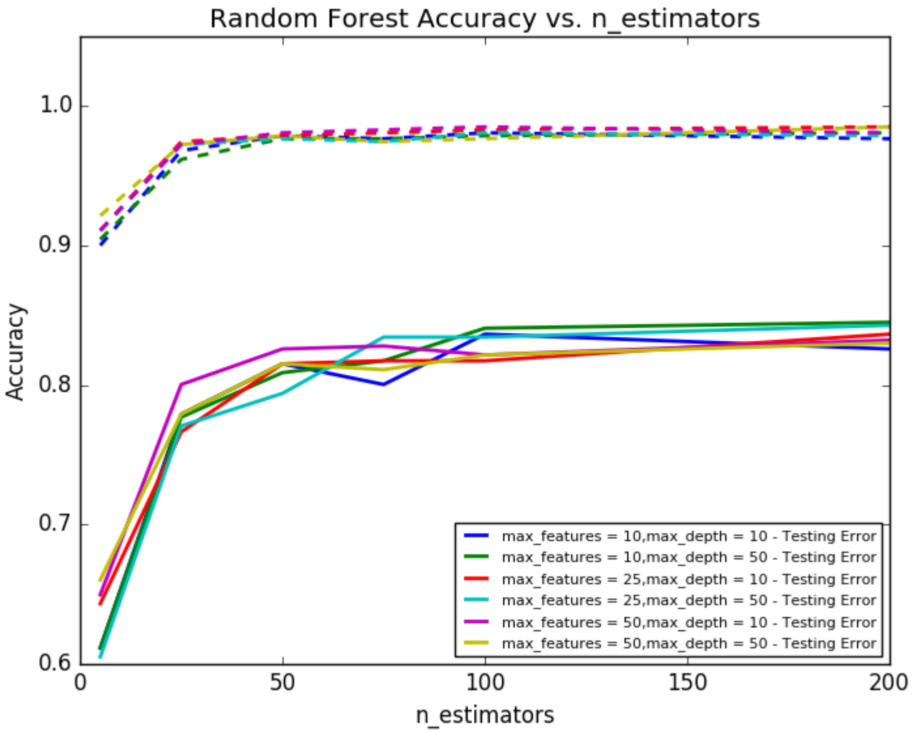
Alec Chapman
Machine Learning
Spring 2017
Homework 6

1. Random Forest

MNIST

a.



All results (sorted by test accuracy, 5 best models in bold):

max_features	max_depth	n_estimators	train accuracy	test accuracy
10	50	200	0.98089172	0.845010616
25	50	200	0.978768577	0.842887473
10	50	100	0.978768577	0.840764331
10	10	100	0.98089172	0.836518047
25	10	200	0.985138004	0.836518047

25	50	75	0.974522293	0.834394904
25	50	100	0.98089172	0.834394904
50	10	200	0.98089172	0.832271762
50	50	200	0.985138004	0.83014862
50	10	75	0.983014862	0.828025478
10	10	200	0.976645435	0.825902335
50	10	50	0.98089172	0.825902335
50	10	100	0.985138004	0.821656051
50	50	100	0.976645435	0.821656051
10	50	75	0.976645435	0.817409766
25	10	75	0.98089172	0.817409766
25	10	100	0.983014862	0.817409766
10	10	50	0.978768577	0.815286624
25	10	50	0.978768577	0.815286624
50	50	50	0.978768577	0.815286624
50	50	75	0.974522293	0.81104034
10	50	50	0.976645435	0.808917197
10	10	75	0.976645435	0.800424628
50	10	25	0.972399151	0.800424628
25	50	50	0.976645435	0.794055202
10	10	25	0.968152866	0.779193206
50	50	25	0.972399151	0.779193206
10	50	25	0.961783439	0.777070064

25	50	25	0.972399151	0.770700637
25	10	25	0.974522293	0.766454352
50	50	5	0.921443737	0.66029724
50	10	5	0.910828025	0.649681529
25	10	5	0.910828025	0.643312102
10	10	5	0.900212314	0.611464968
10	50	5	0.904458599	0.611464968
25	50	5	0.910828025	0.605095541

b. For a Random Forest ensemble, I would use a model with the following parameters:

- Max features = 10
- Max depth = 50
- N of learners = 100 or 200

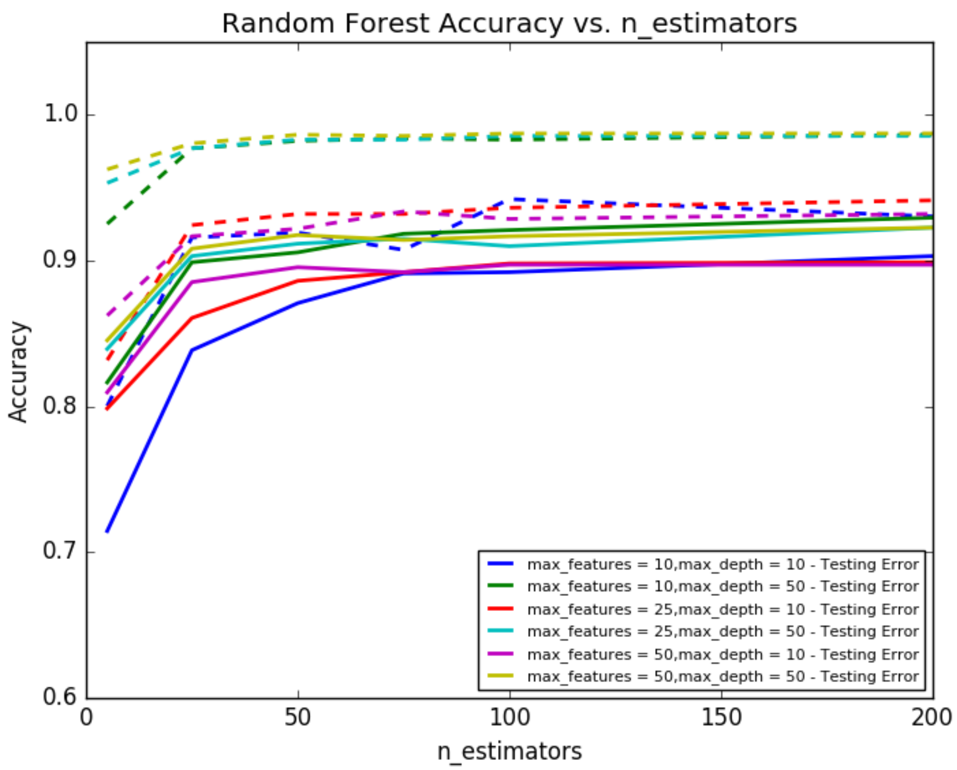
A small number of features works with this dataset. I had also experimented using the square root of features and not setting a limit, but 10 ended up working better. A small number of features allows the learner to focus on important features and disregard ones that aren't helpful. However, I imagine that other datasets might need more than 10 features. I selected a max depth of 50. I'd also experimented with having an unlimited depth, and that didn't work as well. Finally, the best performing model using n=200 learners. However, the difference between that and n=learners was fairly small (0.005), so it may not be worth the computational power to use 200 learners.

10	50	200	0.98089172	0.845010616
10	50	100	0.978768577	0.840764331

c. I do not see any evidence of over or underfitting.

20NG

a.



All results (sorted by test accuracy, 5 best models in bold):

max_features	max_depth	n_estimators	train accuracy	test accuracy
10	50	200	0.986406117	0.929481733
25	50	200	0.9855565	0.922684792
50	50	200	0.987255735	0.922684792
10	50	100	0.983007647	0.920985556
10	50	75	0.983857264	0.918436703
50	50	50	0.986406117	0.917587086
50	50	100	0.987255735	0.916737468
25	50	75	0.983007647	0.915038233
50	50	75	0.9855565	0.914188615

25	50	50	0.983007647	0.911639762
25	50	100	0.9855565	0.909940527
50	50	25	0.980458794	0.908241291
10	50	50	0.982158029	0.905692438
10	10	200	0.930331351	0.903143585
25	50	25	0.977060323	0.903143585
10	50	25	0.977060323	0.898895497
25	10	200	0.941376381	0.898895497
25	10	100	0.936278675	0.898045879
50	10	100	0.928632116	0.897196262
50	10	200	0.932030586	0.897196262
50	10	50	0.921835174	0.895497026
10	10	100	0.942225998	0.892098556
25	10	75	0.932030586	0.892098556
50	10	75	0.933729822	0.892098556
10	10	75	0.907391674	0.891248938
25	10	50	0.932030586	0.886151232
50	10	25	0.916737468	0.885301614
10	10	50	0.919286321	0.870858114
25	10	25	0.924384027	0.860662702
50	50	5	0.962616822	0.845369584
25	50	5	0.953271028	0.83942226
10	10	25	0.91588785	0.838572642
10	50	5	0.925233645	0.816482583
50	10	5	0.862361937	0.809685641
25	10	5	0.831775701	0.798640612

10	10	5	0.800339847	0.714528462

b. I would choose the model with:

- Max features = 10
- Max depth = 50
- N of learners = 200

This is the same model that I had chosen for the other dataset. I also experimented with having 'sqrt' features and no maximum depth, but a smaller number of features and smaller depth still worked best, for reasons discussed above.

10	50	200	0.986406117	0.929481733
25	50	200	0.9855565	0.922684792

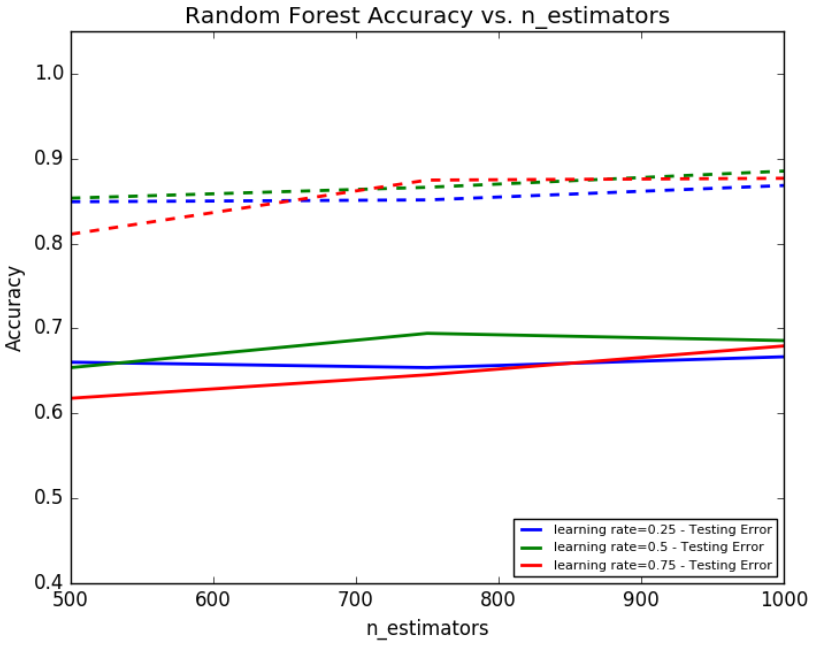
c. There is some slight overfitting with max_features=25, max_depth=50, n_estimators=100 (teal).

AdaBOOST

MNIST

note about this graph: I more parameters than what I listed here (learning rate = 1.0,0.01; n_estimators=200,2000), but since running the program took so long, I only show here the final parameters that I tested that demonstrated the effect of changing parameters.

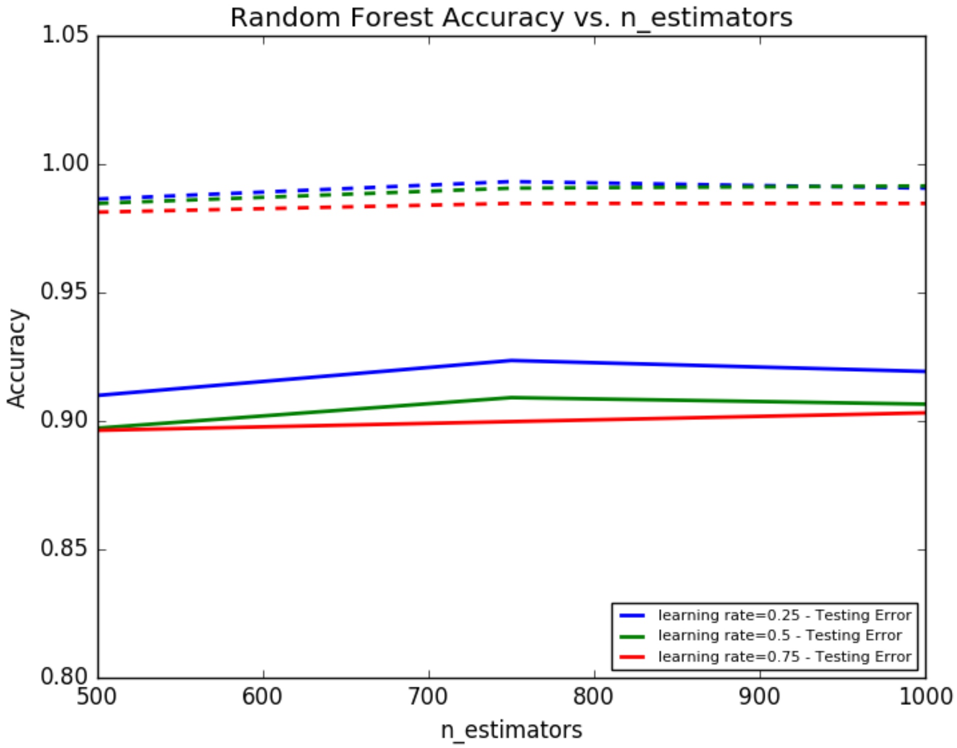
also, should say 'Ada Booster', not 'Random Forest'



There is underfitting with the model learning_rate=0.5 (green) with n_estimators=1000.

learning rate	n_estimators	train accuracy	test accuracy
0.5	750	0.866242038	0.694267516
0.5	1000	0.885350318	0.685774947
0.75	1000	0.876857749	0.67940552
0.25	1000	0.86836518	0.666666667
0.25	500	0.8492569	0.66029724
0.25	750	0.851380042	0.653927813
0.5	500	0.853503185	0.653927813
0.75	750	0.874734607	0.645435244
0.75	500	0.81104034	0.617834395

20NG



learning rate	n_estimators	train accuracy	test accuracy
0.25	750	0.993203059	0.92353441
0.25	1000	0.990654206	0.919286321
0.25	500	0.986406117	0.909940527
0.5	750	0.990654206	0.909090909
0.5	1000	0.991503823	0.906542056
0.75	1000	0.984706882	0.903143585
0.75	750	0.984706882	0.899745115
0.5	500	0.984706882	0.897196262
0.75	500	0.981308411	0.896346644

b. This model performed best with learning rate=0.25 and n_estimators=750. AdaBoost performed consistently better with this dataset than with MNIST. The learning rate affects how different features are weighted. Since there are a large number of features in this dataset, I would guess that having a lower learning rate avoids giving too much weight to certain features while ignoring others.

c. There is some slight overfitting with the learning rate =0.5, n_estimators=1000 (green), just like in the other dataset.

3.

a. Here are the models that I would choose for each dataset:

i. MNIST: Random Forest

max_features	max_depth	n_estimators	train accuracy	test accuracy
10	50	200	0.98089172	0.845010616

ii. 20NG: Random Forest

max_features	max_depth	n_estimators	train accuracy	test accuracy
10	50	200	0.986406117	0.929481733

The choice for the MNIST dataset was easy, as the accuracy was extremely low. However, it was not so easy to pick a model for the 20NG dataset. I had initially decided to pick an AdaBoost classifier, thinking that it was a simpler method that might generalize better. However, the AdaBoost did not generalize to both datasets. It performed very well with the 20NG dataset, but horribly with the MNIST set. The Random Forest classifier performed well with both sets. So I'd feel more comfortable picking a Random Forest and would feel more confident that it would perform well with whatever data I use.