**HW4**

Han Wang, Zhenbang Guo, Zhihan Li

**Q1**

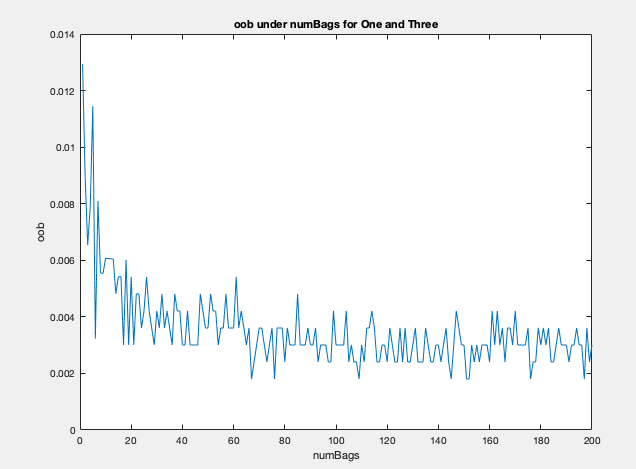
1. Bagged trees implementation results are pasted below.
2. One\_three\_five experiments results are showing below.

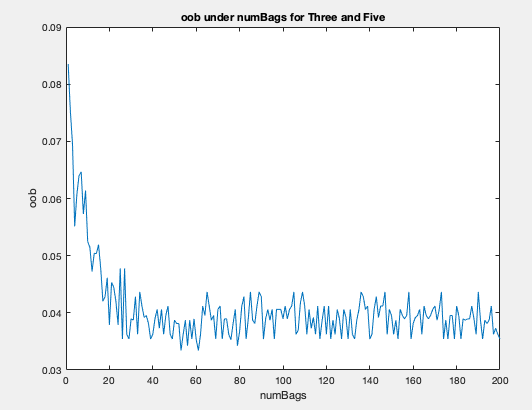
It is a general trend that as number of bags increase, the out of bag error decreases, and converges to a narrow results region. Out of bag error is a technique to estimate the . In this case, as we increase the number of bags, it decreases the chance of model overfitting. The decreasing rate of oob error slows down significantly after numBags equals to 20.

Another observation is that it is more difficult to classify 3\_5 problem compared to 1\_3 problem. Bagged trees and trees trained with cross validation are subject to larger error measurement in 3\_5 problem. An intuition for this is that 3&5 are much more similar to each other than 1&3.

The output from in-built in CV command and bagging result is the following :

|  |  |  |
| --- | --- | --- |
|  | Cross Validation Error | Out of Bag error |
| 1\_vs\_3 | 0.0121 | 0.0040 |
| 3\_vs\_5 | 0.0528 | 0.040 |





1. Now we are comparing test error by a single decision tree model and a single ensemble of 200 bagged trees.

|  |  |  |
| --- | --- | --- |
| Test error | Single decision tree | 200 ensemble bagged trees |
| 1\_vs\_3 | 0.0162 | 0.0096 |
| 3\_vs\_5 | 0.1197 | 0.0616 |

1. For each tree in the tree ensemble (bagging), training data are randomly sampled from the whole training dataset with replacement. This “random selection”, in practical sense, is helpful to reduce sampling bias (similar to random forest). Therefore the data points not sampled for training are called out-of-bag observations. The out-of-bag error (OOB) computes the misclassification probability (for classification trees) for out-of-bag observations in the training data1. OOB is an estimate of the prediction error of the bagged trees. In theory, we know bagged trees overestimates test error. By comparing tables in part (b) and part (c), oob error for both 1\_vs\_3 and 3\_vs\_5 are both higher than test error.

In our training results, an observable trend is that when the number of bags increases, the performance of the bagged trees becomes better (smaller OOB). As shown in the graph *1-vs-3 problem*, the OOB decreases from ~0.013 and converges to around 0.004. As for the graph *3-vs-5 problem*, the OOB decreases from ~0.85 and converges to around 0.040. Unlike 1\_vs\_3, 3\_vs\_5 has OOB error more close to CV error using only training dataset. This is most likely because 1 is more similar to 3, which means under same model assumption, 1\_vs\_3 is more likely to be overfitted compare to 3\_vs\_5. As we increase the number of trees, it decreases the chance of overfitting 1\_vs\_3 bagged trees more dramatically, while a single decision tree is very likely to over fit the training data. Therefore we see a huge drop in 1\_vs\_3 bagged trees compared to CV method. In conclusion, bagging decrease overfitting and have a better performance than single decision trees.

**Q2**

Adaptive boosting, aka AdaBoost, is a learning algorithm that focuses on classification problems and aims to convert a set of weak classifiers into a strong one.

In the One-vs-Three problem, AdaBoost quickly (less than 10 iteration) learns the classifier that achieve 100% correctness for the training set. Shown in the plot below, the error rate in training data decreases as the number of weak hypothesis increases – because previous misclassified data points are now associated with higher weight. The performance of the classifier on the testing data has a similar trend on the plot curve, the error rate levels out to ~0.012, which is very close to the result we obtained in Question 1.

In the Three-vs-Five problem, about 90 iterations the classifier achieves almost 0% misclassification. Shown in the figures, when the number of weak hypothesis grows, the error rate in training sample decreases – likewise because previous misclassified data points are now associated with higher weights. The performance of the classifier on the testing data is similar – error rate decreases as the number of hypothesis increases – but the error rate begins to level out earlier – starting from twenty weak hypothesis, the error rate fluctuates around 0.85, which is very close to the result we obtained in problem 1. Similar reason to why we observe higher test error in 3\_vs\_5 than 1\_vs\_3. Since 1 is more similar to 3 than 5 to 3, the classifier will perform better on 1\_vs\_3 task, in terms of both error rate and convergence time.

