```
clear all
rng("default")

%loadind data
TestData = csvread('.\Test1.csv', 1);
TrainData = csvread('.\Train1.csv', 1);

%feature selection
TestData =TestData(:,[1,2,3,4,5,10]);
TrainData =TrainData(:,[1,2,3,4,5,10]);

% Hyperparametes list
n_estimators = [5,10,20,40,80,160,300];
n_predictors = [2,3,4];

% K-fold Cross validation
cv = cvpartition(size(TrainData, 1),"KFold",10);

% Initialize the hp_perf matrix with the correct size
hp_perf = zeros(length(n_estimators), length(n_predictors));
cv_results =[];
training_times = zeros(length(n_estimators), length(n_predictors));

% Perform the grid search
for i = 1:length(n_estimators)
    for j = 1:length(n_predictors)
        for l = 1:10
        % Splitting train and validation set in each iteration
        idx_Train = training(cv,l);
        TrainData_Kfold =TrainData(idx_Train,:);
        idx_val = test(cv,l);
        ValData_kfold =TrainData(idx_val,:);
        X = TrainData_Kfold(:,1:end-1);
        y = TrainData_Kfold(:,end);

         % Start the timer
        tic;

        % Train the random forest model with the new hyperparameter value
        RFmodel_1 = TreeBagger(n_estimators(i),X,y,'NumPredictorsToSample', ...
            n_predictors(j));

        %Stop the timer and measure the training time
        training_time = toc;
        training_times(i,j) = training_time;


        %cross validation accuracy
        predictions = predict(RFmodel_1,ValData_kfold(:,1:end-1));
        predictions = str2num(cell2mat(predictions));
        iscorrect = predictions == ValData_kfold(:,end);
        correctrate = sum(iscorrect)/numel(predictions);
        cv_results(l) = correctrate;
        avg_cvresults = mean(cv_results); % average Accrurcy
        hp_perf(i,j) = avg_cvresults;

        end
    end
end



% Find the maximum value in the hp_perf matrix
[max_value, max_idx] = max(hp_perf(:));

% Find the indices of the maximum value in the hp_perf matrix
[i, j] = ind2sub(size(hp_perf), max_idx);

% Get the values of the hyperparameters that gave the best performance
best_n_estimators = n_estimators(i);
best_n_predictors = n_predictors(j);

% Compute the minimum, maximum, and average training time
min_training_time = min(training_times(:));
max_training_time = max(training_times(:));
mean_training_time = mean(training_times(:));



% Train the random forest model with the best hyperparameters on the entire
% train data

X = TrainData(:,1:end-1);
y = TrainData(:,end);

tic
RFmodel_best = TreeBagger(best_n_estimators,X,y,'NumPredictorsToSample', ...
    best_n_predictors,OOBPredictorImportance='on',OOBPrediction='on');

Final_model_traintime = toc;




% Evaluate the model on the test set
X_test = TestData(:,1:end-1);
y_test = TestData(:,end);
tic
predictions = predict(RFmodel_best,X_test);
predict_time = toc;

predictions = str2num(cell2mat(predictions));
```

```
iscorrect = predictions == y_test;
test_accuracy = sum(iscorrect) / numel(predictions);


% Compute the confusion matrix
cm = confusionmat(y_test, predictions);

%plot confusion matrix
confusionchart(y_test,predictions,"Normalization","absolute")
confusionchart(y_test,predictions,"Normalization","row-normalized")
confusionchart(y_test,predictions,"Normalization","column-normalized")

%compute percision recall and f1 score
precision = cm(2,2) / (cm(2,2) + cm(1,2));
recall = cm(2,2) / (cm(2,2) + cm(2,1));
f1 = 2 * precision * recall / (precision + recall);

% Get the predicted probabilities for the test set
[predictions, scores] = predict(RFmodel_best,X_test);
% Convert the predicted labels to a binary vector
predictions = str2num(cell2mat(predictions));
% Compute the ROC curve
[fpr, tpr, thr] = perfcurve(y_test, scores(:,2), 1);
% Compute the AUC value
auc = trapz(fpr,tpr);
% Plot the ROC curve
figure;
plot(fpr,tpr);
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title(sprintf('ROC curve (AUC = %0.2f)', auc));



% Plot the validation accuracy as a function of the number of trees
figure;
plot(n_estimators, hp_perf(:,j), 'o-');
xlabel('Number of trees');
ylabel('Validation accuracy');



% Plot the validation accuracy as a function of the number of predictors
figure;
plot(n_predictors, hp_perf(i,:), 'o-');
xlabel('Number of predictors');
ylabel('Validation accuracy');
```

```
% Create a new figure
figure;

% Loop through each value of n_predictors
for j = 1:length(n_predictors)
    % Extract the average accuracy for each value of n_estimators
    avg_accuracy = hp_perf(:,j);

    % Plot the average accuracy as a function of n_estimators
    plot(n_estimators, avg_accuracy);

    % Add a legend entry for this value of n_predictors
    legend_entry = sprintf('Num Predictors = %d', n_predictors(j));
    legend(legend_entry);

    % Hold on to the current plot so we can add more lines to it
    hold on;
end

% Add a title and axis labels to the plot
title('Average Accuracy vs. Num Trees');
xlabel('Num Trees');
ylabel('Average Accuracy');




%plot avg accuracy vs number of trees
accuracy_matrix = zeros(length(n_estimators));
% Create a new figure
figure;

% Loop through each row of the hp_perf matrix
for i = 1:length(n_estimators)
    % Extract the average accuracy for each value of n_estimators
    avg_accuracy = mean(hp_perf(i,:));

    accuracy_matrix(i) = avg_accuracy;
end
% Plot the average accuracy as a function of n_estimators
plot(n_estimators, accuracy_matrix);

% Set the y-axis limits to be from 0.65 to 0.80
ylim([0.6 0.75])

% Add a title and axis labels to the plot
title('Average Accuracy vs. Num Trees');
xlabel('Num Trees');
ylabel('Average Accuracy');
```

```
%plot average accuracy vs number of predictors
accuracy_matrix = zeros(length(n_predictors));
% Create a new figure
figure;

% Loop through each row of the hp_perf matrix
for i = 1:length(n_predictors)
    % Extract the average accuracy for each value of n_estimators
    avg_accuracy = mean(hp_perf(:,i));

    accuracy_matrix(i) = avg_accuracy;
end
% Plot the average accuracy as a function of n_estimators
plot(n_predictors, accuracy_matrix);

% Set the y-axis limits to be from 0.65 to 0.80
ylim([0.65 0.75])
xticks(n_predictors);

% Add a title and axis labels to the plot
title('Average Accuracy vs. Num Predictors');
xlabel('Num predectors');
ylabel('Average Accuracy');




%plot average training time vs number of predictors
train_matrix = zeros(length(n_predictors));
% Create a new figure
figure;

% Loop through each row of the hp_perf matrix
for i = 1:length(n_predictors)
    % Extract the average accuracy for each value of n_estimators
    avg_traintime = mean(training_times(:,i));

    train_matrix(i) = avg_traintime;
end
% Plot the average accuracy as a function of n_estimators
plot(n_predictors, train_matrix);

xticks(n_predictors);

% Add a title and axis labels to the plot
title('Average training time vs. Num Predictors');
xlabel('Num of Predictors');
ylabel('Average training time');
```

```
%plot average training time vs number of trees
train_matrix = zeros(length(n_estimators));
% Create a new figure
figure;

% Loop through each row of the hp_perf matrix
for i = 1:length(n_estimators)
    % Extract the average accuracy for each value of n_estimators
    avg_traintime = mean(training_times(i,:));

    train_matrix(i) = avg_traintime;
end
% Plot the average accuracy as a function of n_estimators
plot(n_estimators, train_matrix);

% Add a title and axis labels to the plot
title('Average training time vs. Num Trees');
xlabel('Num trees');
ylabel('Average training time');
```