Paul Le Tran
plt377
12/05/22
ECO388E

Problem Set 6 (Problem Set 3 for spring 2022)

Overview of the machine learning model and data). As an overview, the machine learning model we use to produce a localisation solution is a random forest model. More specifically, we create an ensemble of learner decision trees and employ bootstrap aggregation (i.e., bagging) by generating many bootstrap replicas of the data set and growing these decision trees on said replicas. The "random forest" aspect comes from how each decision tree randomly selects predictors for each decision split.

The dataset that we employ to train and validate this random forest model is the University of California, Irvine's "UCIIndoorLoc" database. The core of the database every row is location information recorded from an individual's Android device. This includes the "Wi-Fi fingerprint" of the Android device (i.e., the Wireless Access Points (WAPs) accessed by the device and the corresponding Received Signal Strength Intensity (RSSI)), the coordinates of that device when recorded (i.e., latitude, longitude), and the building information in which that device was found in (i.e., building ID and floor number). The training data set consists of 19,937 observations whilst the validation data set consists of 1,111 observations.

Our random forest model described above essentially uses the RSSI from all 520 WAPs in the training data set in order to predict the floor number, building ID, latitude, and longitude of a device. Observe that the former two variables are categorical whilst the latter two are continuous. Therefore, the random forest model faces a classification problem for the former two and a regression problem for the latter two. Accordingly, we develop two types of random forest models to address these different problems.

Overview of addressing overfitting and basic hyperparameter tuning). Due to time and computing constraints, we are not able to finely tune the random forest models for our four problems (i.e., choosing the optimal number of decision trees). We instead use a basic tuning principle of visually observing when the error measure of the model, when applied to the validation data set, plateaus for the first time. We choose the corresponding number of trees as the stopping point for the model in order to minimise overfitting to the training data set. Our error measure for this exercise is the mean squared error (MSE). Specifically, we look at both the testing data set MSE and out-of-bag MSE when visually determining what is the stopping point with respect to number of decision trees for each model.

Results). To address overfitting and perform basic hyperparameter tuning mentioned above, we first train all four random forest models (2 classifications, 2 regressions) using 100 decision trees. From there, we visually examine the plots of testing data set MSE and out-of-bag MSE and look for the first instance of plateauing. This process yielded the following number of decision trees for each model/problem: For predicting floor number, the model has 40 decision trees; for predicting building ID, the model has 19 decision trees; for predicting latitude, the model has 13 decision trees, and for predicting longitude, the model has 42 decision trees. All the following results presented are for models trained with these number of decision trees.

We first check if our random forest models are learning and “understanding” how to use the RSSI of devices in order to predict their corresponding floor number, building ID, latitude, or longitude. We do this by observing if there are large differences between the fitted values and the actual values of the output variables in the training data set. We display this in Figure 1 – 4 below and on the next page.

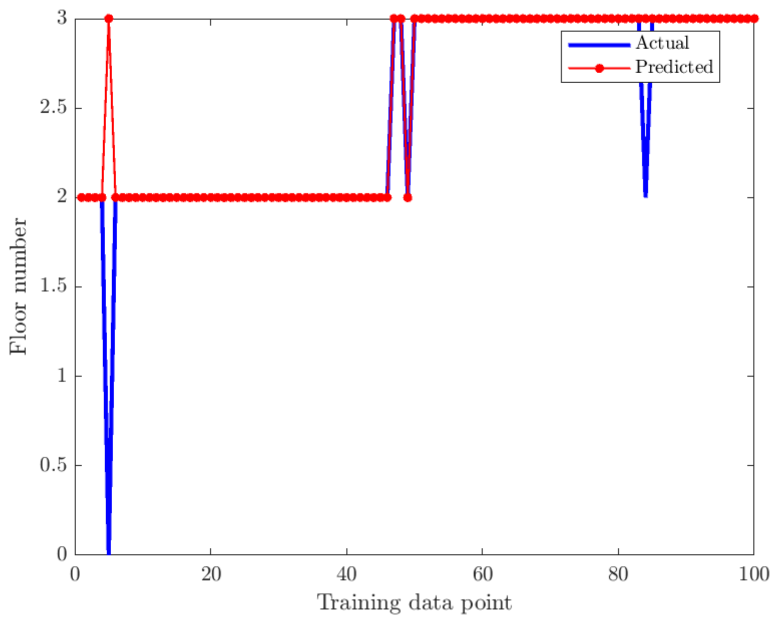

Figure 1: First 100 model predicted values and actual values of floor number in the training data set.

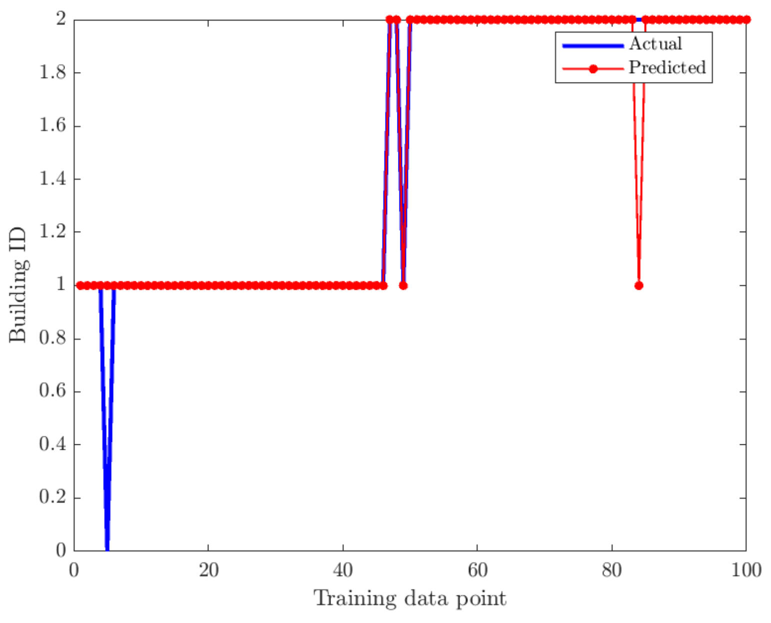

Figure 2: First 100 model predicted values and actual values of building ID in the training data set.

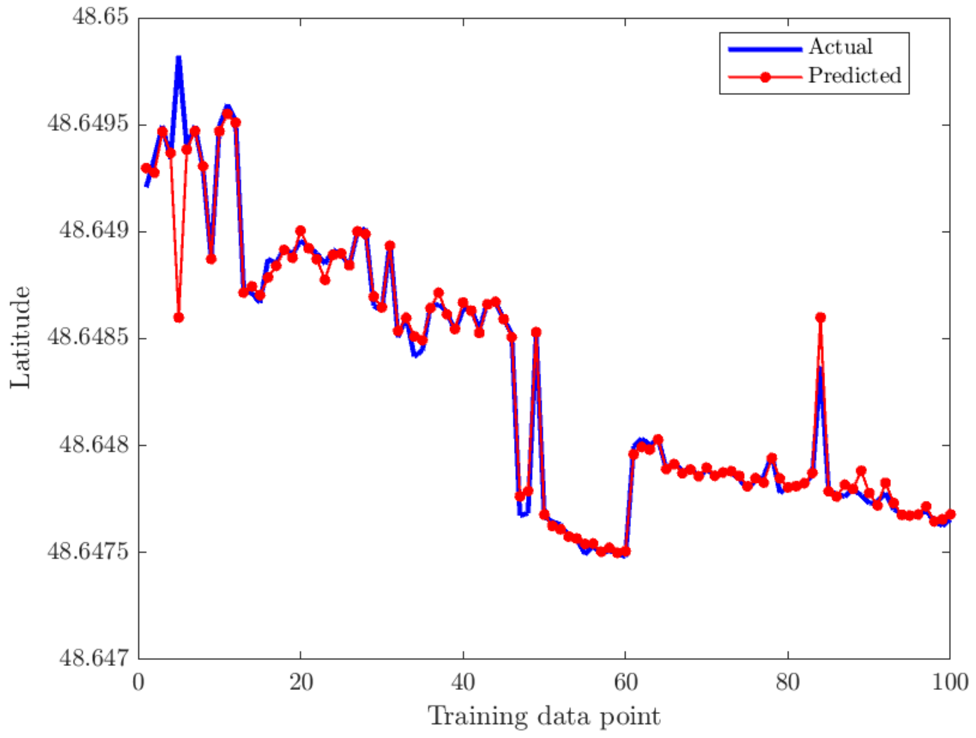

Figure 3: First 100 model predicted values and actual values of latitude in the training data set.

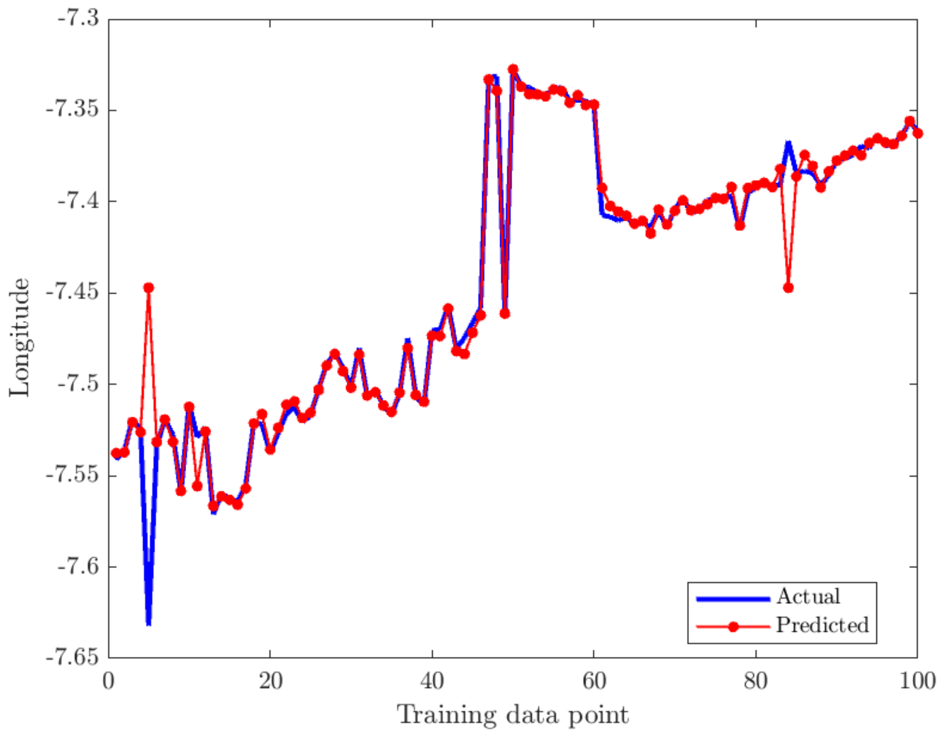

Figure 4: First 100 model predicted values and actual values of longitude in the training data set.

In both classification and regression problems, we see that our random forest models are performing decently in terms of learning the relationships in the training data set. Besides a small handful of data points, all four models gave predicted values that either match or are extremely close to the actual values.

We now present the plots of testing MSE, out-of-bag MSE, and training MSE for all four models in Figures 5 – 8 below.

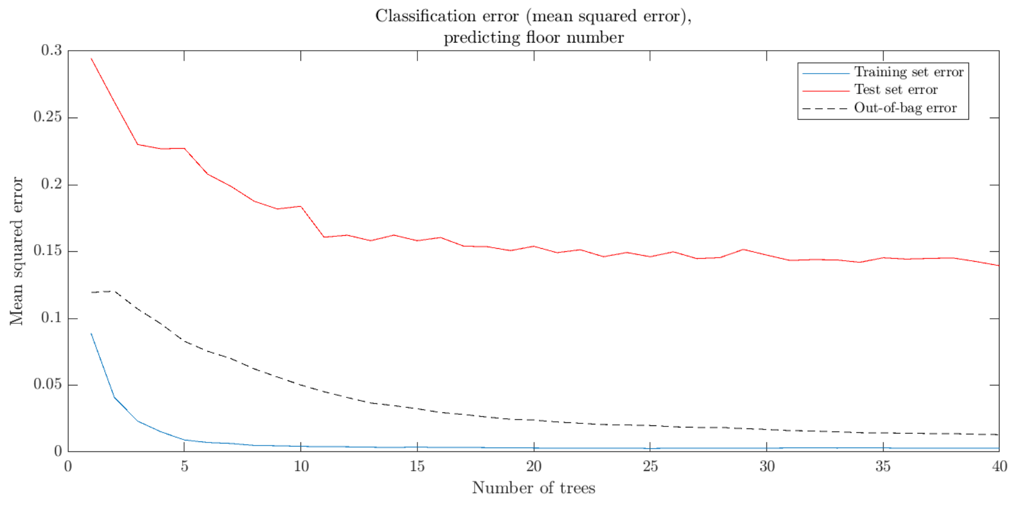

Figure 5: Testing, training, and out-of-bag MSE for random forest model predicting floor number (classification problem).

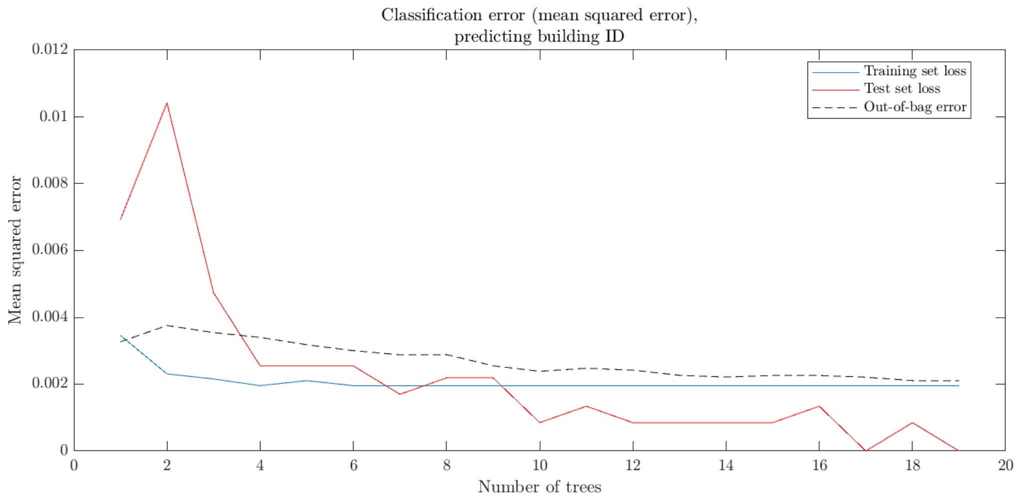

Figure 6: Testing, training, and out-of-bag MSE for random forest model predicting building ID (classification problem).

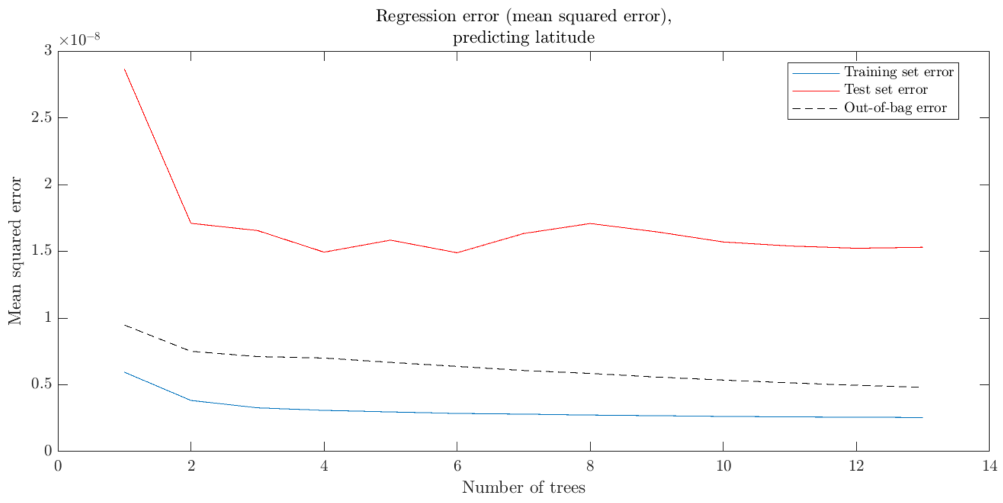

Figure 7: Testing, training, and out-of-bag MSE for random forest model predicting latitude (regression problem).

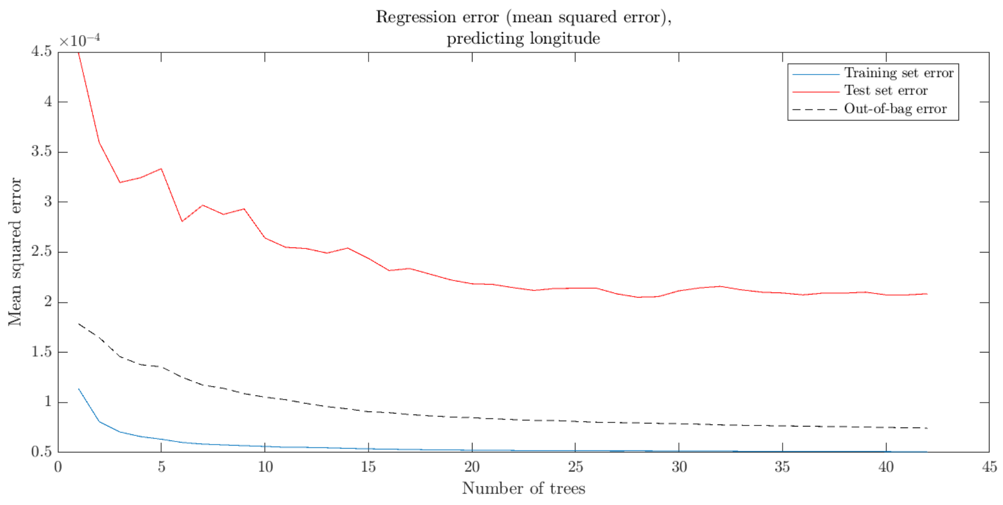

Figure 8: Testing, training, and out-of-bag MSE for random forest model predicting longitude (regression problem).

Initial impressions are that whilst each model's testing MSE (red) follows a different path, they are all quite small in terms of absolute magnitude. This indicates the our random forest models perform pretty well for just basic hyperparameter tuning. An interesting characteristic found in all models is that the out-of-bag MSE (black) is much closer to the training MSE (blue) than to the testing MSE.

Lastly, we report the final testing MSE of each tuned model. For predicting floor number, our testing MSE is 0.1394; for predicting building ID, our testing MSE is essentially zero; for predicting latitude, our testing MSE is 1.5354×10^{-8}; and for predicting longitude, our testing MSE is 1.8548×10^{-4}.

Appendix). The following is the Matlab code used to produce the results presented and discussed in this problem set. For conciseness, we omit the portions of code that import in the dataset.

```
%=========================================================================
%% Setting up workspace
clear all;
close all;
clc;

home_dir = 'Path\to\programmes';
data_dir = 'Path\to\data';

% Setting text interpreter to latex
set(groot, 'defaultAxesTickLabelInterpreter', 'latex');
set(groot, 'defaulttextinterpreter', 'latex');
set(groot, 'defaultLegendInterpreter', 'latex');

% Setting random seed for reproductibility
rng(47);

cd(home_dir);
%=========================================================================

%=========================================================================
%% Training and evaluation ensemble of boosted learner trees for both
classification (building ID and floor number) and regression (latitude and
longitude) problems
% Creating template tree for classification trees
t = ClassificationTree.template('MinLeaf', 1);
%=================================================
% Variable to predict: Floor number (classification)
%=================================================
% Fitting ensemble of learner trees
tic;
disp('Training algorithm for floor');
floor_mdl = fitcensemble(regressors_train, floor_train, 'Method', 'Bag',
'NumLearningCycles', 40, 'Learners', t);
toc;

% Evaluating model
mse_floor = loss(floor_mdl, regressors_eval, floor_eval, 'mode',
'ensemble');
fprintf('Mean-square testing error = %f\n', mse_floor);

%================================================
% Variable to predict: Building ID (classification)
%================================================
% Fitting ensemble of learner trees
tic;
disp('Training algorithm for building ID');
bldg_mdl = fitcensemble(regressors_train, bldg_train, 'Method', 'Bag',
'NumLearningCycles', 19, 'Learners', t);
toc;
```

```
% Evaluating model
mse_bldg = loss(bldg_mdl, regressors_eval, bldg_eval, 'mode', 'ensemble');
fprintf('Mean-square testing error = %f\n', mse_bldg);

% Creating template tree for regression trees
t = RegressionTree.template('MinLeaf', 5);
%============================================
% Variable to predict: Latitude (regression)
%============================================
% Fitting ensemble of learner trees
tic;
disp('Training algorithm for latitude');
lat_mdl = fitrensemble(regressors_train, lat_train, 'Method', 'Bag',
'NumLearningCycles', 13, 'Learners', t);
toc;

% Evaluating model
mse_lat = loss(lat_mdl, regressors_eval, lat_eval, 'mode', 'ensemble');
fprintf('Mean-square testing error = %f\n', mse_lat);

%=============================================
% Variable to predict: Longitude (regression)
%=============================================
% Fitting ensemble of learner trees
tic;
disp('Training algorithm for longitude');
long_mdl = fitrensemble(regressors_train, long_train, 'Method', 'Bag',
'NumLearningCycles', 42, 'Learners', t);
toc;

% Evaluating model
mse_long = loss(long_mdl, regressors_eval, long_eval, 'mode', 'ensemble');
fprintf('Mean-square testing error = %f\n', mse_long);
%===========================================================================

%===========================================================================
%% Examining learning capability of algorithm by plotting actual values in
training data with model fitted values for training data
%====================================================
% Variable to predict: Floor number (classification)
%====================================================
plot(floor_train, 'b', 'LineWidth', 2);
hold on;
plot(predict(floor_mdl, regressors_train), 'r.-', 'LineWidth',
1,'MarkerSize', 15);

% Only showing first 100 points. One has the option to pan to view more
% Observe first hundred points, pan to view more
xlim([0 100]);
legend({'Actual', 'Predicted'}, 'Location', 'Best');
xlabel('Training Data point');
ylabel('Floor number');
hold off;
saveas(gcf, 'Path\to\graphics\floor_1_plot.png');
close(gcf);
```

```matlab
%==================================================
% Variable to predict: Building ID (classification)
%==================================================
plot(bldg_train, 'b', 'LineWidth', 2);
hold on;
plot(predict(bldg_mdl, regressors_train), 'r.-', 'LineWidth',
1,'MarkerSize', 15);

% Only showing first 100 points. One has the option to pan to view more
% Observe first hundred points, pan to view more
xlim([0 100]);
legend({'Actual', 'Predicted'}, 'Location', 'Best');
xlabel('Training Data point');
ylabel('Building ID');
hold off;
saveas(gcf, 'Path\to\graphics\bldg_1_plot.png');
close(gcf);

%============================================
% Variable to predict: Latitude (regression)
%============================================
plot(lat_train, 'b', 'LineWidth', 2);
hold on;
plot(predict(lat_mdl, regressors_train), 'r.-', 'LineWidth',
1,'MarkerSize', 15);

% Only showing first 100 points. One has the option to pan to view more
% Observe first hundred points, pan to view more
xlim([0 100]);
legend({'Actual', 'Predicted'}, 'Location', 'Best');
xlabel('Training Data point');
ylabel('Latitude');
hold off;
saveas(gcf, 'Path\to\graphics\lat_1_plot.png');
close(gcf);

%=============================================
% Variable to predict: Longitude (regression)
%=============================================
plot(long_train, 'b', 'LineWidth', 2);
hold on;
plot(predict(long_mdl, regressors_train), 'r.-', 'LineWidth',
1,'MarkerSize', 15);

% Only showing first 100 points. One has the option to pan to view more
% Observe first hundred points, pan to view more
xlim([0 100]);
legend({'Actual', 'Predicted'}, 'Location', 'Best');
xlabel('Training Data point');
ylabel('Longitude');
hold off;
saveas(gcf, 'Path\to\graphics\long_1_plot.png');
close(gcf);
%============================================================================



%============================================================================
%% Plotting MSE of training data, validation data, and out-of-bag with
respect to number of boosted learner trees
```

```
%====================================================
% Variable to predict: Floor number (classification)
%====================================================
% Creating training error function
training_error = resubLoss(floor_mdl, 'mode', 'cumulative');
% Creating testing error function
test_error = loss(floor_mdl, regressors_eval, floor_eval, 'mode',
'cumulative');
% Creating out-of-bag error function
oob_error = oobLoss(floor_mdl, 'mode', 'cumulative');

plot(training_error);
hold on;
plot(test_error, 'r');
plot(oob_error, 'k--');
title({'Classification error (mean squared error),', 'predicting floor
number'});
legend({'Training set error', 'Test set error', 'Out-of-bag error'},
'Location', 'Best');
xlabel('Number of trees');
ylabel('Mean squared error');
set(gcf,'Position', [249 634 1009 420]);
hold off;
saveas(gcf, 'Path\to\graphics\floor_2_plot.png');
close(gcf);

%===================================================
% Variable to predict: Building ID (classification)
%===================================================
% Creating training loss function
training_error = resubLoss(bldg_mdl, 'mode', 'cumulative');
% Creating testing loss function
test_error = loss(bldg_mdl, regressors_eval, bldg_eval, 'mode',
'cumulative');
% Creating out-of-bag error function
oob_error = oobLoss(bldg_mdl, 'mode', 'cumulative');

plot(training_error);
hold on;
plot(test_error, 'r');
plot(oob_error, 'k--');
title({'Classification error (mean squared error),', 'predicting building
ID'});
legend({'Training set loss', 'Test set loss', 'Out-of-bag error'},
'Location', 'Best');
xlabel('Number of trees');
ylabel('Mean squared error');
set(gcf,'Position', [249 634 1009 420]);
hold off;
saveas(gcf, 'Path\to\graphics\bldg_2_plot.png');
close(gcf);

%============================================
% Variable to predict: Latitude (regression)
%============================================
% Creating training loss function
training_error = resubLoss(lat_mdl, 'mode', 'cumulative');
% Creating testing loss function
test_error = loss(lat_mdl, regressors_eval, lat_eval, 'mode',
'cumulative');
```

```
% Creating out-of-bag error function
oob_error = oobLoss(lat_mdl, 'mode', 'cumulative');

plot(training_error);
hold on;
plot(test_error, 'r');
plot(oob_error, 'k--');
title({'Regression error (mean squared error),', 'predicting latitude'});
legend({'Training set error', 'Test set error', 'Out-of-bag error'},
'Location', 'Best');
xlabel('Number of trees');
ylabel('Mean Squared Error');
set(gcf,'Position', [249 634 1009 420]);
hold off;
saveas(gcf, 'Path\to\graphics\lat_2_plot.png');
close(gcf);

%==============================================
% Variable to predict: Longitude (regression)
%==============================================
% Creating training loss function
training_error = resubLoss(long_mdl, 'mode', 'cumulative');
% Creating testing loss function
test_error = loss(long_mdl, regressors_eval, long_eval, 'mode',
'cumulative');
% Creating out-of-bag error function
oob_error = oobLoss(long_mdl, 'mode', 'cumulative');

plot(training_error);
hold on;
plot(test_error, 'r');
plot(oob_error, 'k--');
title({'Regression error (mean squared error),', 'predicting longitude'});
legend({'Training set error', 'Test set error', 'Out-of-bag error'},
'Location', 'Best');
xlabel('Number of trees');
ylabel('Mean Squared Error');
set(gcf,'Position', [249 634 1009 420]);
hold off;
saveas(gcf, 'Path\to\graphics\long_2_plot.png');
close(gcf);
%============================================================================
```