Random Forrest

(50 points) The purpose of this problem is to write code for bagging decision trees and computing the out-of-bag error. You may use matlab's inbuilt `fitctree` function, which learns decision trees using the CART algorithm (read the documentation carefully), but do not use the inbuilt functions for producing bagged ensembles. In order to do this, you should complete the stub `BaggedTrees` function. Note that it only returns the out-of-bag error. You may want to use other functions that actually construct and maintain the ensemble. You may assume that all the **x** vectors in the input are vectors of real numbers, and there are no categorical variables/features. You will compare the performance of the bagging method with plain decision trees on the handwritten digit recognition problem (the dataset is in `zip.train` and `zip.test`, available from `http://amlbook.com/support.html`. [1])

We will focus on two specific problems – distinguishing between the digit one and the digit three, and distinguishing between the digit three and the digit five.

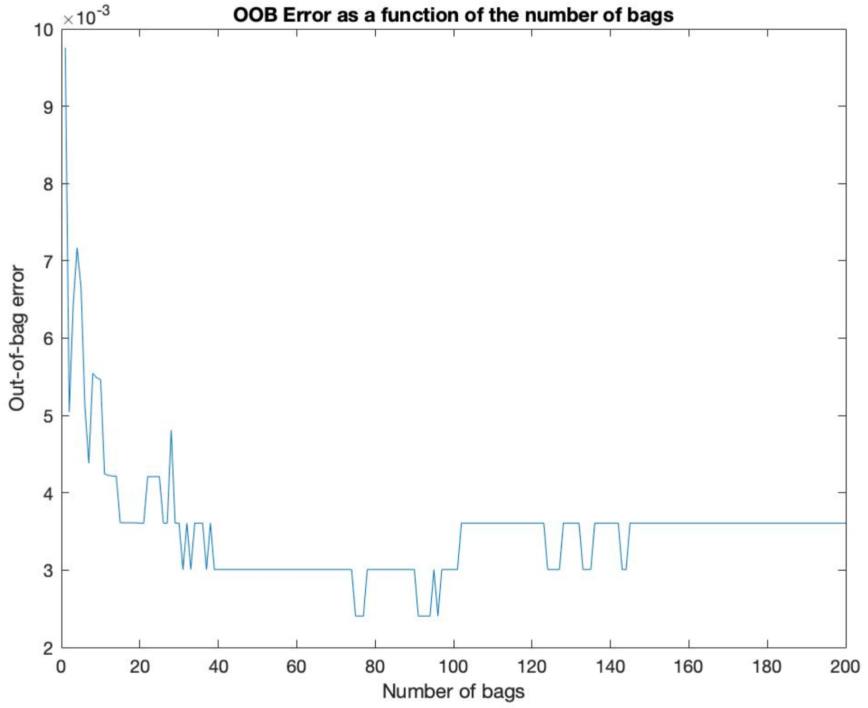
a Complete the implementation of `BaggedTrees`. You may choose any reasonable representation that you wish; the two strict requirements are that you plot the out-of-bag error as a function of the number of bags from 1 to the number specified as input (`numBags`), and that you return the out-of-bag error for the whole ensemble of `numBags` trees. Include the plots (with clearly labeled axes) in your writeup, and, of course, submit your code.

b Run the provided `OneThreeFive` script, which creates training datasets based on the one-vs-three and three-vs-five cases we are interested in, and calls both the in-built decision tree routine and your bagging code, printing out the cross-validation error for decision trees and the OOB error for your bagging implementation. Report the results in your writeup.

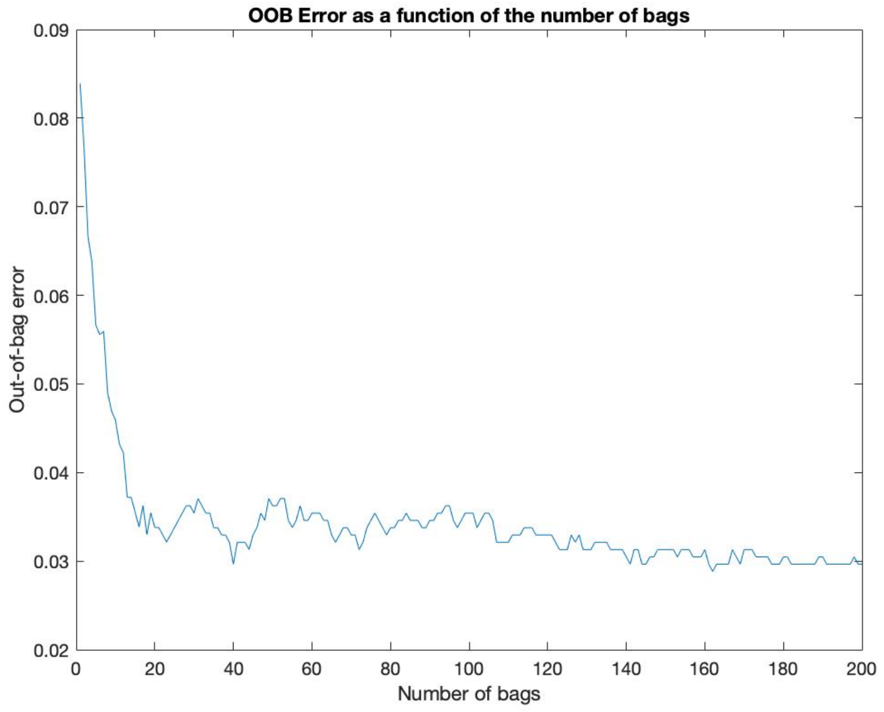
c Now, learn a single decision tree model for each of the two specified problems (one-vs-three and three-vs-five) on the training data, and test their performance on `zip.test` what is the test error? Similarly, learn a single ensemble of 200 trees on the training data for each of the two specified problems and test the performance of the ensembles on the test data. Report your results.

d Summarize and interpret your results in one or two concise paragraphs as part of your writeup.

(a). OOB Error as a function of the number of bags (from 1 to 200) for one-vs-three cases



OOB Error as a function of the number of bags (from 1 to 200) for three-vs-five cases



1(b).
```
>> OneThreeFive
Working on the one-vs-three problem...

The cross-validation error of decision trees is 0.0090
The OOB error of 200 bagged decision trees is 0.0036

Now working on the three-vs-five problem...

The cross-validation error of decision trees is 0.0692
The OOB error of 200 bagged decision trees is 0.0346
```

1(c).
```
>> TestError
Working on the one-vs-three problem...

The test error of a single decision tree is 0.0163
The test error of an ensemble of 200 decision trees is 0.0116

Working on the three-vs-five problem...

The test error of a single decision tree is 0.1196
The test error of an ensemble of 200 decision trees is 0.0859
```

1(d).
From part a, we can observe that when the number of bags increases, the out of bag error decreases. When the number of bags reach a threshold, like 50 in our example, the out of bag error basically remains unchanged, or fluctuates around a certain value. We can conclude that when we have more bags, the performance will be better initially, and when we have enough number of bags, the performance will converge finally.

From part b, we can observe that for both cases (one-vs-three and three-vs-five), the out of bag errors of the random forest are less than the cross-validation errors of a single decision tree. From part c, we can observe that when we apply the trained model to the training data set, the test errors of an ensemble of 200 decision trees are also less than those of a single decision tree. Therefore, in both cases, we can conclude that the performance of a random forest with 200 bootstrapping datasets each associated with a max-depth tree is better than that of a single max-depth decision tree. With 200 decision trees and taking the aggregation of them, we can significantly reduce the high variance of a single decision tree, while maintain the low bias. It is reasonable that a random forest performs better.

Moreover, if we compare part b and part c, we can observe that for a single decision tree, the cross-validation error is less than the test error; for the random forest, the out of bag error is also less than the test error. This is also reasonable, because validation error/oob error are more optimistic than the actual test error.