gamma = 0.04
0.6189
winnow_margin with n = 160, alpha = 1.0001 gamma = 0.006
0.619
winnow_margin with n = 160, alpha = 1.0001 gamma = 0.001
0.6189
bestresult for n = 160: correct1 = 1.0 alpha = 1.1 gamma = 2.0
winnow_margin with n = 200, alpha = 1.1 gamma = 2.0
1.0
winnow_margin with n = 200, alpha = 1.1 gamma = 0.3
0.9999
winnow_margin with n = 200, alpha = 1.1 gamma = 0.04
0.9999
winnow_margin with n = 200, alpha = 1.1 gamma = 0.006
0.9998
winnow_margin with n = 200, alpha = 1.1 gamma = 0.001
0.9998
winnow_margin with n = 200, alpha = 1.01 gamma = 2.0
0.9994
winnow_margin with n = 200, alpha = 1.01 gamma = 0.3
0.9986
winnow_margin with n = 200, alpha = 1.01 gamma = 0.04

0.9983
winnow_margin with n = 200, alpha = 1.01 gamma = 0.006
0.998
winnow_margin with n = 200, alpha = 1.01 gamma = 0.001
0.9983
winnow_margin with n = 200, alpha = 1.005 gamma = 2.0
0.9994
winnow_margin with n = 200, alpha = 1.005 gamma = 0.3
0.9983
winnow_margin with n = 200, alpha = 1.005 gamma = 0.04
0.9981
winnow_margin with n = 200, alpha = 1.005 gamma = 0.006
0.9979
winnow_margin with n = 200, alpha = 1.005 gamma = 0.001
0.998
winnow_margin with n = 200, alpha = 1.0005 gamma = 2.0
0.9552
winnow_margin with n = 200, alpha = 1.0005 gamma = 0.3
0.9266
winnow_margin with n = 200, alpha = 1.0005 gamma = 0.04
0.9226
winnow_margin with n = 200, alpha = 1.0005 gamma = 0.006
0.9225
winnow_margin with n = 200, alpha = 1.0005 gamma = 0.001
0.9221
winnow_margin with n = 200, alpha = 1.0001 gamma = 2.0
0.6143
winnow_margin with n = 200, alpha = 1.0001 gamma = 0.3
0.6092
winnow_margin with n = 200, alpha = 1.0001 gamma = 0.04
0.6084
winnow_margin with n = 200, alpha = 1.0001 gamma = 0.006
0.6085
winnow_margin with n = 200, alpha = 1.0001 gamma = 0.001
0.6081
bestresult for n = 200: correct1 = 1.0 alpha = 1.1 gamma = 2.0

Adagrad:
adagrad with n = 40, r = 1.5
1.0
adagrad with n = 40, r = 0.25
1.0
adagrad with n = 40, r = 0.03

0.7916
adagrad with n = 40, r = 0.005
0.5034
adagrad with n = 40, r = 0.001
0.4925
bestresult for n = 40: correct1 = 1.0 learning rate = 1.5
adagrad with n = 80, r = 1.5
1.0
adagrad with n = 80, r = 0.25
1.0
adagrad with n = 80, r = 0.03
0.852
adagrad with n = 80, r = 0.005
0.7071
adagrad with n = 80, r = 0.001
0.508
bestresult for n = 80: correct1 = 1.0 learning rate = 1.5
adagrad with n = 120, r = 1.5
1.0
adagrad with n = 120, r = 0.25
1.0
adagrad with n = 120, r = 0.03
0.8925
adagrad with n = 120, r = 0.005
0.7441
adagrad with n = 120, r = 0.001
0.4957
bestresult for n = 120: correct1 = 1.0 learning rate = 1.5
adagrad with n = 160, r = 1.5
1.0
adagrad with n = 160, r = 0.25
0.9994
adagrad with n = 160, r = 0.03
0.9218
adagrad with n = 160, r = 0.005
0.7467
adagrad with n = 160, r = 0.001
0.4982
bestresult for n = 160: correct1 = 1.0 learning rate = 1.5
adagrad with n = 200, r = 1.5
0.9982
adagrad with n = 200, r = 0.25
0.9945
adagrad with n = 200, r = 0.03

0.943
adagrad with n = 200, r = 0.005
0.8108
adagrad with n = 200, r = 0.001
0.5011
bestresult for n = 200: correct1 = 0.9982 learning rate = 1.5

Error Converge Matrix:

First line is perceptron, then perceptron with margin, etc.

```
[[   6813.   15613.   29726.   65649.   87072.]
 [   6813.   20237.   33113.   55279.  113703.]
 [   2991.    4164.    4310.    5496.    4864.]
 [   4155.    3481.    4974.    5909.    4809.]
 [  11487.   23261.   55118.  129357.  183558.]]
```

Part3 Results

Perceptron
perceptron with n = 100
0.8016
perceptron with n = 500
0.6478
perceptron with n = 1000
0.6754
bestresult: correct1 = 0.8016 m = 100

Perceptron with margin
perceptron_margin with m = 100, r = 1.5
0.8016
perceptron_margin with m = 100, r = 0.25
0.7958
perceptron_margin with m = 100, r = 0.03
0.8224
perceptron_margin with m = 100, r = 0.005
0.7634
perceptron_margin with m = 100, r = 0.001
0.6798
bestresult for m = 100 : correct1 = 0.8224 learning rate = 0.03
perceptron_margin with m = 500, r = 1.5
0.6478
perceptron_margin with m = 500, r = 0.25
0.6574
perceptron_margin with m = 500, r = 0.03
0.6454
perceptron_margin with m = 500, r = 0.005
0.628
perceptron_margin with m = 500, r = 0.001
0.5588
bestresult for m = 500 : correct1 = 0.6574 learning rate = 0.25
perceptron_margin with m = 1000, r = 1.5
0.6754
perceptron_margin with m = 1000, r = 0.25
0.7168
perceptron_margin with m = 1000, r = 0.03
0.7228
perceptron_margin with m = 1000, r = 0.005

0.6272
perceptron_margin with m = 1000, r = 0.001
0.56
bestresult for m = 1000 : correct1 = 0.7228 learning rate = 0.03

Winnow:
winnow with m = 100, alpha = 1.1
0.7784
winnow with m = 100, alpha = 1.01
0.819
winnow with m = 100, alpha = 1.005
0.7466
winnow with m = 100, alpha = 1.0005
0.5982
winnow with m = 100, alpha = 1.0001
0.5832
bestresult m = 100: correct1 = 0.819 alpha = 1.01
winnow with m = 500, alpha = 1.1
0.7978
winnow with m = 500, alpha = 1.01
0.545
winnow with m = 500, alpha = 1.005
0.5294
winnow with m = 500, alpha = 1.0005
0.5248
winnow with m = 500, alpha = 1.0001
0.5188
bestresult m = 500: correct1 = 0.7978 alpha = 1.1
winnow with m = 1000, alpha = 1.1
0.7406
winnow with m = 1000, alpha = 1.01
0.654
winnow with m = 1000, alpha = 1.005
0.4932
winnow with m = 1000, alpha = 1.0005
0.4932
winnow with m = 1000, alpha = 1.0001
0.4932
bestresult m = 1000: correct1 = 0.7406 alpha = 1.1

Winnow with margin:
winnow_margin with m = 100, alpha = 1.1 gamma = 2.0
0.8916

winnow_margin with m = 100, alpha = 1.1 gamma = 0.3
0.8808
winnow_margin with m = 100, alpha = 1.1 gamma = 0.04
0.8756
winnow_margin with m = 100, alpha = 1.1 gamma = 0.006
0.8414
winnow_margin with m = 100, alpha = 1.1 gamma = 0.001
0.8908
winnow_margin with m = 100, alpha = 1.01 gamma = 2.0
0.9002
winnow_margin with m = 100, alpha = 1.01 gamma = 0.3
0.8334
winnow_margin with m = 100, alpha = 1.01 gamma = 0.04
0.8208
winnow_margin with m = 100, alpha = 1.01 gamma = 0.006
0.8188
winnow_margin with m = 100, alpha = 1.01 gamma = 0.001
0.8184
winnow_margin with m = 100, alpha = 1.005 gamma = 2.0
0.8648
winnow_margin with m = 100, alpha = 1.005 gamma = 0.3
0.7606
winnow_margin with m = 100, alpha = 1.005 gamma = 0.04
0.7486
winnow_margin with m = 100, alpha = 1.005 gamma = 0.006
0.7462
winnow_margin with m = 100, alpha = 1.005 gamma = 0.001
0.7464
winnow_margin with m = 100, alpha = 1.0005 gamma = 2.0
0.609
winnow_margin with m = 100, alpha = 1.0005 gamma = 0.3
0.6
winnow_margin with m = 100, alpha = 1.0005 gamma = 0.04
0.5984
winnow_margin with m = 100, alpha = 1.0005 gamma = 0.006
0.5982
winnow_margin with m = 100, alpha = 1.0005 gamma = 0.001
0.5984
winnow_margin with m = 100, alpha = 1.0001 gamma = 2.0
0.5854
winnow_margin with m = 100, alpha = 1.0001 gamma = 0.3
0.5822
winnow_margin with m = 100, alpha = 1.0001 gamma = 0.04
0.5828

winnow_margin with m = 100, alpha = 1.0001 gamma = 0.006
0.583
winnow_margin with m = 100, alpha = 1.0001 gamma = 0.001
0.583
bestresult for m = 100: correct1 = 0.9002 alpha = 1.01 gamma = 2.0
winnow_margin with m = 500, alpha = 1.1 gamma = 2.0
0.7996
winnow_margin with m = 500, alpha = 1.1 gamma = 0.3
0.7912
winnow_margin with m = 500, alpha = 1.1 gamma = 0.04
0.7986
winnow_margin with m = 500, alpha = 1.1 gamma = 0.006
0.803
winnow_margin with m = 500, alpha = 1.1 gamma = 0.001
0.7938
winnow_margin with m = 500, alpha = 1.01 gamma = 2.0
0.551
winnow_margin with m = 500, alpha = 1.01 gamma = 0.3
0.5492
winnow_margin with m = 500, alpha = 1.01 gamma = 0.04
0.5478
winnow_margin with m = 500, alpha = 1.01 gamma = 0.006
0.5448
winnow_margin with m = 500, alpha = 1.01 gamma = 0.001
0.5464
winnow_margin with m = 500, alpha = 1.005 gamma = 2.0
0.5308
winnow_margin with m = 500, alpha = 1.005 gamma = 0.3
0.5292
winnow_margin with m = 500, alpha = 1.005 gamma = 0.04
0.5294
winnow_margin with m = 500, alpha = 1.005 gamma = 0.006
0.529
winnow_margin with m = 500, alpha = 1.005 gamma = 0.001
0.5304
winnow_margin with m = 500, alpha = 1.0005 gamma = 2.0
0.5202
winnow_margin with m = 500, alpha = 1.0005 gamma = 0.3
0.525
winnow_margin with m = 500, alpha = 1.0005 gamma = 0.04
0.5244
winnow_margin with m = 500, alpha = 1.0005 gamma = 0.006
0.525
winnow_margin with m = 500, alpha = 1.0005 gamma = 0.001

0.5246
winnow_margin with m = 500, alpha = 1.0001 gamma = 2.0
0.5182
winnow_margin with m = 500, alpha = 1.0001 gamma = 0.3
0.5186
winnow_margin with m = 500, alpha = 1.0001 gamma = 0.04
0.5186
winnow_margin with m = 500, alpha = 1.0001 gamma = 0.006
0.5188
winnow_margin with m = 500, alpha = 1.0001 gamma = 0.001
0.5188
bestresult for m = 500: correct1 = 0.803 alpha = 1.1 gamma = 0.006
winnow_margin with m = 1000, alpha = 1.1 gamma = 2.0
0.738
winnow_margin with m = 1000, alpha = 1.1 gamma = 0.3
0.7398
winnow_margin with m = 1000, alpha = 1.1 gamma = 0.04
0.7394
winnow_margin with m = 1000, alpha = 1.1 gamma = 0.006
0.739
winnow_margin with m = 1000, alpha = 1.1 gamma = 0.001
0.7406
winnow_margin with m = 1000, alpha = 1.01 gamma = 2.0
0.6542
winnow_margin with m = 1000, alpha = 1.01 gamma = 0.3
0.6548
winnow_margin with m = 1000, alpha = 1.01 gamma = 0.04
0.6544
winnow_margin with m = 1000, alpha = 1.01 gamma = 0.006
0.654
winnow_margin with m = 1000, alpha = 1.01 gamma = 0.001
0.654
winnow_margin with m = 1000, alpha = 1.005 gamma = 2.0
0.4932
winnow_margin with m = 1000, alpha = 1.005 gamma = 0.3
0.4932
winnow_margin with m = 1000, alpha = 1.005 gamma = 0.04
0.4932
winnow_margin with m = 1000, alpha = 1.005 gamma = 0.006
0.4932
winnow_margin with m = 1000, alpha = 1.005 gamma = 0.001
0.4932
winnow_margin with m = 1000, alpha = 1.0005 gamma = 2.0
0.4932

winnow_margin with m = 1000, alpha = 1.0005 gamma = 0.3
0.4932
winnow_margin with m = 1000, alpha = 1.0005 gamma = 0.04
0.4932
winnow_margin with m = 1000, alpha = 1.0005 gamma = 0.006
0.4932
winnow_margin with m = 1000, alpha = 1.0005 gamma = 0.001
0.4932
winnow_margin with m = 1000, alpha = 1.0001 gamma = 2.0
0.4932
winnow_margin with m = 1000, alpha = 1.0001 gamma = 0.3
0.4932
winnow_margin with m = 1000, alpha = 1.0001 gamma = 0.04
0.4932
winnow_margin with m = 1000, alpha = 1.0001 gamma = 0.006
0.4932
winnow_margin with m = 1000, alpha = 1.0001 gamma = 0.001
0.4932
bestresult for m = 1000: correct1 = 0.7406 alpha = 1.1 gamma = 0.001

Adagrad
adagrad with m = 100, r = 1.5
0.8482
adagrad with m = 100, r = 0.25
0.8884
adagrad with m = 100, r = 0.03
0.666
adagrad with m = 100, r = 0.005
0.565
adagrad with m = 100, r = 0.001
0.5024
bestresult for m = 100: correct1 = 0.8884 learning rate = 0.25
adagrad with m = 500, r = 1.5
0.7664
adagrad with m = 500, r = 0.25
0.7568
adagrad with m = 500, r = 0.03
0.5936
adagrad with m = 500, r = 0.005
0.544
adagrad with m = 500, r = 0.001
0.5014
bestresult for m = 500: correct1 = 0.7664 learning rate = 1.5

adagrad with m = 1000, r = 1.5
0.7394
adagrad with m = 1000, r = 0.25
0.6508
adagrad with m = 1000, r = 0.03
0.5472
adagrad with m = 1000, r = 0.005
0.5066
adagrad with m = 1000, r = 0.001
0.5066
bestresult for m = 1000: correct1 = 0.7394 learning rate = 1.5

TESTs

Perceptron:
TEST perceptron with m = 100
0.966
TEST perceptron with m = 500
0.9175
TEST perceptron with m = 1000
0.7278

Perceptron with margin:
TEST perceptron_margin with m = 100, learning rate = 0.005
0.9935
TEST perceptron_margin with m = 500, learning rate = 0.03
0.9488
TEST perceptron_margin with m = 1000, learning rate = 0.25
0.7843

Winnow
TESTwinnow with m = 100, alpha = 1.01
0.9667
TESTwinnow with m = 500, alpha = 1.1
0.911
TESTwinnow with m = 1000, alpha = 1.1
0.7695

Winnow with margin:
TEST winnow_margin with m = 100, alpha = 1.01 gamma = 2.0
0.998
TEST winnow_margin with m = 500, alpha = 1.1 gamma = 0.006
0.9088

TEST winnow_margin with m = 1000, alpha = 1.1 gamma = 0.001
0.7579

Adagrad:
TEST adagrad with m = 100, r = 0.25
0.9996
TEST adagrad with m = 100, r = 1.5
0.937
TEST adagrad with m = 100, r = 1.5
0.7767