實驗3

實驗代碼：

```
function [params] = nbayes_learn(traindata, trainlabels)

nclasses = 2;
[n, d] = size(traindata);

% Estimate class probabilities and conditional probabilities
for i=1:nclasses
    index = find(trainlabels==i);

    data_ci = traindata(index,:);
    ni = size(index, 1);

    params(1).classprobs(i) = ni/n;  % estimate the probability of class i

    for j = 1:d
        data_jcol = data_ci(:,j);
        for k=1:2
            data_k = find(data_jcol == k);
            % estimate prob(variable j = k | class i)
            params(j).cprobs(k,i) = (size(data_k, 1) + 0.5)/(ni + 1);
        end
    end
end

% Estimate the probabilities p(x_j = value_k)
for j = 1:d
    for k=1:2;
        index = find(traindata(:,j)==k);
        % estimate  prob(variable x_j = k)
        params(j).mprobs(k) = size(index, 1)/n;
    end
end

end


function [predictions] = nbayes_predict(params, testdata)

nclasses = 2;
[ntest, d] = size(testdata);

predictions = zeros(ntest, 1);

% for each of the test data points
for m=1:ntest
    x = testdata(m,:);
    predict_prob = zeros(1,2);
    % for each class value calculate log[ p(x|c) p(c) ]
    for classi=1:nclasses;
        pci = params(1).classprobs(classi);
        for varj=1:d
            predict_prob(classi) = predict_prob(classi) ...
                + log(params(varj).cprobs(x(1,varj), classi));
        end
        predict_prob(classi) = predict_prob(classi) + log(pci);
    end
    % select the maximum value over all classes as the predicted class
    % store the class prediction for each test data point
    [~, predictions(m)] = max(predict_prob);
end

end
```

自動化腳本：

```
features_mat = dlmread('spam_features.txt');
labels_mat = dlmread('spam_labels.txt');

[row, ~] = size(labels_mat);
trainamount = floor(row/2);

traindata = features_mat(1:trainamount, :);
trainlabels = labels_mat(1:trainamount, :);
testdata = features_mat(trainamount+1:row, :);
testlabels = labels_mat(trainamount+1:row, :);

params = nbayes_learn(traindata, trainlabels);
predictions = nbayes_predict(params, testdata);

temp = abs(predictions - testlabels);
rate = sum(temp)/size(temp, 1);

fprintf('Training Data size = %d\n', trainamount);
fprintf('Accuracy = %f\n', 1-rate);
```

實驗結果：

```
Training Data size = 2300
Accuracy = 0.884833
```