云南大学数学与统计学院  
《数据挖掘与决策支持实验》上机实践报告

|  |  |  |
| --- | --- | --- |
| **课程名称**：数据挖掘与决策支持实验 | **年级**：2015级 | **上机实践成绩**： |
| **指导教师**：彭程 | **姓名**：刘鹏 | **专业：**信息与计算科学 |
| **上机实践名称**：对iris数据进行贝叶斯分类 | **学号**：20151910042 | **上机实践日期**：2018-07-05 |
| **上机实践编号**：04 | **组号**： |  |

# 一、实验目的

学习使用R语言进行变量选择。

# 二、实验内容

对iris数据进行贝叶斯分类。

# 三、实验平台

Windows 10 Pro 1803；

Microsoft© Visual Studio 2017 Enterprise。

Version 1.1.442 – © 2009-2018 RStudio, Inc.

# 四、算法设计

贝叶斯方法是一种分类法。数据并不是总体或待建模系统的唯一可用的信息资源。贝叶斯方法提供了一套将这些外部信息融入数据分析过程的原理方法。这个过程先给出待分析数据集的概率分布。因为这个分布在给出时没有考虑任何数据，所以称为先验分布（prior distribution）。新的数据集将先验分布修正后得到后验分布（posterior distribution）。进行这个修正的基本工具就是贝叶斯定理。

设是一个类标号未知的数据样本，为某种假定：数据样本属于某特定的类。要求确定，即给定了观测数据样本，假定的概率。其实这就是贝叶斯公式的一种具体形式。表示给定数据集之后，我们对假设成立的后验概率。后验概率比先验概率含有更多的信息。

贝叶斯定理可以表示如下：

# 五、程序代码

5.1 程序描述

5.2 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260 | # Load libraries  **import** pandas  **import** numpy **as** np  **import** scipy**.**stats **as** stats  **import** matplotlib**.**pyplot **as** plt  **from** sklearn **import** model\_selection # 模型比较和选择包  **from** sklearn**.**naive\_bayes **import** GaussianNB  **class** **Bayes\_Test():**  # 读取样本 数据集  **def** load\_dataset**(**self**):**  url **=** 'Iris.csv'  names **=** **[**'sepal-length'**,** 'sepal-width'**,** 'petal-length'**,** 'petal-width'**,** 'class'**]**  dataset **=** pandas**.**read\_csv**(**url**,** names**=**names**)**  **return** dataset  # 提取样本特征集和类别集 划分训练/测试集  **def** split\_out\_dataset**(**self**,** dataset**):**  array **=** dataset**.**values # 将数据库转换成数组形式  X **=** array**[:,** 0**:**4**].**astype**(**float**)** # 取特征数值列  Y **=** array**[:,** 4**]** # 取类别列  validation\_size **=** 0.20 # 验证集规模  seed **=** 7  # 分割数据集 测试/验证  X\_train**,** X\_validation**,** Y\_train**,** Y\_validation **=** \  model\_selection**.**train\_test\_split**(**X**,** Y**,** test\_size**=**validation\_size**,** \  random\_state**=**seed**)**  **return** X\_train**,** X\_validation**,** Y\_train**,** Y\_validation  """第一步：划分样本集"""  # 提取样本 特征  **def** split\_out\_attributes**(**self**,** X**,** Y**):**  # 提取 每个类别的不同特征  # c1 第一类的特征数组  c1\_1 **=** c1\_2 **=** c1\_3 **=** c1\_4 **=** **[]**  c2\_1 **=** c2\_2 **=** c2\_3 **=** c2\_4 **=** **[]**  c3\_1 **=** c3\_2 **=** c3\_3 **=** c3\_4 **=** **[]**  **for** i **in** range**(**len**(**Y**)):**  **if** **(**Y**[**i**]** **==** 'Iris-setosa'**):**  c1\_1**.**append**(**X**[**i**,** 0**])**  c1\_2**.**append**(**X**[**i**,** 1**])**  c1\_3**.**append**(**X**[**i**,** 2**])**  c1\_4**.**append**(**X**[**i**,** 3**])**  **elif** **(**Y**[**i**]** **==** 'Iris-versicolor'**):**  # c2 第二类的特征数组  c2\_1**.**append**(**X**[**i**,** 0**])**  c2\_2**.**append**(**X**[**i**,** 1**])**  c2\_3**.**append**(**X**[**i**,** 2**])**  c2\_4**.**append**(**X**[**i**,** 3**])**  **elif** **(**Y**[**i**]** **==** 'Iris-virginica'**):**  # c3 第三类的特征数组  c3\_1**.**append**(**X**[**i**,** 0**])**  c3\_2**.**append**(**X**[**i**,** 1**])**  c3\_3**.**append**(**X**[**i**,** 2**])**  c3\_4**.**append**(**X**[**i**,** 3**])**  **else:**  **pass**  **return** **[**c1\_1**,** c1\_2**,** c1\_3**,** c1\_4**,**  c2\_1**,** c2\_2**,** c2\_3**,** c2\_4**,**  c3\_1**,** c3\_2**,** c3\_3**,** c3\_4**]**  """因为符合多变量正态分布，所以需要(μ，∑)两个样本参数"""  """第二步：计算样本期望μ和样本方差s"""  # 计算样本期望  **def** cal\_mean**(**self**,** attributes**):**  c1\_1**,** c1\_2**,** c1\_3**,** c1\_4**,** c2\_1**,** c2\_2**,** c2\_3**,** c2\_4**,** c3\_1**,** c3\_2**,** c3\_3**,** c3\_4 **=** attributes  # 第一类的期望值μ  e\_c1\_1 **=** np**.**mean**(**c1\_1**)**  e\_c1\_2 **=** np**.**mean**(**c1\_2**)**  e\_c1\_3 **=** np**.**mean**(**c1\_3**)**  e\_c1\_4 **=** np**.**mean**(**c1\_4**)**  # 第二类的期望值μ  e\_c2\_1 **=** np**.**mean**(**c2\_1**)**  e\_c2\_2 **=** np**.**mean**(**c2\_2**)**  e\_c2\_3 **=** np**.**mean**(**c2\_3**)**  e\_c2\_4 **=** np**.**mean**(**c2\_4**)**  # 第三类的期望值μ  e\_c3\_1 **=** np**.**mean**(**c3\_1**)**  e\_c3\_2 **=** np**.**mean**(**c3\_2**)**  e\_c3\_3 **=** np**.**mean**(**c3\_3**)**  e\_c3\_4 **=** np**.**mean**(**c3\_4**)**  **return** **[**e\_c1\_1**,** e\_c1\_2**,** e\_c1\_3**,** e\_c1\_4**,**  e\_c2\_1**,** e\_c2\_2**,** e\_c2\_3**,** e\_c2\_4**,**  e\_c3\_1**,** e\_c3\_2**,** e\_c3\_3**,** e\_c3\_4**]**  # 计算样本方差  **def** cal\_var**(**self**,** attributes**):**  c1\_1**,** c1\_2**,** c1\_3**,** c1\_4**,** c2\_1**,** c2\_2**,** c2\_3**,** c2\_4**,** c3\_1**,** c3\_2**,** c3\_3**,** c3\_4 **=** attributes  # 第一类的方差var  var\_c1\_1 **=** np**.**var**(**c1\_1**)**  var\_c1\_2 **=** np**.**var**(**c1\_2**)**  var\_c1\_3 **=** np**.**var**(**c1\_3**)**  var\_c1\_4 **=** np**.**var**(**c1\_4**)**  # 第二类的方差s  var\_c2\_1 **=** np**.**var**(**c2\_1**)**  var\_c2\_2 **=** np**.**var**(**c2\_2**)**  var\_c2\_3 **=** np**.**var**(**c2\_3**)**  var\_c2\_4 **=** np**.**var**(**c2\_4**)**  # 第三类的方差s  var\_c3\_1 **=** np**.**var**(**c3\_1**)**  var\_c3\_2 **=** np**.**var**(**c3\_2**)**  var\_c3\_3 **=** np**.**var**(**c3\_3**)**  var\_c3\_4 **=** np**.**var**(**c3\_4**)**  **return** **[**var\_c1\_1**,** var\_c1\_2**,** var\_c1\_3**,** var\_c1\_4**,**  var\_c2\_1**,** var\_c2\_2**,** var\_c2\_3**,** var\_c2\_4**,**  var\_c3\_1**,** var\_c3\_2**,** var\_c3\_3**,** var\_c3\_4**]**  # 计算先验概率P(Y=ck)  **def** cal\_prior\_probability**(**self**,** Y**):**  a **=** b **=** c **=** 0  **for** i **in** Y**:**  **if** **(**i **==** 'Iris-setosa'**):**  a **+=** 1  **elif** **(**i **==** 'Iris-versicolor'**):**  b **+=** 1  **elif** **(**i **==** 'Iris-virginica'**):**  c **+=** 1  **else:**  **pass**  pa **=** a **/** len**(**Y**)**  pb **=** b **/** len**(**Y**)**  pc **=** c **/** len**(**Y**)**  **return** pa**,** pb**,** pc  # 计算后验概率P(Y=ck|X)=P(X|Y=ck)\*P(Y=ck)/∑  **def** cal\_posteriori\_probability**(**self**,** X**,** Y**,** p**,** means**,** vars**):**  pa**,** pb**,** pc **=** p  e\_c1\_1**,** e\_c1\_2**,** e\_c1\_3**,** e\_c1\_4**,** e\_c2\_1**,** e\_c2\_2**,** e\_c2\_3**,** \  e\_c2\_4**,** e\_c3\_1**,** e\_c3\_2**,** e\_c3\_3**,** e\_c3\_4 **=** means  var\_c1\_1**,** var\_c1\_2**,** var\_c1\_3**,** var\_c1\_4**,** var\_c2\_1**,** var\_c2\_2**,** \  var\_c2\_3**,** var\_c2\_4**,** var\_c3\_1**,** var\_c3\_2**,** var\_c3\_3**,** var\_c3\_4 **=** vars  **print(**'p:'**,** p**)**  **print(**'means:'**,** means**)**  **print(**'vars:'**,** vars**)**  # 分解四维输入向量X=[X1，X2，X3，X4]为4个一维正态分布函数  X1 **=** X**[:,** 0**]**  X2 **=** X**[:,** 1**]**  X3 **=** X**[:,** 2**]**  X4 **=** X**[:,** 3**]**  # 分类正确数/分类错误数=>计算正确率  true\_test **=** 0  false\_test **=** 0  # 遍历训练整个输入空间，计算后验概率并判决  **for** i **in** range**(**len**(**X1**)):**  # 计算后验概率=P(X|Y=C1)P(Y=C1)  P\_1 **=** stats**.**norm**.**pdf**(**X1**[**i**],** e\_c1\_1**,** var\_c1\_1**)** **\*** stats**.**norm**.**pdf**(**X2**[**i**],** \  e\_c1\_2**,** var\_c1\_2**)** **\*** stats**.**norm**.**pdf**(**  X3**[**i**],** e\_c1\_3**,**  var\_c1\_3**)** **\*** stats**.**norm**.**pdf**(**  X4**[**i**],** e\_c1\_4**,** var\_c1\_4**)** **\*** pa  # 计算后验概率=P(X|Y=C2)P(Y=C2)  P\_2 **=** stats**.**norm**.**pdf**(**X1**[**i**],** e\_c2\_1**,** var\_c2\_1**)** **\*** stats**.**norm**.**pdf**(**X2**[**i**],** \  e\_c2\_2**,** var\_c2\_2**)** **\*** stats**.**norm**.**pdf**(**  X3**[**i**],** e\_c2\_3**,**  var\_c2\_3**)** **\*** stats**.**norm**.**pdf**(**  X4**[**i**],** e\_c2\_4**,** var\_c2\_4**)** **\*** pb  # 计算后验概率=P(X|Y=C3)P(Y=C3)  P\_3 **=** stats**.**norm**.**pdf**(**X1**[**i**],** e\_c3\_1**,** var\_c3\_1**)** **\*** stats**.**norm**.**pdf**(**X2**[**i**],** \  e\_c3\_2**,** var\_c3\_2**)** **\*** stats**.**norm**.**pdf**(**  X3**[**i**],** e\_c3\_3**,**  var\_c3\_3**)** **\*** stats**.**norm**.**pdf**(**  X4**[**i**],** e\_c3\_4**,** var\_c3\_4**)** **\*** pc  # 计算判别函数，选取概率最大的类  max\_P **=** max**(**P\_1**,** P\_2**,** P\_3**)**  # 输出分类结果，并检测正确率  **if** **(**max\_P **==** P\_1**):**  **if** **(**Y**[**i**]** **==** 'Iris-setosa'**):**  **print(**'分为第一类，正确'**)**  true\_test **+=** 1  **else:**  **print(**'分为第一类，错误'**)**  false\_test **+=** 1  **elif** **(**max\_P **==** P\_2**):**  **if** **(**Y**[**i**]** **==** 'Iris-versicolor'**):**  **print(**'分为第二类，正确'**)**  true\_test **+=** 1  **else:**  **print(**'分为第二类，错误'**)**  false\_test **+=** 1  **elif** **(**max\_P **==** P\_3**):**  **if** **(**Y**[**i**]** **==** 'Iris-virginica'**):**  **print(**'分为第三类，正确'**)**  true\_test **+=** 1  **else:**  **print(**'分为第三类，错误'**)**  false\_test **+=** 1  **else:**  **print(**'未分类'**)**  false\_test **+=** 1  # 打印分类正确率  **print(**'训练正确率为:'**,** **(**true\_test **/** **(**true\_test **+** false\_test**)))**  # 模板方法对照  **def** cal\_dataset**(**self**,** X\_train**,** Y\_train**):**  # Test options and evaluation metric  seed **=** 7  scoring **=** 'accuracy'  # Check Algorithms  model **=** GaussianNB**()**  name **=** 'bayes classifier'  # 建立K折交叉验证 10倍  kfold **=** model\_selection**.**KFold**(**n\_splits**=**10**,** random\_state**=**seed**)**  # cross\_val\_score() 对数据集进行指定次数的交叉验证并为每次验证效果评测  cv\_results **=** \  model\_selection**.**cross\_val\_score**(**model**,** X\_train**,** Y\_train**,** \  cv**=**kfold**,** scoring**=**scoring**)**  results **=** cv\_results  msg **=** "%s: %f (%f)" **%** **(**name **+** '精度'**,** cv\_results**.**mean**(),** cv\_results**.**std**())**  **print(**msg**)**  # Show Algorithms  dataresult **=** pandas**.**DataFrame**(**results**)**  dataresult**.**plot**(**title**=**'Bayes accuracy analysis'**,** kind**=**'density'**,** \  subplots**=True,** layout**=(**1**,** 1**),** sharex**=False,**  sharey**=False)**  dataresult**.**hist**()**  plt**.**show**()**  bayes **=** Bayes\_Test**()**  dataset **=** bayes**.**load\_dataset**()**  # 划分训练集 测试集  X\_train**,** X\_validation**,** Y\_train**,** Y\_validation **=** bayes**.**split\_out\_dataset**(**dataset**)**  **print(**'得到的X\_train'**,** X\_train**)**  **print(**'得到的Y\_train'**,** Y\_train**)**  # 分割属性--训练集  attributes **=** bayes**.**split\_out\_attributes**(**X\_train**,** Y\_train**)**  **print(**'得到的训练集'**,** attributes**)**  # 计算期望--训练集  means **=** bayes**.**cal\_mean**(**attributes**)**  **print(**'得到的means'**,** means**)**  # 计算方差--训练集  vars **=** bayes**.**cal\_var**(**attributes**)**  **print(**'得到的vars'**,** vars**)**  # 计算先验概率--训练集  prior\_p **=** bayes**.**cal\_prior\_probability**(**Y\_train**)**  # 验证分类准确性--测试集  bayes**.**cal\_posteriori\_probability**(**X\_train**,** Y\_train**,** prior\_p**,** means**,** vars**)**  # 模板方法--性能对比  bayes**.**cal\_dataset**(**X\_validation**,** Y\_validation**)** |

程序代码 1

# 六、运行结果

运行结果 1 （经过了反相处理）

代码分析

# 七、实验体会

# 八、参考文献

<https://zhuanlan.zhihu.com/p/29431399>

<http://www.cnblogs.com/leoo2sk/archive/2010/09/17/naive-bayesian-classifier.html>

<http://www.ruanyifeng.com/blog/2013/12/naive_bayes_classifier.html>