```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1 Part (a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load data_SFcrime_train.mat
% binarize DayOfWeek
DayOfWeek = lower(DayOfWeek); % convert to lower cases
Day_unique = unique(DayOfWeek); % get the unique elements
K_DayOfWeek = numel(Day_unique); % number of possible values
Day_bin = spalloc(numel(DayOfWeek), K_DayOfWeek, numel(DayOfWeek)); %
binarize
for i = 1:K_DayOfWeek
    Day_bin(:,i) = strcmp(DayOfWeek,Day_unique{i});
end
% binarize PdDistrict
PdDistrict = lower(PdDistrict)
Pd_unique = unique(PdDistrict);
K_Pd = numel(Pd_unique);
Pd_bin = spalloc(numel(PdDistrict), K_Pd, numel(PdDistrict));
for i = 1:K_Pd
    Pd_bin(:,i) = strcmp(PdDistrict,Pd_unique{i});
end
% binarize hour information
Hour = zeros(numel(Dates),1);
for i =1:numel(Dates)
    t = Dates{i};
    Hour(i) = str2double(t(end-4:end-3));
end
Hour_bin = spalloc(numel(Dates),24,numel(Dates));
for i=0:23
    Hour_bin(:,i+1) = (Hour == i);
end
%% most-likely hour for each crime
Crimes = unique(Category);
K_Crimes = numel(Crimes);
Crimes_Y = zeros(numel(Category),1);
mostlikely_hours = zeros(K_Crimes,1);
for i =1:K_Crimes
    temp = strcmp(Category,Crimes{i});
    Crimes_Y(temp)=i;
    % Find mostlikely hours
    hour_acc_temp = sum(Hour_bin(temp,:),1);
    [~,ind]=max(hour_acc_temp);
    mostlikely_hours(i)=ind-1;
end
clear hour_acc_temp ind i temp
%% Most-likely crime for each PD
mostlikely_crime_byPD = cell(K_Pd,1);
for i = 1:K_Pd
    temp = Pd_bin(:,i)==1;
    numtemp= mode(Crimes_Y(temp));
    mostlikely_crime_byPD{i}=Crimes{numtemp};
end
clear i numtemp temp
%% concat features
Features = [Day_bin, Pd_bin, Hour_bin];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Script for Part (b) Regularized Multi-class Logistic Regression
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Load features
if ~exist('Features','var')
  load data_SFcrime_train.mat;
  binarizeSFdata;
end

% Get the training/test split
[N,D]=size(Features);
N_train = ceil(0.6*N);
N_test = N - N_train;
X_train = Features(1:N_train,:);
Y_train = Crimes_Y(1:N_train);
X_test = Features(N_train+1:end, :);
Y_test = Crimes_Y(N_train+1:end, :);

% Train Logistic Regression classifier with L2 regularization
lambda = 1000; % regularization param.
c=1e-5; % gradient descent step size
[W_0, te_err_0, tr_err_0, te_logloss, obj_0]=logistic_reg_l2(Y_train, ...
                X_train, Y_test, X_test, lambda, c);
% Generate all the plots
figure;
plot([1:1000], -obj_0,'k-');
xlabel('number of iteration'), ylabel('objective values')
figure;
plot([1:1000],te_err_0,'k-');
xlabel('number of iteration'), ylabel('test CCR')
figure;
plot([1:1000],te_logloss,'k-');
xlabel('number of iteration'),ylabel('test Logloss')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Supporting function for part (b) to pre-process the raw data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% binarize DayOfWeek
DayOfWeek = lower(DayOfWeek); % convert to lower cases
Day_unique = unique(DayOfWeek); % get the unique elements
K_DayOfWeek = numel(Day_unique); % number of possible values
Day_bin = spalloc(numel(DayOfWeek), K_DayOfWeek, numel(DayOfWeek)); %
binarize
for i = 1:K_DayOfWeek
    Day_bin(:,i) = strcmp(DayOfWeek,Day_unique{i});
end
clear i
%% binarize PdDistrict
PdDistrict = lower(PdDistrict);
Pd_unique = unique(PdDistrict);
K_Pd = numel(Pd_unique);
Pd_bin = spalloc(numel(PdDistrict), K_Pd, numel(PdDistrict));
for i = 1:K_Pd
    Pd_bin(:,i) = strcmp(PdDistrict,Pd_unique{i});
end
clear i
%% binarize hour information
Hour = zeros(numel(Dates),1);
for i =1:numel(Dates)
    t = Dates{i};
    Hour(i) = str2double(t(end-4:end-3));
end
clear t
Hour_bin = spalloc(numel(Dates),24,numel(Dates));
for i=0:23
    Hour_bin(:,i+1) = (Hour == i);
end
clear t i
%% numerize Crimes class label
Crimes = unique(Category);
K_Crimes = numel(Crimes);
Crimes_Y = zeros(numel(Category),1);
for i =1:K_Crimes
    temp = strcmp(Category,Crimes{i});
    Crimes_Y(temp)=i;
    % Find mostlikely hours
end
clear i temp
%% concat features
Features = [Day_bin, Pd_bin, Hour_bin];
clear X Y
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [W, test_acc, train_acc, test_logloss,obj]=logistic_reg_l2(y, X,
test_Y, test_X, lambda, c)
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Supporting function for part b):
% Logistic Regression with L2 norm regularization for classification
% Inputs:
% y: training labels
% X: training data
% test_Y: test label
% test_X: test data
% lambda: L2 regularizaion parameter
% c: step-size, chose to be fixed
% the number of iterations is set to 1000 by default
%
% Output:
% test_acc: test accuracies in CCR for each iteration (a 1000 * 1 vector)
% train_acc: training accuracies in CCR for each iteration (a 1000 * 1
% vector)
% test_logloss: test accuracies in Logloss for each iteration (a 1000 * 1
% vector)
% obj: objective function value for each iteration (a 1000 * 1 vector)
% W: dim * K matrix, each column is the weight vector for a class
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % some default setting if not specified
    if ~exist('lambda', 'var')
        lambda=1000;
    end
    if ~exist('c', 'var')
        c=1e-5;
    end
    % set up maximum iteration as stopping criteria
    maxiter=1000;
    K=length(unique(y)); % number of classes
    p=size(X,2);  % dimension of feature
    W=zeros(p,K);
    % saving traing/test accuracy and objective values for each iteration
    test_acc=zeros(maxiter,1);
    train_acc=zeros(maxiter,1);
    test_logloss = zeros(maxiter,1);
    obj=zeros(maxiter,1);
    % start gradient descent algorithm
    iter=1;
    while (iter <= maxiter)
        % Calculate gradient (for w's) in matrix form
        G=log_grad(y,X,W)-lambda*W;
        W=W+c*G;
        % Compute objective function value, the trainning/test accuracy
at the current iteration
```

```
        obj(iter)=log_obj(y, X, W)-lambda/2*sum(W(:).^2);
        train_acc(iter)=cal_te_acc(W, y, X);
        test_acc(iter)=cal_te_acc(W, test_Y, test_X);
        test_logloss(iter)=cal_te_logloss(W,test_Y, test_X);
        % Print itermediate result
        if  mod(iter,10)==0
            fprintf('Iter=%d, Obj=%f, tr_acc=%f, te_acc=%f\n,
te_logloss=%f\n',   iter, obj(iter), train_acc(iter), test_acc(iter),
test_logloss(iter));
        end
        iter=iter+1;
    end
end
```

%%

```
function G=log_grad(y, X, W)
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Supporting function for part b):
% function to calculate gradients (for w's) in logistic regression
% (NLL only, without regularization terms)
% Input:
% y - labels (N * 1 vector)
% X - data (N * Dim data vector), each row represent a data point
% W - the current weight parameters
%   - Dim * K-1 matrix, each column is the weight vector w for one class
%   - (from class 1 up to class K-1)
% Output:
% The gradient of the maximum likelihood estimator for logsitic regression
% G: Dim * K matrix (same dimension as W), each column is the gradient
% vector d_W for one class
    K=size(W,2);
    eXB=exp(X*W);
    s_eXB=1./(sum(eXB, 2)+1);
    eXB=eXB.*repmat(s_eXB, 1, K);
    for k= 1: K
        eXB(:,k)=(y==k)-eXB(:,k);
    end
    G=(X'*eXB);
End
```

%%

```
function obj=log_obj(y, X, W)
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Supporting function for part b):
% Calculate objective function value
% Input:
% y : (labels) N * 1 vector, numberical values from 1 to K, where K is the
% number of classes
% X : (data) N * dim matrix, each row is a data points. dim is the
dimension
% of the features
% W - the current weight parameters
%   - Dim * K matrix, each column is the weight vector w for one class
%
% Output:
% obj : the objective value / Negative Log-Likelihood
    K=size(W,2);
    n=length(y);
    XW=X*W;  % inner products (n * K)
    obj_1=log(sum(exp(XW), 2));
    idx=sub2ind([n,K], [1:n]', y);
    obj_2=zeros(n,1);
    obj_2(I)=XW(idx);
    obj=sum(obj_2-obj_1);
end
```

%%

```
function acc=cal_te_acc(W, te_label, te_data)
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Supporting function for part b):
% Calculate the accuracy of the Logisic regression predictor
% The accuracy is measured using Correct classification rates (CCR)
% assuming K classes. the labels are numbers between 1 and K
% Input:
% W - the weight parameters
%   - Dim * K matrix, each column is the weight vector w for one class
% te_label - the test labels
% te_data - the test data points
% Output:
% acc - CCR
    n_te=size(te_data,1);
    w=te_data*W;
    [~, pred_label]=max(w,[],2);
    acc=1-sum(pred_label~=te_label)/n_te;
end
```

%%

```
function logloss=cal_te_logloss(W, te_label, te_data)
% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.1
% Supporting function for part b):
% Calculate the accuracy of the Logisic regression predictor
% The accuracy is measured using Log loss
% assuming K classes. the labels are numbers between 1 and K
% Input:
% W - the weight parameters
%   - Dim * K-1 matrix, each column is the weight vector w for one class
%   - (from class 1 up to class K-1)
% te_label - the test labels
% te_data - the test data points

    n_te=size(te_data,1);
    w=te_data*W;
    % calculate log-loss
    Idx = sub2ind(size(w),[1:n_te]', te_label);
    obj1 = exp(w(Idx));
    obj2 = sum(exp(w),2);
    logloss = sum(log(obj2./obj1))/n_te;
end
```

%%

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% supporting functions
% returns a sparse matrix where rows are documents and cols are words
function counts=read_docs(fname,ndocs,nwords);
  [docid wordid counts]=textread(fname,'%d %d %d');
  counts=sparse(docid, wordid, counts, ndocs, nwords);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% supporting function for reading and parsing text data
%
words=textread('data/vocabulary.txt','%s');
newsgroup_names=textread('data/newsgrouplabels.txt','%s');
labels_train=textread('data/train.label','%d');
%
labels_test=textread('data/test.label','%d');
%
data_train=read_docs('data/train.data',numel(labels_train),numel(words));
%
data_test=read_docs('data/test.data',numel(labels_test),numel(words));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% supporting function to remove stop words
stopwords=textread('data/stoplist.txt','%s');
stopwords_id = zeros(numel(stopwords),1);
for i =1:numel(stopwords)
    z = strcmp(words, stopwords{i});
    if ~isempty(find(z==1))
        stopwords_id(i) = find(z==1);
    end
end
stopwords_id = stopwords_id(stopwords_id>0);
data_train(:,stopwords_id) = [];
data_test(:, stopwords_id) = [];
words(stopwords_id) = [];
clear z i

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% Script for part (a)

% read documents
textparsing;
removestopwords;
% normalize to get word-frequency vector for each doc.
[ntrain, W]=size(data_train);
[ntest, W]=size(data_test);
data_train= data_train';
data_test= data_test';
for i = 1:ntrain
    data_train(:,i)=data_train(:,i)/sum(data_train(:,i));
end
for i = 1:ntest
    data_test(:,i)=data_test(:,i)/sum(data_test(:,i));
end
data_train= data_train';
data_test= data_test';
% get data from class 1(atheism) and 20(religion) for binary
classification
grp1_idx = find(labels_train==1);
grp2_idx = find(labels_train==20);
idx_train = [grp1_idx;grp2_idx];
data_binary_train = full(data_train(idx_train,:));
labels_binary_train = labels_train(idx_train,:);
grp1_idx = find(labels_test==1);
grp2_idx = find(labels_test==20);
idx_test = [grp1_idx;grp2_idx];
data_binary_test = full(data_test(idx_test,:));
labels_binary_test = labels_test(idx_test,:);
clear grp1_idx grp2_idx idx_train idx_test
%% train SVM
K_cv=5;
CVO = cvpartition(labels_binary_train,'k',K_cv); % 5 fold cross validation
C_values = 2.^[-5:15];
cv_svm_CCR = zeros(K_cv, length(C_values));
for iter_cv = 1: K_cv
    trIdx = CVO.training(iter_cv);
    teIdx = CVO.test(iter_cv);
    for iter_c = 1:length(C_values)
        c = C_values(iter_c);
        mysvm_linear=svmtrain(data_binary_train(trIdx,:),
labels_binary_train(trIdx),...
            'autoscale','false','kernel_function','linear', ...
            'boxconstraint',c);
        y_pred_lin = svmclassify(mysvm_linear,
data_binary_train(teIdx,:));
```

```
        cv_svm_CCR(iter_cv,iter_c)=
sum(labels_binary_train(teIdx)~=y_pred_lin)/sum(teIdx);
    end
end
cv_ccr = 1- sum(cv_svm_CCR,1)/K_cv;
% cv-ccr figure
figure, plot(log2(C_values), cv_ccr,'*k-')
xlabel('regularization parameter C in (log) scale (base 2)')
ylabel('Cross-validation CCR')
%% test
[~,optimal]=max(cv_ccr);
c_star = C_values(optimal);
mysvm_linear=svmtrain(data_binary_train, labels_binary_train,...
            'autoscale','false','kernel_function','linear', ...
            'boxconstraint',c_star);
y_pred_lin = svmclassify(mysvm_linear, data_binary_test);
testCCR = 1 -
sum(y_pred_lin~=labels_binary_test)/length(labels_binary_test)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% Script for part (b):

% clear all, close all, clc
% read documents
textparsing;
removestopwords;
% normalize to get word-frequency vector for each doc.
[ntrain, W]=size(data_train);
[ntest, W]=size(data_test);
data_train= data_train';
data_test= data_test';
for i = 1:ntrain
    data_train(:,i)=data_train(:,i)/sum(data_train(:,i));
end
for i = 1:ntest
    data_test(:,i)=data_test(:,i)/sum(data_test(:,i));
end
data_train= data_train';
data_test= data_test';
% get data from class 1(atheism) and 20(religion) for binary
classification
grp1_idx = find(labels_train==1);
grp2_idx = find(labels_train==20);
idx_train = [grp1_idx;grp2_idx];
data_binary_train = data_train(idx_train,:);
labels_binary_train = labels_train(idx_train,:);

grp1_idx = find(labels_test==1);
grp2_idx = find(labels_test==20);
```

```
idx_test = [grp1_idx;grp2_idx];
data_binary_test = data_test(idx_test,:);
labels_binary_test = labels_test(idx_test,:);
clear grp1_idx grp2_idx idx_train idx_test
%% train SVM
K_cv=5;
CVO = cvpartition(labels_binary_train,'k',K_cv); % 5 fold cross validation
C_values = 2.^[-5:15];
sig_values = 2.^[-13:3];
cv_svm_CCR = zeros(length(C_values),length(sig_values));
%
for iter_cv = 1: K_cv
    trIdx = CVO.training(iter_cv);
    teIdx = CVO.test(iter_cv);
    for iter_c = 1:length(C_values)
        c = C_values(iter_c)
        for iter_sig = 1:length(sig_values)
            rbf_sig=sig_values(iter_sig);
            mysvm_rbf=svmtrain(data_binary_train(trIdx,:),
labels_binary_train(trIdx),...
               'autoscale','false','kernel_function','rbf', ...
               'boxconstraint',c, 'rbf_sigma', rbf_sig);
            y_pred_reb = svmclassify(mysvm_rbf,
data_binary_train(teIdx,:));
            cv_svm_CCR(iter_c,iter_sig)= cv_svm_CCR(iter_c,iter_sig) +
sum(labels_binary_train(teIdx)~=y_pred_reb)/sum(teIdx);
        end
    end
end
cv_ccr = 1-cv_svm_CCR/K_cv;
% cv-ccr plot
figure;
[c,h] = contour(log2(C_values),log2(sig_values),cv_ccr'), clabel(c,h);
colorbar
xlabel('regularization parameter in log scale (base 2)')
ylabel('rbf-sig in log scale (base 2)')
%% test
[~,optimal_sig]=max(max(cv_ccr,[],1))
[~,optimal_c]=max(max(cv_ccr,[],2))
c_star = C_values(optimal_c);
sig_star = sig_values(optimal_sig);
mysvm_rbf=svmtrain(data_binary_train, labels_binary_train,...
            'autoscale','false','kernel_function','rbf', ...
            'boxconstraint',c_star, 'rbf_sigma', sig_star);
y_pred_reb = svmclassify(mysvm_rbf, data_binary_test);
testCCR = 1 -
sum(y_pred_reb~=labels_binary_test)/length(labels_binary_test)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data

% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% Script for part (c)

clear all, close all, clc
% read documents
textparsing;
removestopwords;
% normalize to get word-frequency vector for each doc.
[ntrain, W]=size(data_train);
[ntest, W]=size(data_test);
data_train= data_train';
data_test= data_test';
for i = 1:ntrain
    data_train(:,i)=data_train(:,i)/sum(data_train(:,i));
end
for i = 1:ntest
    data_test(:,i)=data_test(:,i)/sum(data_test(:,i));
end
data_train= data_train';
data_test= data_test';

pos_class = 17;
data_ova_train = data_train;
labels_ova_train = zeros(size(labels_train));
labels_ova_train(labels_train==pos_class)=1;
data_ova_test = data_test;
labels_ova_test = zeros(size(labels_test));
labels_ova_test(labels_test==pos_class)=1;
%% train SVM
K_cv=5;
CVO = cvpartition(labels_ova_train,'k',K_cv); % 5 fold cross validation
C_values = 2.^[-5:15];
cv_svm_CCR = zeros(K_cv, length(C_values));
%
for iter_cv = 1: K_cv
    trIdx = CVO.training(iter_cv);
    teIdx = CVO.test(iter_cv);
    for iter_c = 1:length(C_values)
        c = C_values(iter_c);
        mysvm_linear=svmtrain(data_ova_train(trIdx,:),
labels_ova_train(trIdx),...
            'autoscale','false','kernel_function','linear', ...
            'boxconstraint',c,'kernelcachelimit', 12000);
        y_pred_lin = svmclassify(mysvm_linear, data_ova_train(teIdx,:));
        cv_svm_CCR(iter_cv,iter_c)=
sum(labels_ova_train(teIdx)~=y_pred_lin)/sum(teIdx);
    end
```

```
end
cv_ccr = 1- sum(cv_svm_CCR,1)/K_cv;
% cv-ccr plot
figure
plot(log2(C_values), cv_ccr,'*k-')
xlabel('regularization parameter C in {log} scale (base 2)')
ylabel('Cross-validation CCR')
%% test
[~,optimal]=max(cv_ccr);
c_star = C_values(optimal);
mysvm_linear=svmtrain(data_ova_train, labels_ova_train,...
            'autoscale','false','kernel_function','linear', ...
            'boxconstraint',c_star,'kernelcachelimit', 12000);
y_pred_lin = svmclassify(mysvm_linear, data_ova_test);
testCCR = 1 - sum(y_pred_lin~=labels_ova_test)/length(labels_ova_test)
confusionmat(y_pred_lin,labels_ova_test);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% Script for part (d)

clear all, close all, clc
% read documents
textparsing;
removestopwords;
% normalize to get word-frequency vector for each doc.
[ntrain, W]=size(data_train);
[ntest, W]=size(data_test);
data_train= data_train';
data_test= data_test';
for i = 1:ntrain
    data_train(:,i)=data_train(:,i)/sum(data_train(:,i));
end
for i = 1:ntest
    data_test(:,i)=data_test(:,i)/sum(data_test(:,i));
end
data_train= data_train';
data_test= data_test';

pos_class = 17;
data_ova_train = data_train;
labels_ova_train = zeros(size(labels_train));
labels_ova_train(labels_train==pos_class)=1;

data_ova_test = data_test;
labels_ova_test = zeros(size(labels_test));
labels_ova_test(labels_test==pos_class)=1;

%% train SVM
K_cv=5;
```

```
CVO = cvpartition(labels_ova_train,'k',K_cv); % 5 fold cross validation
C_values = 2.^[-5:15];
cv_svm_precision = zeros(K_cv, length(C_values));
cv_svm_recall = zeros(K_cv, length(C_values));
cv_svm_Fscore = zeros(K_cv, length(C_values));
%
for iter_cv = 1: K_cv
    trIdx = CVO.training(iter_cv);
    teIdx = CVO.test(iter_cv);
    for iter_c = 1:length(C_values)
        % iter_c
        % c = ones(sum(trIdx),1)*C_values(iter_c);
        c = C_values(iter_c)
        mysvm_linear=svmtrain(data_ova_train(trIdx,:),
labels_ova_train(trIdx),...
            'autoscale','false','kernel_function','linear', ...
            'boxconstraint',c,'kernelcachelimit', 12000);
        y_pred_lin = svmclassify(mysvm_linear, data_ova_train(teIdx,:));
        % calculate precision, f-score, etc.
        confmtx = confusionmat(y_pred_lin, labels_ova_train(teIdx));

        cv_svm_recall(iter_cv,iter_c)=
confmtx(2,2)/(confmtx(1,2)+confmtx(2,2));

cv_svm_precision(iter_cv,iter_c)=confmtx(2,2)/(confmtx(2,1)+confmtx(2,2));
        cv_svm_Fscore(iter_cv,iter_c)=
0.5*(1/cv_svm_recall(iter_cv,iter_c) +
1/cv_svm_precision(iter_cv,iter_c));
    end
end
cv_precision = sum(cv_svm_precision,1)/K_cv;
cv_recall = sum(cv_svm_recall,1)/K_cv;
cv_svm_Fscore=1./cv_svm_Fscore;
cv_Fscore = sum(cv_svm_Fscore,1)/K_cv;
% cv plot
figure
hold on
plot(log2(C_values), cv_precision,'*k-')
plot(log2(C_values), cv_recall,'or-')
plot(log2(C_values), cv_Fscore,'+g-')
hold off
xlabel('regularization parameter C in {log} scale (base 2)')
ylabel('Cross-validation Metrics')
legend('Precision','Recall','F-score')
%% test
[~,optimal]=max(cv_recall);
c_star = C_values(optimal);
mysvm_linear=svmtrain(data_ova_train, labels_ova_train,...
            'autoscale','false','kernel_function','linear', ...
            'boxconstraint',c_star,'kernelcachelimit', 12000);
y_pred_lin = svmclassify(mysvm_linear, data_ova_test);
testCCR = 1 - sum(y_pred_lin~=labels_ova_test)/length(labels_ova_test);
confusionmat(y_pred_lin,labels_ova_test);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% Script for part (e)

% clear all, close all, clc
% read documents
textparsing;
removestopwords;
%normalize to get word-frequency vector for each doc.
[ntrain, W]=size(data_train);
[ntest, W]=size(data_test);
data_train= data_train';
data_test= data_test';
for i = 1:ntrain
    data_train(:,i)=data_train(:,i)/sum(data_train(:,i));
end
for i = 1:ntest
    data_test(:,i)=data_test(:,i)/sum(data_test(:,i));
end
data_train= data_train';
data_test= data_test';

%% one-vs.-one multi-class classification
t_train_start = tic;
svm_ovo_repo = cell(20,20);
for class1 =1:20
    for class2 = class1+1:20
        % get data from two classes for binary classification
        grp1_idx = find(labels_train==class1);
        grp2_idx = find(labels_train==class2);
        idx_train = [grp1_idx;grp2_idx];
        data_binary_train = data_train(idx_train,:);
        labels_binary_train = labels_train(idx_train,:);

        grp1_idx = find(labels_test==class1);
        grp2_idx = find(labels_test==class2);
        idx_test = [grp1_idx;grp2_idx];
        data_binary_test = data_test(idx_test,:);
        labels_binary_test = labels_test(idx_test,:);
        clear grp1_idx grp2_idx idx_train idx_test

        % train svms
        svm_ovo_repo{class1,class2}=svmtrain(data_binary_train,
labels_binary_train,...
                    'autoscale','false','kernel_function','linear');
    end
end
t_train = toc(t_train_start);

%% testing
```

```
desicion_pool = zeros(length(labels_test),20*19/2);
iter = 1;
t_test_start = tic;
for class1 = 1:20
    for class2 = class1+1: 20
        desicion_pool(:,iter) = svmclassify(svm_ovo_repo{class1,class2},
data_test);
        iter = iter+1;
    end
end
y_pred_lin = mode(desicion_pool,2);
t_test = toc(t_test_start);
testCCR = 1 - sum(y_pred_lin~=labels_test)/length(labels_test);
confmtx = confusionmat(y_pred_lin, labels_test);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ENG EC 500 B1: Introduction to Learning from Data
% Boston University, Fall Semester, 2015
% Instructor: Prakash Ishwar
% MATLAB Exercise 2, Problem 2.2
% Script for part (f)

clear all, close all, clc
% read documents
textparsing;
removestopwords;
% normalize to get word-frequency vector for each doc.
[ntrain, W]=size(data_train);
[ntest, W]=size(data_test);
data_train= data_train';
data_test= data_test';
for i = 1:ntrain
    data_train(:,i)=data_train(:,i)/sum(data_train(:,i));
end
for i = 1:ntest
    data_test(:,i)=data_test(:,i)/sum(data_test(:,i));
end
data_train= data_train';
data_test= data_test';
%% one-vs.-one multi-class classification
svm_ovo_repo = cell(20,20);
t_train_start = tic;
for class1 =1:20
    for class2 = class1+1:20
        % get data from two classes for binary classification
        grp1_idx = find(labels_train==class1);
        grp2_idx = find(labels_train==class2);
        idx_train = [grp1_idx;grp2_idx];
        data_binary_train = data_train(idx_train,:);
        labels_binary_train = labels_train(idx_train,:);

        grp1_idx = find(labels_test==class1);
        grp2_idx = find(labels_test==class2);
        idx_test = [grp1_idx;grp2_idx];
        data_binary_test = data_test(idx_test,:);
        labels_binary_test = labels_test(idx_test,:);
        clear grp1_idx grp2_idx idx_train idx_test

        % train svms
        svm_ovo_repo{class1,class2}=svmtrain(data_binary_train,
labels_binary_train,...
                    'autoscale','false','kernel_function','rbf');
    end
end
t_train = toc(t_train_start);
%% test
desicion_pool = zeros(length(labels_test),20*19/2);
iter = 1;
t_test_start = tic;
for class1 = 1:20
    for class2 = class1+1: 20
        desicion_pool(:,iter) = svmclassify(svm_ovo_repo{class1,class2},
data_test);
        iter = iter+1;
    end
end
y_pred_rbf = mode(desicion_pool,2);
t_test = toc(t_test_start);
testCCR = 1 - sum(y_pred_rbf~=labels_test)/length(labels_test);
confmtx = confusionmat(y_pred_rbf, labels_test);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```