University of Washington Bothell
Computing & Software Systems

Course title:	Introduction to Machine Learning
Course number:	CSS 581
Term:	Winter 2014
Instructor:	Jeff Howbert

Project 3

Date assigned:	Thur. Feb. 27, 2014
Date due:	Wed. Mar. 19, 2014

There are two parts to the project.

This project is focused on ensemble learning.

Part 1. [20 points] For this Part, you will write a MATLAB script that trains and uses an ensemble of regression trees on the prostate dataset.

For your base regressor, use a standard MATLAB regression tree with default settings, e.g.:

```
minparent = 10;      % default value = 10
minleaf = 1;         % default value = 1

tr = classregtree( trainData, trainResponse, 'method', 'regression', ...
        'minparent', minparent, 'minleaf', minleaf );
testPred = eval( tr, testData );
```

Requirements:

- First train a single tree on all the training data, make predictions on the test set using the tree, and report the mean squared error (MSE) on the test set to the console. See `matlab_demo_08.m` if you need a reminder on how to calculate MSE.

- Generate 8 ensembles of regression trees, with the following compositions:
    1. Each tree trained on a different random subset of 33 (out of 67) training samples, where the ensemble contains either:
        a. 10 regression trees
        b. 100 regression trees
        c. 1000 regression trees or
        d. 10000 regression trees

    To generate random sample subsets for these ensembles, use `randperm()`, similar to what you did for Project 1.

2. Each tree trained on a different bootstrap sample of the training data, where the ensemble contains either:
    a. 10 regression trees
    b. 100 regression trees
    c. 1000 regression trees or
    d. 10000 regression trees

    Remember that a bootstrap sample has the same number of samples in it as the original data (see slides 25-26 of Lecture 13). If you have a training dataset with `nTrain` samples, a convenient way to generate a bootstrap sample in MATLAB is with the `randi()` function (see Help for explanation):

```
bs = randi( nTrain, nTrain, 1 );
trainDataSub = trainData( bs, : );
trainLabelSub = trainLabel( bs, : );
```

For both modes of random sampling, reset the default random number generator with a new random seed before each sampling (please use the sequence of seeds 1 through n = number of base regressors).

- For each of the 8 ensembles, calculate and report to the console:
    1. The number of base regressors in the ensemble.
    2. The average MSE of the base regressors on the test set.
    3. The MSE of the ensemble predictions on the test set.

I would like to see the output in roughly this format:

```
>> regressionTreeEnsemble
test MSE - single tree, all data                 =   0.5029
number of random trees in ensemble = 10000
test MSE - average of single trees, subset data  =   x.xxxx
test MSE - ensemble of single trees, subset data =   x.xxxx
Elapsed time is 80.128975 seconds.
```

where, of course, your calculated MSEs appear in place of the `x.xxxx`.

Recall (slide 8 of Lecture 13) that the prediction of a regression ensemble for a test sample is simply the average of the predictions for that test sample from the individual base regressors. The MSE over all such ensemble test sample predictions becomes:

`"test MSE - ensemble of single trees, subset data"`

Don't confuse this with:

`"test MSE - average of single trees, subset data"`

which is the average of the MSEs calculated for each individual tree, using only the predictions from that individual tree. Comparing the two is a key measure of the performance increase afforded by an ensemble.

To reiterate, the first is an MSE based on averages of predictions across all trees, and the second is the average of MSEs for individual trees. If you don't fully understand the distinction, ask – early and often.

For Part 1, turn in the following:

- A file `regressionTreeEnsemble.m` containing your script. It is fine to have a single script that you modify to generate the single tree and the 8 different ensembles; just make sure the modifications stay in the script in commented form.
- The console outputs from the single tree trained on all the data and all 8 ensembles (cut-and-paste into a document). Identify which came from random subset sampling, and which from bootstrap sampling.
- Examine the trend in MSE vs. number of base regressors for the two methods of random sampling, and answer the following questions:
    - Which method has the best ultimate improvement in MSE, relative to a single tree trained on all the data?
    - Which method shows the better improvement in MSE at the smaller ensemble sizes?
    - Does one of the methods take longer (wall clock time) for the ensemble to run?
    - Based on your answers to the above questions, which sampling method would you recommend for this relatively small dataset?

As a bit of guidance, the best improvement in RMSE for Part 1, relative to a single tree trained on all the data, lies somewhere between 0.08 and 0.12.

Part 2. [20 points] For this Part, you will write a MATLAB script that trains and uses an ensemble of classification trees on the mnistabridged handwritten digits dataset.

Aggregating predictions may be simpler for this dataset if you first convert the class labels (0-9) to group labels (1-10) with `grp2idx()`, as was illustrated in `matlab_demo_16.m`.

For your base classifier, use a standard MATLAB classification tree with default settings, as was done for this dataset in Part C of `matlab_demo_04.m`.

Requirements:

- First train a single tree on all the training data, use the tree to make predictions on the 1000 samples in the test set, and report to the console the number of correct classifications and percent correct on the test set.

- Generate 3 ensembles of 100 classification trees, where each tree is trained on a random subset of the 5000 training samples. The 3 ensembles should use different random subset sizes, either:
    1. 500 samples
    2. 1000 samples
    3. 2000 samples

To generate random sample subsets for these ensembles, use `randperm()`, similar to what you did for Project 1.

- Generate one ensemble of 100 classification trees, where each tree is trained on a bootstrap sample of the 5000 training samples. To generate bootstrap samples, use `randi()`, as explained in Part 1.

- For all four ensembles, reset the default random number generator with a new random seed before each sampling (please use the sequence of seeds 1 through n = number of base classifiers).

- For each of the 4 ensembles, calculate and report to the console:
    1. The number of base classifiers in the ensemble.
    2. The number of random samples in each base classifier.
    3. The average number and percent correct of the base classifiers on the test set.
    4. The number and percent correct of the ensemble predictions on the test set.

  I would like to see the output in roughly this format:

```
>> classificationTreeEnsemble
correct predictions on test set         :   752 / 1000,  75.20%
Elapsed time is 15.758007 seconds.
number of random trees in ensemble    = 100
number of random samples in each tree = 500
average correct predictions on test set :   xxx.xx / 1000,  xx.xx%
ensemble correct predictions on test set:   xxx / 1000,  xx.xx%
Elapsed time is 181.763556 seconds.
```

  where, of course, your numbers and percents correct appear in place of the `xxx`.

  Recall (slides 8-11 of Lecture 13) that the prediction for a test sample by a committee-style classification ensemble is simply the majority (or plurality) vote across the predictions of the individual base classifiers for that test sample. This particular dataset has 10 class labels. One convenient approach to aggregating predictions would be to set up a 10 x 1000 vote matrix, where each column holds the tallies of votes for the 10 labels for one of the 1000 test samples. As each base classifier returns its predictions for the test samples, you would increment a single element in each column by 1, with the element depending on what class was predicted for that test sample. After votes are tallied from all 100 base classifiers, the ensemble prediction is made for each test sample by finding the element in that sample's column with the largest tally.

For Part 2, turn in the following:

- A file `classificationTreeEnsemble.m` containing your script. It is fine to have a single script that you modify to generate the single tree and the 4 different ensembles; just make sure the modifications stay in the script in commented form.

- The console outputs from the single tree trained on all the data and all 4 ensembles (cut-and-paste into a document). Identify which came from random subset sampling, and which from bootstrap sampling.
- Compare the number of correct ensemble classifications for the two methods of random sampling, and answer the following questions:
    - Which method ultimately shows the best gain in accuracy relative to a single tree trained on all the data?
    - What is the trend in accuracy vs. random sample size for the random subset method?
    - Does one of the methods take longer (wall clock time) for the ensemble to run?
    - Based on your answers to the above questions, which sampling method would you recommend for this particular dataset, assuming highest accuracy is your goal?

As a bit of guidance, the best gain in accuracy for an ensemble in Part 2, relative to a single tree trained on all the data, lies somewhere between 80 / 1000 and 160 / 1000.