```
function R = bpw2_classify3b(matfile)
% As before, but do cross-validation following https://www.mathworks.com/help/stats/fitcecoc.html.

% After running, do this to examine the result R.
% load('/local/matlab/bpstress/bpw2_classify3c1.mat')

% Initialize the result.
R = {};
% The initial part of this is like bpw2_stat1.
if nargin < 1
    %matfile = '/local/matlab/Kaldi-alignments-matlab/data-bpn/tab4-sample.mat'; % Made with token_data_bpw2.
    matfile = '/local/matlab/bpstress/data-bpn/tab4.mat'; % All the data, 15388 bisyllables
    savename = '/local/matlab/bpstress/data-bpn/bpw2_classify3b';
end



% Load sets L to a structure. It has to be initialized first.
L = 0;
load(matfile);

% Initialize the result.
R = {};

% Scale for combining the two weights.
acoustic_scale = 0.083333;
% Then combine by this formula, see
% /projects/speech/sys/kaldi-master/egs/bp_ldcWestPoint/bpw2/exp/u1/decode_word_1/tab-min.awk
% weight = weight1 +  acoustic_scale * weight2;

% Duration in frames
D = cellfun(@sum,L.phonedur)';

% Combined weights
W1 = cellfun(@(x,y) x + acoustic_scale * y,L.weight1,L.weight2,'UniformOutput',false)';

% Combined weights scaled down by duration.
% This produces weights in the range 7.0 to 9.5.
W2 = cellfun(@(x,y) x ./ y,W1,num2cell(D),'UniformOutput',false);

% This part is like bpw2_stat3.m
% Logical indices of ultimate-stressed triplus-syllables
% and penultimate-stressed triplus, and
% ante-penultimate tripus
U31 = L.syl > 2 & L.cstress == 1;
U32 = L.syl > 2 & L.cstress == 2;
U33 = L.syl > 2 & L.cstress == 3;

% Logical indices of all tokens with three or more syllables
U3 = L.syl > 2;

% Indices that are 1 in U3, for mapping back to L.
I3 = find(U3);

% Corresponding matrices of weights, with varying number of readings.
% Cell3mat can't be applied.
U31wv = W2(U31);  % 1584 3
U32wv = W2(U32);  % 7331 3
U33wv = W2(U33);  %  336 3
U3wv = W2(U3);

% Select three columns and map to matrix
% Each token is characterized by its weights in three readings.
U31w = cell2mat(cellfun(@(x) [x(1),x(2),x(3)], U31wv,'UniformOutput',false));
U32w = cell2mat(cellfun(@(x) [x(1),x(2),x(3)], U32wv,'UniformOutput',false));
U33w = cell2mat(cellfun(@(x) [x(1),x(2),x(3)], U33wv,'UniformOutput',false));
U3w = cell2mat(cellfun(@(x) [x(1),x(2),x(3)], U3wv,'UniformOutput',false));

%%%%%%%% Duration %%%%%%%%
% Vowel durations. L.voweldur is not of uniform length,
% and the vowels need to count from the end. This is
% adjusted by the anonymous function.
% We assume L.voweldur has vowel lengths in time order.
U31d = cell2mat(cellfun(@(x) [x(length(x)),x(length(x)-1),x(length(x)-2)], L.voweldur(U31)','UniformOutput',false));
U32d = cell2mat(cellfun(@(x) [x(length(x)),x(length(x)-1),x(length(x)-2)], L.voweldur(U32)','UniformOutput',false));
U33d = cell2mat(cellfun(@(x) [x(length(x)),x(length(x)-1),x(length(x)-2)], L.voweldur(U33)','UniformOutput',false));
U3d = cell2mat(cellfun(@(x) [x(length(x)),x(length(x)-1),x(length(x)-2)], L.voweldur(U3)','UniformOutput',false));

% Feature matrix, with six colums. Row indices are items.
X = [U3w,U3d];
```

```
% 1 indicating final stress
% 2 penultimate stress
% 3 antepenultimate stress.
Y = U31 + U32 * 2 + U33 * 3;
Y = Y(U3);


% Save the data
R.X = X;
R.Y = Y;

dim = length(X(:,1));
R.dim = dim;

% Fit 3-class svm using just weights, just durations
% and both. Durations help a bit.

R.svm1 = fitcecoc(X(:,1:3),Y); % Cepstral weight
R.svm2 = fitcecoc(X(:,4:6),Y); % Duration
R.svm3 = fitcecoc(X,Y);        % Both



% Evaluate on training data.
% resubLoss(R.svm1) % 0.0626  weights
% resubLoss(R.svm2) % 0.2101  durations
% resubLoss(R.svm3) % 0.0507  both
disp(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%  Crossvalidate following
%  https://www.mathworks.com/help/stats/fitcecoc.html.

% Template
t1 = templateSVM('Standardize',1);
t2 = templateSVM('Standardize',1);
t3 = templateSVM('Standardize',1);

% Train the ECOC classifier

%mdl1 = fitcecoc(X(:,1:3),Y,'Learners',t1);
%mdl2 = fitcecoc(X(:,4:6),Y,'Learners',t2);
%mdl3 = fitcecoc(X,Y,'Learners',t3);

% Thu Nov 22 09:29:01 EST 2018
% The prior can be 'empirical' (default) or 'uniform', see https://www.mathworks.com/help/stats/fitcecoc.html
R.mdl1 = fitcecoc(X(:,1:3),Y,'Learners',t1,'Prior','empirical');
R.mdl2 = fitcecoc(X(:,4:6),Y,'Learners',t2,'Prior','empirical');
R.mdl3 = fitcecoc(X,Y,'Learners',t3,'Prior','empirical');

% Cross-validate Mdl using 10-fold cross-validation.

R.cmdl1 = crossval(R.mdl1);
R.cmdl2 = crossval(R.mdl2);
R.cmdl3 = crossval(R.mdl3);

% The following values were obtained on Jan 2, 2024
%                           basic   uniform
R.loss1 = kfoldLoss(R.cmdl1) %  0.0556  0.0753
R.loss2 = kfoldLoss(R.cmdl2) %  0.2101  0.3465
R.loss3 = kfoldLoss(R.cmdl3) %  0.0461  0.0582

% What is the major-class baseline? Is basic weight-only any better?

% Save R
% To see R,
%   load('/local/matlab/bpstress/bpw2_classify3c1.mat')
save(savename,'R');

disp(1);

% Parse a line into a key and a vector of int.
function [key,a] = parse_alignment(line)
    key = sscanf(line,'%s',1);
    [~,klen] =  size(key);
    [~,llen] = size(line);
    line = line((klen+1):llen);
    a = sscanf(line,'%d')';
end

% Parse a line from the table.
% The input line looks like this.
% f58br08b11k1-s087-2   abacaxi abacaxi_U411    4       1       1       4.45933 4.46457 4.43014 4.40614 5115.16 5122.39 5166.43 5153.47 362_364_3
```

```
% uid                  wf1     wf2             syl cit dec [w1] [w2]
%   bns04_st1921_trn 1 12 ; 6 7 ; 143 3 ; 50 8 ; 60 3 ; 143 4 ; 146 13
function [uid,word_form1,word_form2,syl_count,citation_stress,decode_stress,weight1,weight2] = parse_line(line)
    part = strsplit(line,'\t');
    uid = part{1};
    word_form1 = part{2};
    word_form2 = part{3};
    syl_count = str2num(part{4});
    citation_stress = str2num(part{5});
    decode_stress = str2num(part{6});
    weight1 = str2num(part{7});
    weight2 = str2num(part{8});
end

% Result of 'OptimizeHyperparameters','all'
% Best estimated feasible point (according to models):
%    BoxConstraint    KernelScale    KernelFunction    PolynomialOrder    Standardize
%    _____________    ___________    ______________    _______________    ___________
%
%       6.9424            NaN          polynomial             2              true
%
%Estimated objective function value = 0.082355
%Estimated function evaluation time = 0.48633

% 0.0846 weight
% 0.3035 duration
% 0.0871 both--it's a bit worse

end
```

```
     1


R =

  struct with fields:

        X: [9281×6 double]
        Y: [9281×1 double]
      dim: 9281
     svm1: [1×1 ClassificationECOC]
     svm2: [1×1 ClassificationECOC]
     svm3: [1×1 ClassificationECOC]
     mdl1: [1×1 ClassificationECOC]
     mdl2: [1×1 ClassificationECOC]
     mdl3: [1×1 ClassificationECOC]
    cmdl1: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
    cmdl2: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
    cmdl3: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
    loss1: 0.0560


R =

  struct with fields:

        X: [9281×6 double]
        Y: [9281×1 double]
      dim: 9281
     svm1: [1×1 ClassificationECOC]
     svm2: [1×1 ClassificationECOC]
     svm3: [1×1 ClassificationECOC]
     mdl1: [1×1 ClassificationECOC]
     mdl2: [1×1 ClassificationECOC]
     mdl3: [1×1 ClassificationECOC]
    cmdl1: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
    cmdl2: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
    cmdl3: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
    loss1: 0.0560
    loss2: 0.2101


R =

  struct with fields:

        X: [9281×6 double]
        Y: [9281×1 double]
      dim: 9281
     svm1: [1×1 ClassificationECOC]
     svm2: [1×1 ClassificationECOC]
     svm3: [1×1 ClassificationECOC]
     mdl1: [1×1 ClassificationECOC]
```

```
      mdl2: [1×1 ClassificationECOC]
      mdl3: [1×1 ClassificationECOC]
     cmdl1: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
     cmdl2: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
     cmdl3: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
     loss1: 0.0560
     loss2: 0.2101
     loss3: 0.0462

      1


ans =

  struct with fields:

         X: [9281×6 double]
         Y: [9281×1 double]
       dim: 9281
      svm1: [1×1 ClassificationECOC]
      svm2: [1×1 ClassificationECOC]
      svm3: [1×1 ClassificationECOC]
      mdl1: [1×1 ClassificationECOC]
      mdl2: [1×1 ClassificationECOC]
      mdl3: [1×1 ClassificationECOC]
     cmdl1: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
     cmdl2: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
     cmdl3: [1×1 classreg.learning.partition.ClassificationPartitionedECOC]
     loss1: 0.0560
     loss2: 0.2101
     loss3: 0.0462
```

Published with MATLAB® R2023b