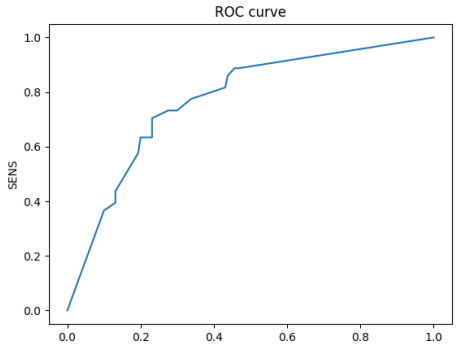
Problem 1. Single decision tree baselines



1) option 1: min_samples_leaf = 5

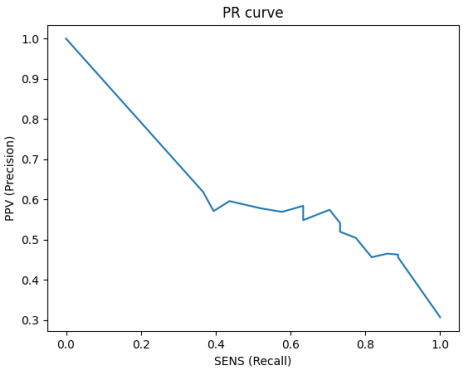
Node count 109

***** Train data stats *****

Train score: 0.88

[[311 29]

[37 160]]



***** Test data stats *****

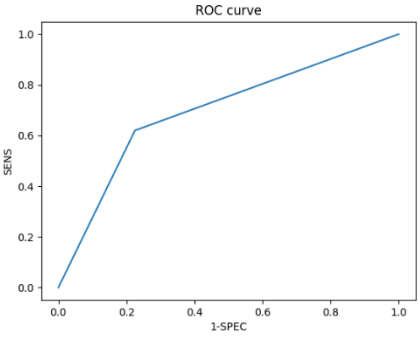
Test score: 0.75

[[123 37]

[21 50]]

AUROC score: 0.77

There is no sign of overfitting or underfitting.
Train accuracy is not low, and there is not a big discrepancy between the train and test accuracies or errors.



2) option 2: full decision tree

Node count 195

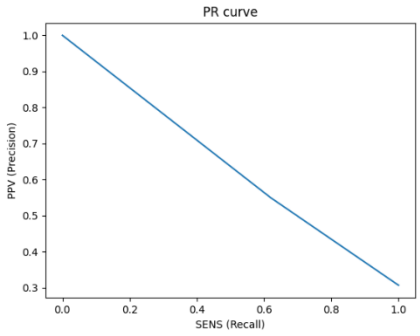
***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]

***** Test data stats *****



Test score: 0.73

[[124 36]

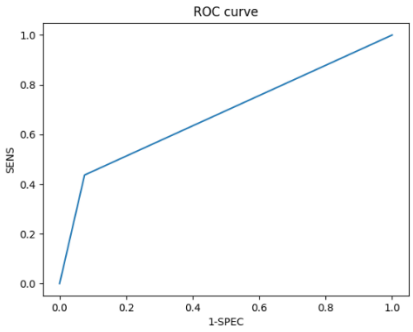
[27 44]]

AUROC score: 0.70

There is no underfitting because the train accuracy is high. But it's too perfect while the test accuracy is not

high enough, we should worry about overfitting here.

3) option 3: one level decision tree



Node count 3

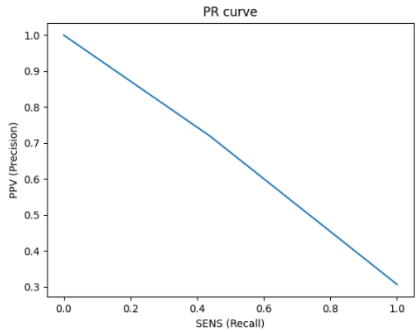
***** Train data stats *****

Train score: 0.74

[[328 12]

[130 67]]

***** Test data stats *****



Test score: 0.77

[[148 12]

[40 31]]

AUROC score: 0.68

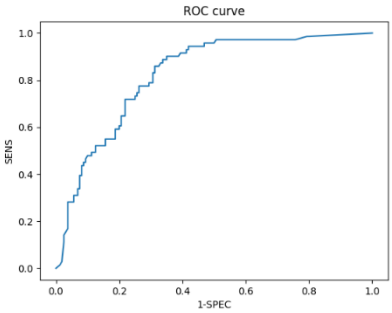
There is no obvious sign of under/overfitting because the train accuracy is not low and there is no big discrepancy between train and test accuracies/errors. But since the test accuracy is even higher than the train accuracy, it may indicate that there may be a meaningful difference between the train and test data.

Compare the 3 models, the option 1 model with min_samples_leaf = 5 has the best performance. The performance rank is option 1 > option 2 > option 3. The PR curves of both option 2 and 3 are almost a line crossing the diagonal, denoting the bad performance.

Problem 2. Bagging

Part a

(option 1)



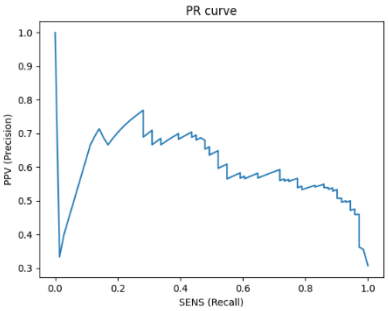
1) 5 classifiers

***** Train data stats *****

Train score: 0.87

[[311 29]

 [42 155]]



***** Test data stats *****

Test score: 0.76

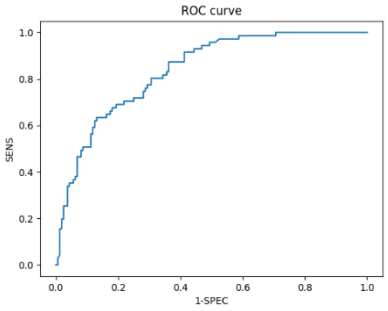
[[125 35]

 [21 50]]

AUROC score: 0.82

Compare with the single tree model for min_samples_leaf = 5, this bagging model improves the performance in both test accuracy (0.76 vs 0.75) and AUROC score (0.82 vs 0.77). Thus, bagging helps improve the performance.

2) 10 classifiers



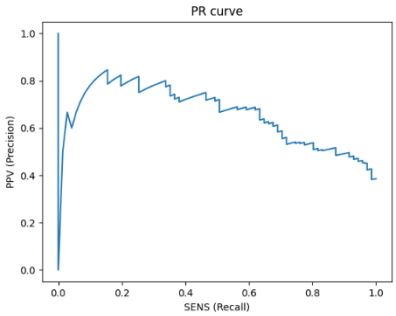
***** Train data stats *****

Train score: 0.90

[[319 21]

 [34 163]]

***** Test data stats *****



Test score: 0.75

[[125 35]

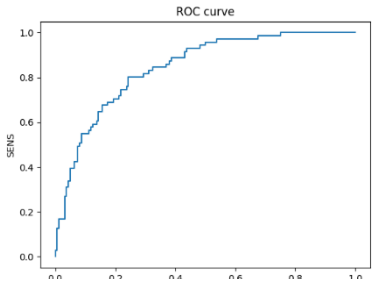
 [22 49]]

AUROC score: 0.84

Compare with the model with 5 classifiers, this one has a higher train accuracy (0.90 vs 0.87) and higher AUROC

(0.84 vs 0.82), and by the PR curve, the area under PR curve for this model is also larger. The test accuracy is similar to the 5-classifier model. But overall, the 10 classifier model has a better performance than the 5-classifier one.

3) 30 classifiers

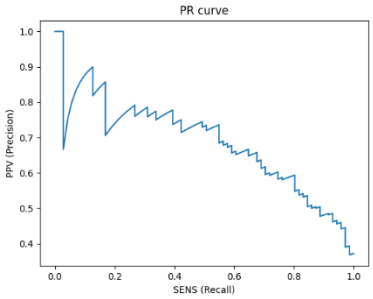


***** Train data stats *****

Train score: 0.92

[[328 12]

 [29 168]]



***** Test data stats *****

Test score: 0.77

[[126 34]

 [20 51]]

AUROC score: 0.84

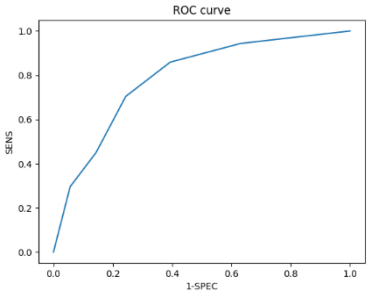
Both train and test accuracies increase, and AUROC score (0.84) stays as the 10-classifier one and is higher than the 5-classifier one.

Compare all 3 models, we can conclude that for option 1 model, bagging helps, and more trees help to improve the model performance.

Part b

(option 2)

1) 5 classifiers

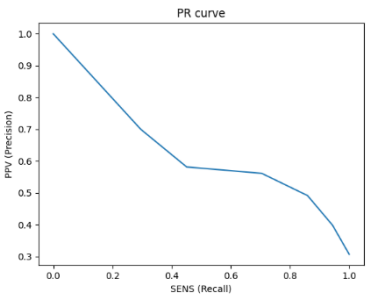


***** Train data stats *****

Train score: 0.97

[[331   9]

 [7 190]]



***** Test data stats *****

Test score: 0.74

[[121 39]

 [21 50]]

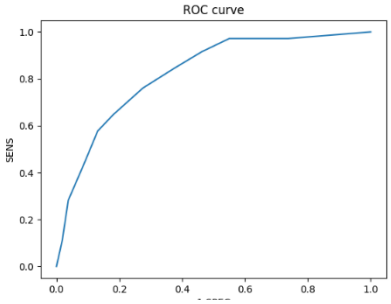
AUROC score: 0.79

Compare with the single tree model for option 2, the

ensemble of 5 classifiers increases both test accuracy and the AUROC score. And the train accuracy is no longer perfect, easing the overfitting problem to some extent.

Why performance improvement:
Bagging works in under/over fitting problems because it combines results of all trained models and decreases variance by averaging. In the single tree option 2 model we have in problem 1, there is a possibility of overfitting. Overfitting is depicted by large variance, thus, bagging can improve the performance.

2) 10 classifiers

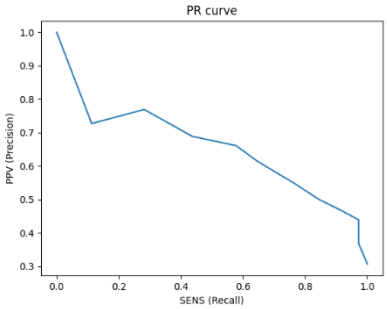


***** Train data stats *****

Train score: 0.99

[[338 2]

[5 192]]



***** Test data stats *****

Test score: 0.77

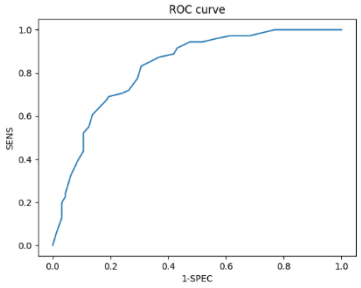
[[131 29]

[25 46]]

AUROC score: 0.82

Both test accuracy and AUROC score increase in this model while the train accuracy is still very high (0.99). But overall still consider the performance is improved.

3) 30 classifiers

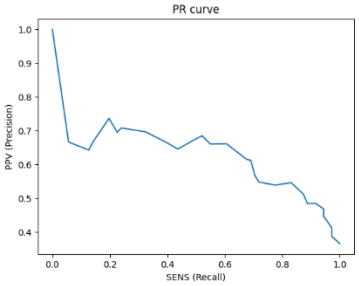


***** Train data stats *****

Train score: 1.00

[[340 0]

[1 196]]



***** Test data stats *****

Test score: 0.77

[[129 31]

[22 49]]

AUROC score: 0.83

This model doesn't have significant improvement in

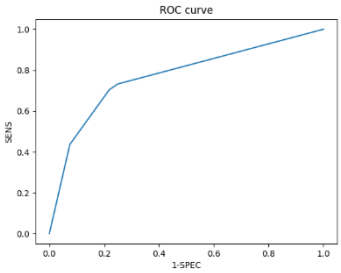
performance, only the AUROC score increase from 0.82 to 0.83. Moreover, the overfitting problem doesn't get improved, even worse than the 10-classifier model.

By these 3 models, we can conclude that for a full decision tree, increasing the number of classifiers in the ensemble may improve the test accuracy and AUROC score, but when there is an overfitting problem, just by increasing the number of classifiers cannot solve the problem.

Part c

(option 3)

1) 5 classifiers

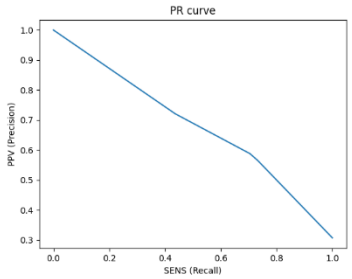


***** Train data stats *****

Train score: 0.73

[[266 74]

 [73 124]]



***** Test data stats *****

Test score: 0.76

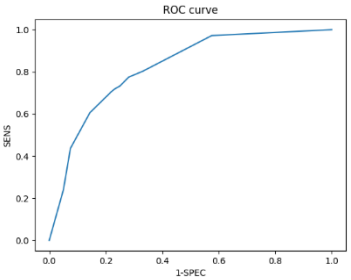
[[125 35]

 [21 50]]

AUROC score: 0.77

Compare with the single tree for option 3 in problem 1, the train and test accuracies even decrease a little in this model, while the AUROC score increases from 0.68 to 0.77. The result kind of makes sense because option 3 model has tree depth 1 and only has 1 split, thus, adding 4 more classifiers and combine the results will not make a large difference.

2) 10 classifiers

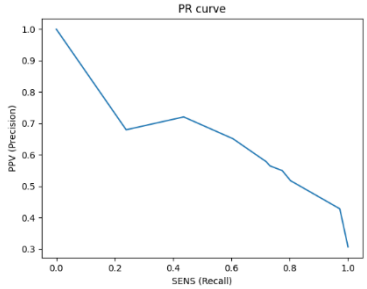


***** Train data stats *****

Train score: 0.75

[[301 39]

 [93 104]]



***** Test data stats *****

Test score: 0.78

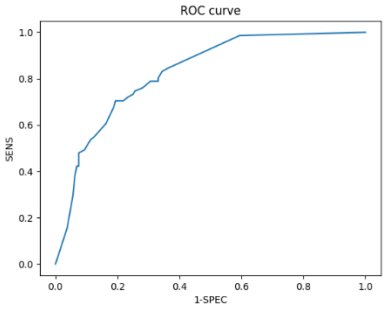
[[137 23]

 [28 43]]

AUROC score: 0.82

Both train and test accuracies and AUROC score improve compared with the option 3 single-tree model and the 5-classifier model. This is reasonable since we gave the model more classifiers for a simpler model with the tree depth is 1 and only has 1 split.

3) 30 classifiers



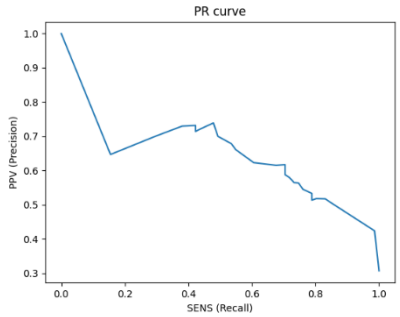
***** Train data stats *****

Train score: 0.73

[[317 23]

[122 75]]

***** Test data stats *****



Test score: 0.78

[[145 15]

[36 35]]

AUROC score: 0.83

Compare with previous 2 models and the single-tree one, the 30-classifier model has higher AUROC score, and the test accuracies is one of the highest among the models for option 3. The train accuracy drops a little but not significantly. We may interpret this as that for option 3, increasing the number of classifiers may improve the model performance a little, but it's not linear: after some number of classifiers, the performance may remain similar, decrease a little, or show very little improvement.

Part d

	Option1: min_samples_leaf = 5	Option 2: Full decision tree	Option 3: max_depth = 1
Does bagging help?	Yes, improves the performance	Ease overfitting a little	Limited improvement
Does more trees help?	Yes, more trees = better performance	Improves accuracies but doesn't solve overfitting	Depends on the number of trees

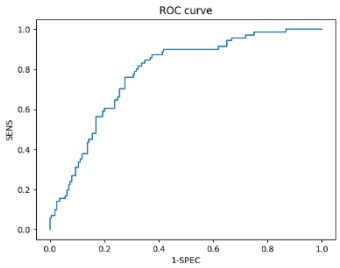
Overall, bagging can improve the model's performance by combining multiple classifiers and deals with the overfitting problem when the number of classifiers is appropriate.

Problem 3. Boosting: Adaboost

Part a

(option 1)

1) 5 classifiers

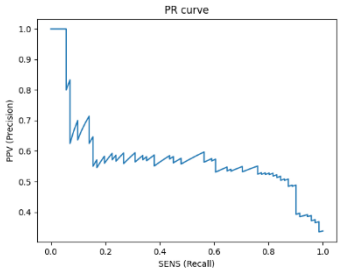


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.73

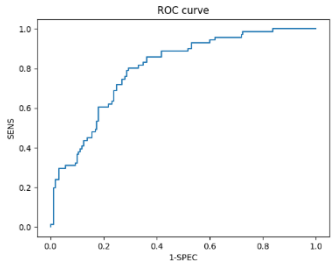
[[122 38]

[25 46]]

AUROC score: 0.78

Compare with the single-tree model, although the AUROC score increases a little (from 0.77 to 0.78), the test accuracy drops (from 0.75 to 0.73) and the overfitting problem appears. Thus, overall, we say Adaboost didn't help to improve the model performance.

2) 10 classifiers

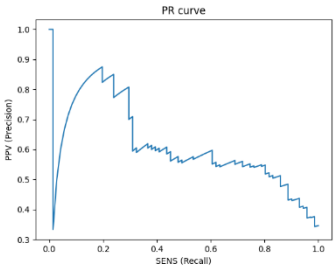


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.73

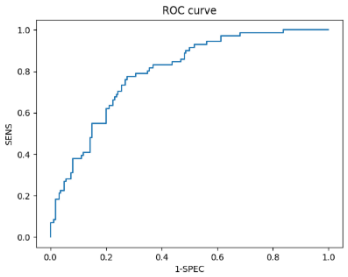
[[122 38]

[25 46]]

AUROC score: 0.80

Still, the AUROC score increases a little but test accuracy remains 0.73 and the overfitting problem didn't get resolved. So in this case when increasing the number of trees from 5 to 10, Adaboost didn't help to improve the performance.

3) 30 classifiers

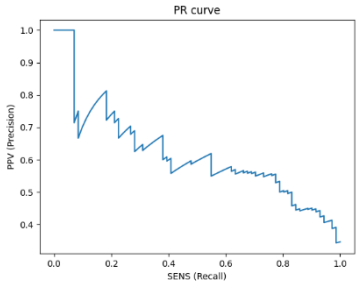


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.74

[[123 37]

[24 47]]

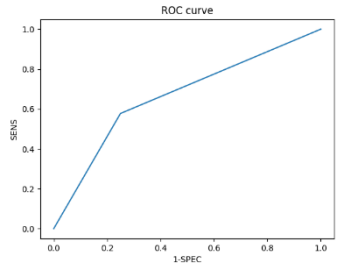
AUROC score: 0.80

Compare with the previous 2 models (5 trees and 10 trees), test accuracy improves a little (from 0.73 to 0.74) but still lower than the single-tree model. And the AUROC score remains the same as the 10-tree one. But the overfitting problem is still there. Thus, when increasing the number of trees to 30, Adaboost still didn't help to improve the model performance.

Part b

(option 2)

1) 5 classifiers

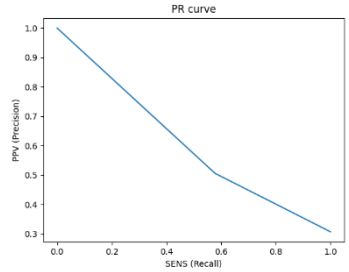


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.70

[[120 40]

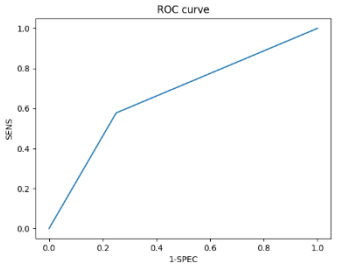
[30 41]]

AUROC score: 0.66

Compare with the single-tree model for option 2, both test accuracy and AUROC score drop and the overfitting problem still exists, thus the performance didn't get improved. This can possibly be that we already have a perfect fit with accuracy 1 in the single tree model, i.e., no samples got

incorrectly classified. So Adaboost cannot effectively cover misclassified samples (we have none) and didn't show an improvement.

2) 10 classifiers

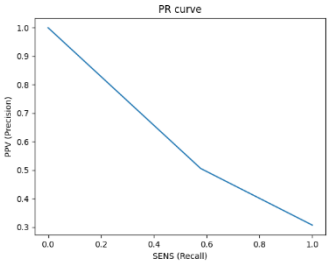


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.70

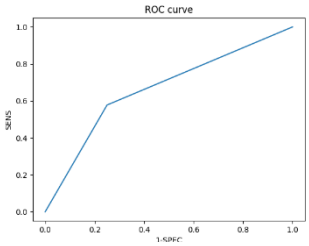
[[120 40]

[30 41]]

AUROC score: 0.66

No performance improvement: the results stay the same as the 5-classifier one.

3) 30 classifiers

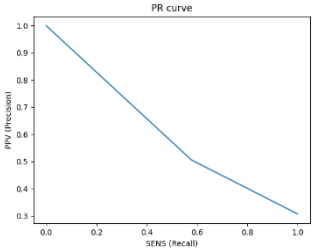


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.70

[[120 40]

[30 41]]

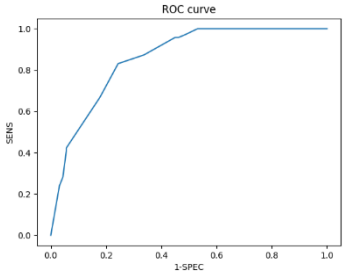
AUROC score: 0.66

Still no performance improvement when increasing the number of classifiers to 30.

Part c

(option 3)

1) 5 classifiers

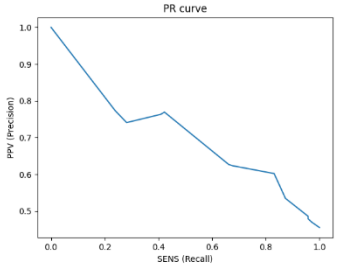


***** Train data stats *****

Train score: 0.78

[[297 43]

 [74 123]]



***** Test data stats *****

Test score: 0.77

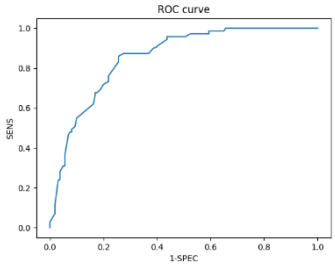
[[131 29]

 [23 48]]

AUROC score: 0.86

Compare with the single-tree model, both the train accuracy and the AUROC score increase, by which now the train accuracy is higher than the test accuracy and makes more sense. The increase in train accuracy makes sense because in Adaboost the samples got misclassified earlier will get a higher weight in later trainings.

2) 10 classifiers

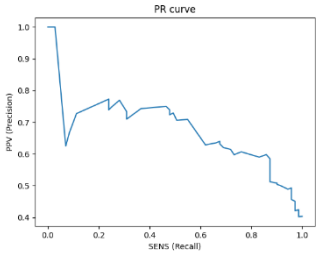


***** Train data stats *****

Train score: 0.79

[[290 50]

 [64 133]]



***** Test data stats *****

Test score: 0.77

[[128 32]

 [20 51]]

AUROC score: 0.85

The test accuracy stays the same as the 5-classifier model and the AUROC score drops a little from 0.86 to 0.85, but is still higher than the single-tree model AUROC score. The train accuracy also increases in this case due to the same reason of the intrinsic nature of Adaboost as stated in 1).

3) 30 classifiers

***** Train data stats *****

Train score: 0.82

[[303 37]

 [60 137]]

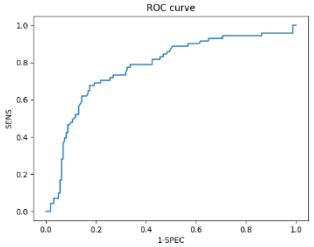
***** Test data stats *****

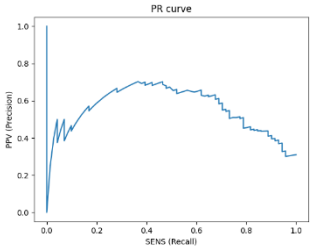
Test score: 0.78

[[132 28]

 [23 48]]

AUROC score: 0.78





Both the train and test accuracies improve in this case while the AUROC score drops compared with 5-classifier and 10-classifier models, but still higher than the single-tree model. The increase in accuracies make sense due to the intrinsic nature of Adaboost as stated in 1).

Part d

	Option1: min_samples_leaf = 5	Option 2: Full decision tree	Option 3: max_depth = 1
Does Adaboost help?	No, even cause overfitting	No, cannot deal with existing overfitting	Yes, improves the performance
Does more trees help?	No	No	Yes, overall more trees improve accuracies

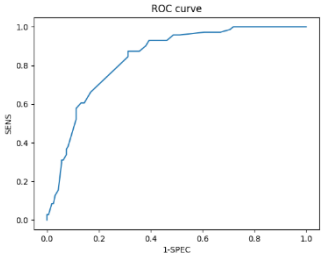
Adaboost can improve train accuracy and test accuracy and is more effective when the tree depth is smaller. But when there is an overfitting, Adaboost cannot deal with the problem effectively. Compare with bagging, bagging can deal with the overfitting to some extent and shows improvement in performance in all 3 models.

Problem 4. Boosting: GradientBoosting

Part a

(option 1)

1) 5 classifiers

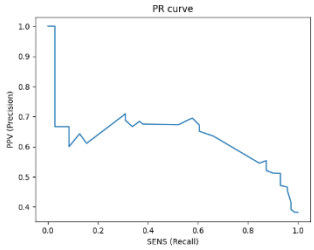


***** Train data stats *****

Train score: 0.74

[[336 4]

[134 63]]



***** Test data stats *****

Test score: 0.74

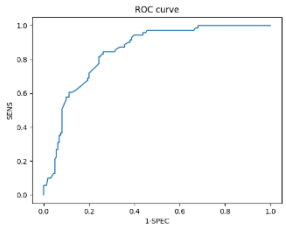
[[148 12]

[47 24]]

AUROC score: 0.84

Compare with the single-tree model, both train and test accuracies drop even though the AUROC score increases. So overall we say Gradient boosting didn't help to improve the model performance.

2) 10 classifiers

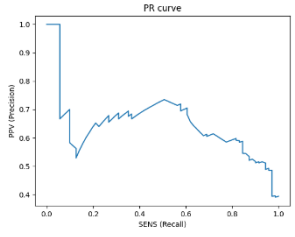


***** Train data stats *****

Train score: 0.79

[[318 22]

[92 105]]



***** Test data stats *****

Test score: 0.79

[[142 18]

[30 41]]

AUROC score: 0.85

Compare with the previous 5-classifier model, this 10-classifier one shows performance in train, test accuracies and AUROC score. The train accuracy is still lower than the single-tree model, but the test accuracy and AUROC score are both higher. So we consider this model improves the model performance.

	Gradient boosting (n_estimator = 10)	Bagging (n_estimator = 10)	Adaboost (n_estimator = 10)
Train accuracy	0.79	0.90	1.00
Test accuracy	0.79	0.75	0.73
AUROC score	0.85	0.84	0.80

For 10-tree ensemble in option 1, Gradient boosting model has the best test accuracy and AUROC score. Although Bagging model has a better train accuracy, its test accuracy is lower. And the Adaboost model has the potential of overfitting problems. Thus we consider Gradient boosting model has a better performance among the three.

3) 30 classifiers

***** Train data stats *****

Train score: 0.86

[[313 27]

 [50 147]]

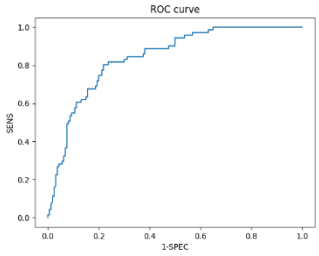
***** Test data stats *****

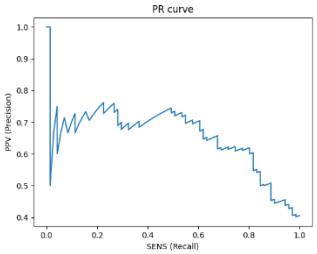
Test score: 0.77

[[130 30]

 [23 48]]

AUROC score: 0.84





Although train accuracy is higher than the 10-classifer model, test score drops as well as AUROC score. So increasing the number of trees from 10 to 30 didn't see an improved performance. Compare with the single-tree and 5-classifier model, the test accuracy in this model is higher than both of them, so this is an improvement.

	Gradient boosting (n_estimator = 30)	Bagging (n_estimator = 30)	Adaboost (n_estimator = 30)
Train accuracy	0.86	0.92	1.00
Test accuracy	0.77	0.77	0.74
AUROC score	0.84	0.84	0.80

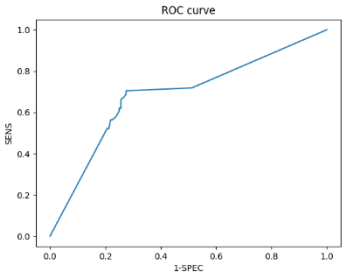
When n_estimator is 30, Gradient boosting and bagging show the same test accuracy and AUROC score while the train accuracy of Gradient boosting is a little lower. And the Adaboost

model still has potential overfitting problem. Overall, Bagging has a better performance among the three.

Part b

(option 2)

1) 5 classifiers

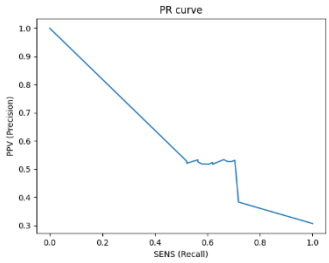


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.71

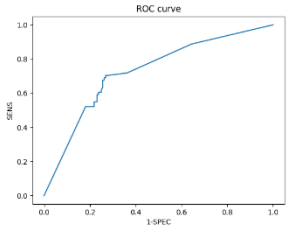
[[122 38]

[30 41]]

AUROC score: 0.68

Compare with single-tree model (train score 1, test score 0.73, auroc score 0.7), the Gradient model here has a worse performance with lower test accuracy and AUROC score and the potential overfitting problem still exists. So the ensemble didn't help to improve the performance of option 2 baseline. I think this is because we already have overfitting problem in the baseline model. Similar to Adaboost, Gradient boosting is trying to fix the limitation of the previous models sequentially. When we already have the overfitting problem, the model has limited ability to improve the performance.

2) 10 classifiers

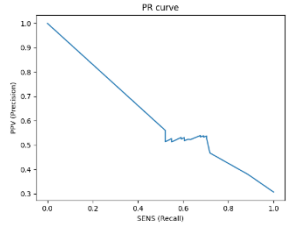


***** Train data stats *****

Train score: 1.00

[[340 0]

[0 197]]



***** Test data stats *****

Test score: 0.71

[[121 39]

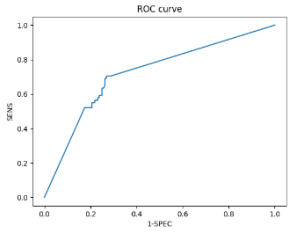
[28 43]]

AUROC score: 0.73

Although the AUROC score is a little higher, there is no improvement in test accuracy and the overfitting problem compared with the 5-classifier mode. And still, the test accuracy is lower than then single-tree model. So we say increasing the number of trees from 5 to 10 didn't help to improve the model performance.

3) 30 classifiers

***** Train data stats *****

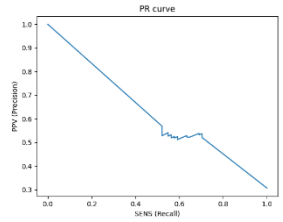


Train score: 1.00

[[340 0]

[0 197]]

***** Test data stats *****



Test score: 0.71

[[120 40]

[28 43]]

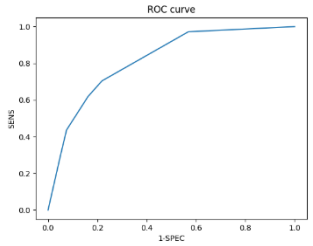
AUROC score: 0.72

The auroc score drops compared with the 10-classifier one while the train and test accuracy remain the same. The test accuracy is still lower than the single-tree model. Overall, 30-tree ensemble still didn't improve the model performance.

Part c

(option 3)

1) 5 classifiers



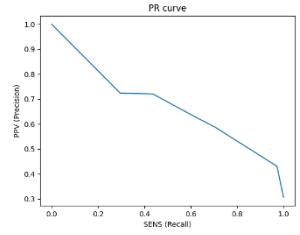
***** Train data stats *****

Train score: 0.71

[[331 9]

[145 52]]

***** Test data stats *****



Test score: 0.75

[[152 8]

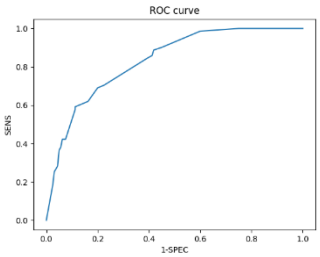
[50 21]]

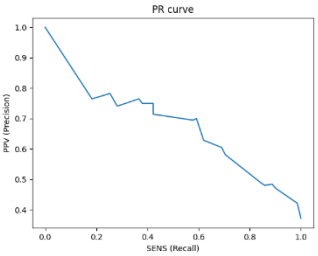
AUROC score: 0.82

Compare with the single-tree model baseline (train score 0.74, test score 0.77, auroc score 0.68), this model has lower train and test accuracies even though the auroc score is higher. So we

consider the Gradient boosting didn't help to improve the performance. This kind of make sense because option 3 models only have 1 split and the tree depth is 1. Gradient boosting uses the gradient of the loss function to learn the new model, so in this case, combining only 5 classifiers with depth 1 and 1 split constrain the models' ability.

2) 10 classifiers





***** Train data stats *****

Train score: 0.74

[[329 11]

[130 67]]

***** Test data stats *****

Test score: 0.78

[[150 10]

[41 30]]

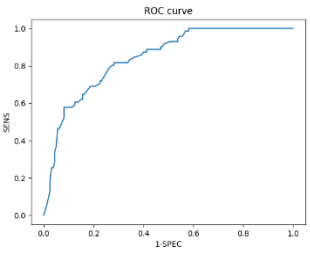
AUROC score: 0.83

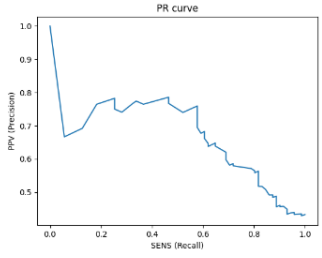
Compare with the 5-classifier model, all metrics improved. Compare with the single-tree model, train accuracy is the same while the test accuracy and auroc score are higher. So we consider this as a model performance improvement.

	Gradient boosting (n_estimator = 10)	Bagging (n_estimator = 10)	Adaboost (n_estimator = 10)
Train accuracy	0.74	0.75	0.79
Test accuracy	0.78	0.78	0.77
AUROC score	0.83	0.82	0.85

Among the three models for n_estimator = 10 in option 3, Adaboost model has better train accuracy and AUROC score while the test accuracy is only a little fewer than other 2 models. So we consider Adaboost model has a better performance among the three.

3) 30 classifiers





***** Train data stats *****

Train score: 0.77

[[316 24]

[97 100]]

***** Test data stats *****

Test score: 0.80

[[143 17]

 [30 41]]

AUROC score: 0.84

Compare with the 10-classifier model, all metrics improved in this model, and is also higher than the single-tree baseline. So increasing the number of trees to 30 shows an improved performance.

	Gradient boosting (n_estimator = 30)	Bagging (n_estimator = 30)	Adaboost (n_estimator = 30)
Train accuracy	0.77	0.73	0.82
Test accuracy	0.80	0.78	0.78
AUROC score	0.84	0.83	0.78

Among the three models for n_estimators = 30, although Adaboost model has a higher train accuracy, Gradient boosting has higher test accuracy and AUROC score among all three. Overall we consider the Gradient boosting model has a better performance among the three.

Part d

	Option1: min_samples_leaf = 5	Option 2: Full decision tree	Option 3: max_depth = 1
Does Gradient boosting help?	No	No, cannot deal with existing overfitting	Yes, improves the performance
Does more trees help?	Yes, but doesn't mean more trees the better	No	Yes, more trees improve accuracies

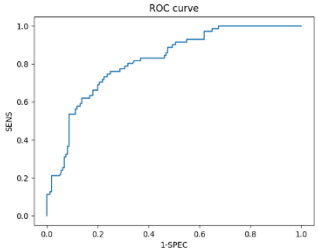
Gradient boosting can improve train accuracy and test accuracy and generally has a better performance when the number of trees increase. Compare with Bagging and Adaboost, Bagging can deal with the overfitting problem to some extent, and Adaboost cannot do it either. Bagging has a better performance in option 1 models and Adaboost and Gradient boosting perform better in option 3 models.

Problem 5. Random Forests

Part a

(option 1)

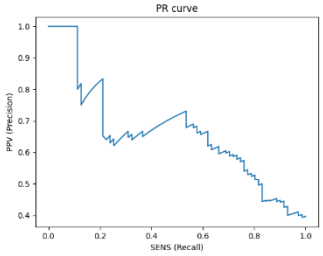
1) 5 classifiers



***** Train data stats *****

Train score: 0.85

[[313 27]
[52 145]]



***** Test data stats *****

Test score: 0.77

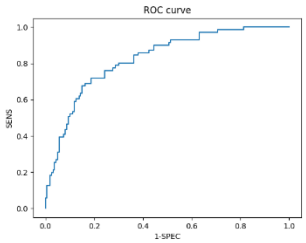
[[131 29]

 [24 47]]

AUROC score: 0.82

Compare with the single-tree classifier for option 1 (train score 0.88, test score 0.75, AUROC score 0.77), the 5-classifier model has improved test accuracy and AUROC score. So overall we consider Random Forest 5-classifier model helps to improve model performance.

2) 10 classifiers

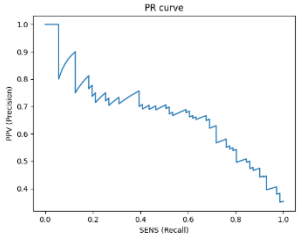


***** Train data stats *****

Train score: 0.87

[[314 26]

 [44 153]]



***** Test data stats *****

Test score: 0.78

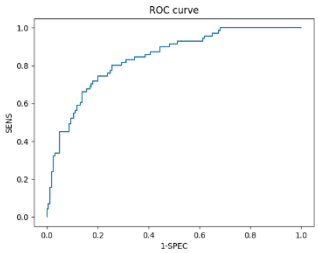
[[132 28]

 [22 49]]

AUROC score: 0.83

All metrics improve compare with the 5-classifier model, and both test accuracy and AUROC score is higher than the single-tree model. So we say that Random Forest classifier helps, and increasing the number of trees from 5 to 10 improves the model performance.

3) 30 classifiers

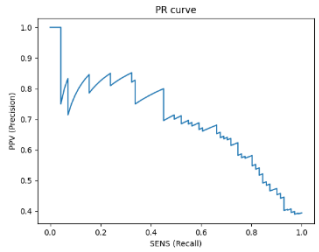


***** Train data stats *****

Train score: 0.90

[[318 22]

 [32 165]]



***** Test data stats *****

Test score: 0.78

[[132 28]

 [22 49]]

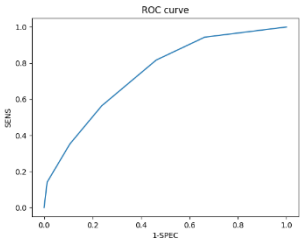
AUROC score: 0.84

Both train accuracy and AUROC score improve while the test accuracy remains the same based on the 10-classifier model. So we say that increasing the number of trees to 30 still improves the performance, just the improvement is limited in test accuracy.

Part b

(option 2)

1) 5 classifiers

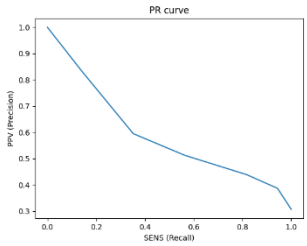


***** Train data stats *****

Train score: 0.98

[[335   5]

 [8 189]]



***** Test data stats *****

Test score: 0.70

[[122 38]

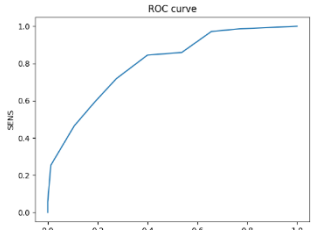
 [31 40]]

AUROC score: 0.74

Compare with the single-tree model for option 2 (train score 1.0, test score 0.73, AUROC score 0.70), this model has a lower test score while the AUROC score is a little higher, and the train score is not perfect now (kind of reduce the overfitting). So the 5-classifier ensemble only helps improve the model performance a little. Random Forest helps to ease the overfitting a little in our model (it's not 1 anymore at least) because it takes random selection of a subset of features to

calculate the output and combines the individual outputs, so this randomness and combination may help with overfitting a little.

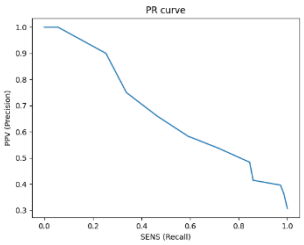
2) 10 classifiers



***** Train data stats *****

Train score: 0.99

[[338 2]
 [4 193]]



***** Test data stats *****

Test score: 0.74

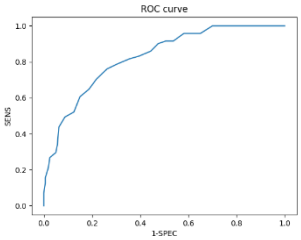
[[130 30]

 [29 42]]

AUROC score: 0.80

Compare with the single-tree model, both test score and AUROC score increase and train score is not perfect 1. Compare with the 5-classifier model, both test score and AUROC score increase. So increasing the number of trees to 10 increases the model performance more.

3) 30 classifiers

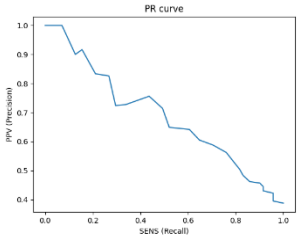


***** Train data stats *****

Train score: 1.00

[[340 0]

 [1 196]]



***** Test data stats *****

Test score: 0.76

[[130 30]

 [25 46]]

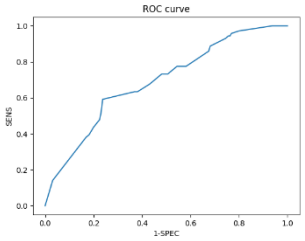
AUROC score: 0.82

Compare with previous model, although the test accuracy and AUROC score of this model keeps increasing, the train score is perfect again. So we may say that Random Forest helps to improve the test accuracy but the ability to deal with overfitting still depends on the number of trees.

Part c

(option 3)

1) 5 classifiers

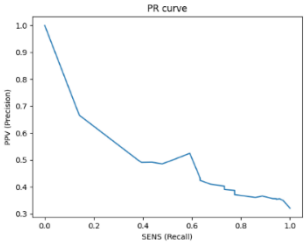


***** Train data stats *****

Train score: 0.63

[[340 0]

[197 0]]



***** Test data stats *****

Test score: 0.69

[[160 0]

[71 0]]

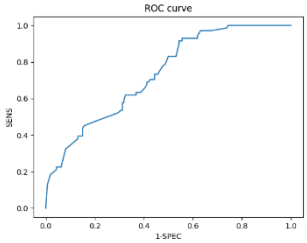
AUROC score: 0.69

Compare with the single-tree model (train score 0.74, test score 0.77, AUROC score 0.68) with this 5-classifier model, only the AUROC score increases a little bit while both train and test accuracies are lower. So for option 3 models, Random Forest classifier didn't help to improve the performance in 5-classifier model. That's probably because we only have 1 split in option 3 model, and the power of Random Forest comes from random input/feature subsets generated at every split. So this ability is constrained here when we only has 5 classifiers.

	Random Forest (n_estimator = 5)	Gradient boosting (n_estimator = 5)	Bagging (n_estimator = 5)	Adaboost (n_estimator = 5)
Train accuracy	0.63	0.71	0.73	0.78
Test accuracy	0.69	0.75	0.76	0.77
AUROC score	0.69	0.82	0.77	0.86

Compare with previous models for n_estimator = 5 in option 3, all 3 metrics of the Random Forest model are lower than others. Among them, Random Forest has the poorest performance while the Adaboost one has the best performance.

2) 10 classifiers



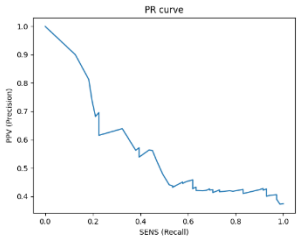
***** Train data stats *****

Train score: 0.71

[[321 19]

[138 59]]

***** Test data stats *****



Test score: 0.72

[[150 10]

[55 16]]

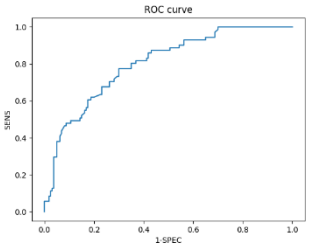
AUROC score: 0.73

Although the train and test accuracies are still lower than the single-tree model (while AUROC score is higher), all metrics have improvement based on the 5-classifier model. So we can see that when increasing the number of trees from 5 to 10 here, model performance improves.

	Random Forest (n_estimator = 10)	Gradient boosting (n_estimator = 10)	Bagging (n_estimator = 10)	Adaboost (n_estimator = 10)
Train accuracy	0.71	0.74	0.75	0.79
Test accuracy	0.72	0.78	0.78	0.77
AUROC score	0.73	0.83	0.82	0.85

Still, when n_estimator=10, the Random Forest model has lower train and test accuracies and the AUROC score, thus has the poorest performance among the three.

3) 30 classifiers



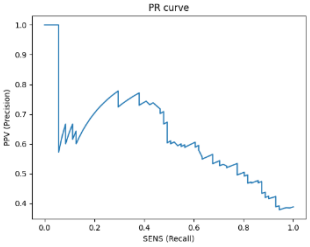
***** Train data stats *****

Train score: 0.70

[[335   5]

[157 40]]

***** Test data stats *****



Test score: 0.71

[[154   6]

[60 11]]

AUROC score: 0.80

Although AUROC score increases, both train and test accuracies are lower than the 10-classifier model and single-tree while are higher than the 5-classifier one. So we can see that in Random Forest for option 3, increasing the number of trees doesn't mean that the performance will definitely improve.

	Random Forest (n_estimator = 30)	Gradient boosting (n_estimator = 30)	Bagging (n_estimator = 30)	Adaboost (n_estimator = 30)
Train accuracy	0.70	0.77	0.73	0.82
Test accuracy	0.71	0.80	0.78	0.78
AUROC score	0.80	0.84	0.83	0.78

Same as above, all metrics of the Random Forest model have the lowest values among the three, thus it has the worst performance.

Part d

	Option1: min_samples_leaf = 5	Option 2: Full decision tree	Option 3: max_depth = 1
Does Random Forest help?	Yes	Yes, but if overfitting depends on the number of trees	No
Does more trees help?	Yes, but the improvement in test accuracy can be limited	Yes, but may not help with overfitting	Overall yes, but doesn't mean more trees the better

Random Forest classifier helps improve the model performance in option1 and 2 and can help to ease the overfitting problem a little, but is not very helpful when the model only has 1 split (option 3). But compare with previous classifiers, Random Forest classifier has worse performance in all 3 options.