朴素贝叶斯代码说明

作者：蔡中恒
2018.07.14

一、原理讲解

1、 朴素贝叶斯具体原理，详见李航《统计学习方法》第 4 章朴素贝叶斯法。

二、算法思路

本次使用 matlab 来写原型代码，训练数据采用《统计学习方法》上第 50 页的表 4.1，代码同时支持朴素贝叶斯和贝叶斯估计，并支持 lamda 可调节。

三、代码讲解

此区域进行初始赋值，根据书上给出的表格，分别赋值到相应数组中。

```
%{
name   :  navie bayes demo
author :  CaiZhongheng

date          version         record
2018.07.14    v1.0            init
%}

clc;
clear;
close all;
%% setting
lamda_laplace  = 1; % 0: naive bayes; 1: naive bayes and laplace smoothing
%% training data
x_feature_num  = 2; % 特征数目
% line1: x1 特征1 取值范围：1 2 3
X1_num   = [1;2;3];
% line2: x2 特征2 取值范围：0:S, 1:M, 2:L
X2_num   = [0;1;2];
% line3: Y  分类  -1 1
Y_num    = [-1;1];
training_data_x     = [1 1 1 1 1 2 2 2 2 2 3 3 3 3 3;...
                       0 1 1 0 0 0 1 1 2 2 2 1 1 2 2;];
training_data_class = [-1 -1 1 1 -1 -1 -1 1 1 1 1 1 1 1 -1];

test_data = [2,0]';
```

初始化贝叶斯估计的矩阵，数组当中每个元素代表的概率也写在注释中。目前该代码只支持每种特征取值一样的数据，如果不同特征的取值可能性不同（比如 x1 取值范围为 1 到 5，而 x2 的取值范围是 1 到 6，本代码运行就要报错），需要单独再修改代码。

```
27
28 —     max_x_num         = max(length(X1_num),length(X2_num));
29 —     P_test_data_class = zeros(length(Y_num),1);%初始化后验概率矩阵
30 —     P_class           = zeros(length(Y_num),1);%初始化分类概率矩阵
31 —     P_bayes           = zeros(size(training_data_x,1)*length(Y_num),max_x_num);
32
33       % create the bayes matrix
34       % P_class = [P(Y=-1); P(Y=1)];
35       % P_bayes = [P(x1=1|Y=-1), P(x1=2|Y=-1), P(x1=3|Y=-1);
36       %            P(x2=S|Y=-1), P(x2=M|Y=-1), P(x2=L|Y=-1);
37       %            P(x1=1|Y=1),  P(x1=2|Y=1),  P(x1=3|Y=1);
38       %            P(x2=S|Y=1),  P(x2=M|Y=1),  P(x2=L|Y=1);];
39
40 —     if(length(X1_num)~=length(X2_num))
41 —         error('Please check the X1_num and X2_num!!!');
42 —     else
43 —     end
```

接下来是计算并填充概率矩阵。这个地方其实按照书上给出的步骤，一行一行单独写代码也是可以的，不过这里为了提升代码的通用性，所以就使用了 for 循环自动根据矩阵、特征、分类的大小，从数据中自动提取出来计算概率。并使用 matlab 自带的 find 函数来搜索数据中满足该特征的个数，再加上 intersect 函数来取两个数组的交集，方便计算条件概率。

```
44       %% calc P matrix
45 —     for y_idx=1:length(Y_num)
46 —         P_class(y_idx)            = (length(find(training_data_class==Y_num(y_idx)))+lamda_laplace)...
47               /(length(training_data_class)+length(Y_num)*lamda_laplace); % P(Y=-1) or P(Y=1)
48 —     end
49
50 —     for y_idx=0:(length(Y_num)*x_feature_num-1)
51 —         for x_idx=1:max_x_num
52 —             feature_idx           = mod(y_idx,2)+1;% 特征编号
53 —             class_idx             = floor(y_idx/2)+1;% 分类编号
54 —             tmp_feature_num       = eval(['X' num2str(feature_idx,'%d') '_num']);
55 —             P_bayes(y_idx+1,x_idx) = (length(intersect(find(training_data_x(feature_idx,:)==tmp_feature_num(x_idx)),find(training_data_class==Y_num(class_idx))))+lamda_laplace)...
56                   /(length(find(training_data_class==Y_num(class_idx)))+length(tmp_feature_num)*lamda_laplace);
57 —         end
58 —     end
```

最后计算测试数据属于每个分类时候的后验概率，然后寻找最大的后验概率所在的分类，输出。

```
59
60       % using test data to calc the Possibility
61       % if test data is in Y=-1, then the P(Y=-1|X=test_data) = arg max P(Y)*prod(P(xi=test_data(i)|Y=-1))
62 —     for y_idx=1:length(Y_num)
63 —         x1_idx                  = find(X1_num==test_data(1));
64 —         x2_idx                  = find(X2_num==test_data(2));
65 —         P_test_data_class(y_idx) = P_class(y_idx)*P_bayes((y_idx-1)*size(training_data_x,1)+1,x1_idx)*P_bayes((y_idx-1)*size(training_data_x,1)+2,x2_idx);
66 —     end
67
68 —     [P_out, class_out] = max(P_test_data_class);
69 —     fprintf('The class of test data is Y = %d.\n', Y_num(class_out));
70 —     fprintf('The max Prossibility of test data is %f.\n', P_out);
```

四、后续思考

1. 朴素贝叶斯和拉普拉斯平滑本身的算法并不复杂，使用朴素贝叶斯的难点在于源头的数据清洗和 lamda 的参数调节。数据清洗需要多考虑使用正则表达式等手段过滤掉无用的信息，这样得到的输入数据才比较准确。
2. 朴素贝叶斯的前提条件是，不同特征之间的取值分布独立。比方说一件 T 恤的特征有性别、尺寸，如果采用朴素贝叶斯估计，就会强行认为性别和尺寸完全独立。实际上性别和尺寸并不是完全独立。如果碰到特征之间相关性比较强的场景，朴素贝叶斯方法就会出现较大偏差。这个手可以考虑用 k-mean 或者 KNN 来做分类。