**数据挖掘课程**

**期末大作业报告**

**实验题目：基于UCI数据集的分类算法应用及其分析**

姓 名：王梓 24320172203211

唐玉萧 24320172203204

周兴 30320162200103

完成时间：5. 27

版 本：V4.0

目录

[一、数据预处理 3](#_Toc42260971)

[1、mfeat 3](#_Toc42260972)

[2、cmc 4](#_Toc42260973)

[3、Haberman 5](#_Toc42260974)

[4、Sonar 6](#_Toc42260975)

[二、模型介绍 7](#_Toc42260976)

[1、决策树 7](#_Toc42260977)

[2、KNN 7](#_Toc42260978)

[3、贝叶斯 7](#_Toc42260979)

[4、CNN卷积神经网络 8](#_Toc42260980)

[5、BP神经网络 8](#_Toc42260981)

[6、线性回归 8](#_Toc42260982)

[三、训练过程 8](#_Toc42260983)

[1、决策树、KNN和贝叶斯 8](#_Toc42260984)

[2、CNN卷积神经网络 18](#_Toc42260985)

[3、BP神经网络 22](#_Toc42260986)

[4、线性回归 25](#_Toc42260987)

[5、模型融合 27](#_Toc42260988)

[四、结果分析 29](#_Toc42260989)

[1、mfeat数据集 29](#_Toc42260990)

[2、cmc数据集 30](#_Toc42260991)

[3、Haberman数据集 31](#_Toc42260992)

[4、Sonar数据集 32](#_Toc42260993)

[5、模型融合十折交叉验证 34](#_Toc42260994)

# 组员分工

王梓：负责数据预处理，所有中间数据生成，模型选取，报告整合。

唐玉萧：负责所有机器学习相关的模型的训练，优化，包括knn,决策树，贝叶斯，模型融合等。

周兴：负责深度学习相关的模型的训练，优化，包括cnn,bp神经网络，线性模型，模型融合等。

# 二、数据预处理

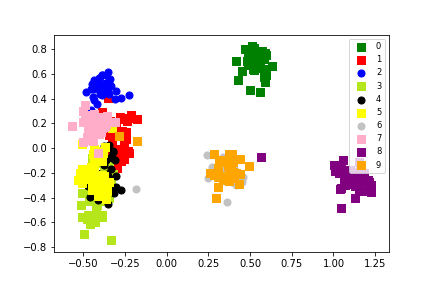
本次实验共有四个数据集，分别是mfeat，cmc，Haberman和sonar,下面分别对四个数据集以及他们的预处理进行介绍。

## 1、mfeat

**数据介绍**：

数据主要是手写体识别的图像特征，一共分为三个文件，分别从手写体图像的pix像素点，图像zer计算方法，图像的五个特征方面提取数据。

**数据二维点图：**

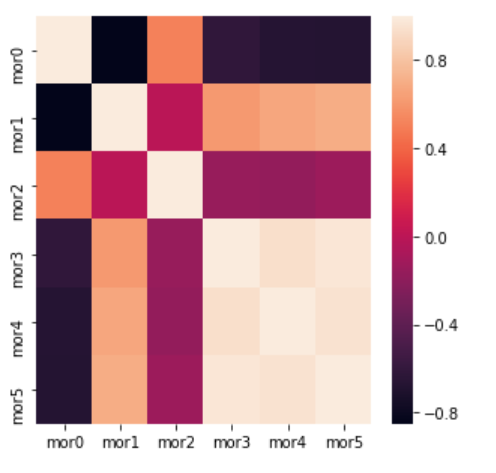
****

**预处理方法：**

本次实验包含两种预处理方法：

* 对三个数据表进行降维并合并：

对于mfeat-mor-left数据表，进行读取后我们得到各个特征之间的关联度，用热力图进行表示。



可以看出在热力图种mor3,mor4,mor5数据列关联较紧密，因此我们在3，4，5种取3进行降维。

对于mfeat-pix-left, mfeat-zer-left由于数据维数较大，我们采用lda降维的方法，因为lda降维的维数和数据的标签数有关，我们最终将两个数据都降为9维进行模型的训练。

* 单独对pix数据表进行训练。

因为在后期的训练中发现，加入right数据后正确率明显降低，而只采用pix进行训练则准确率较高，则后面我们选择采用未处理的pix数据进行训练。

数据说明如下：

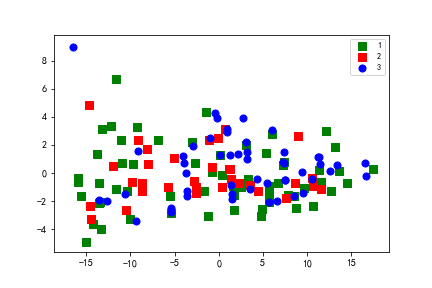
|  |  |
| --- | --- |
| 数据集 | 数据说明 |
| mfeat | 2 x 3窗口中的平均240像素；15 x 16像素；240个属性，共1600条数据 |
| pix | 47个泽尼克时刻；47个属性，共1600条数据 |
| mor | 6个形态特征，共1600条数据 |

## 2、cmc

**数据介绍**：

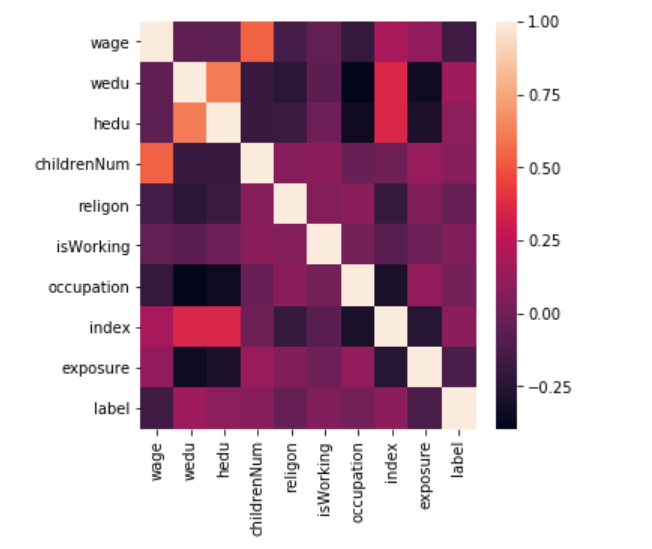
数据集主要是1987年进行的一项避孕措施相关的研究的子集，收集妻子年龄，孩子年龄，丈夫受教育程度，妻子宗教等数据，对夫妻的避孕方法进行研究，其中标签的含义1表示没有使用避孕措施，2表示使用长期避孕措施，3表示使用短期避孕措施。

**数据二维点图：**



**数据预处理：**

首先对数据特征的关联度热力图做具体分析：



可以看出丈夫和妻子的教育关联度较高，其他数据关联度较低，故不作增减数据维度的操作。

数据的大多数据列为标签列：

|  |  |  |  |
| --- | --- | --- | --- |
| 数据项名称 | 数据项解释 | 数据取值 | 数据是否为离散值 |
| Wife age | 妻子年龄（数字） | 10——100 | 否 |
| Wife education | 妻子教育（分类） | 1，2，3，4 | 是 |
| Husband education | 丈夫教育（分类） | 1，2，3，4 | 是 |
| Children Num | 出生儿童数（数字） | 10——100 | 否 |
| religion | 妻子的宗教信仰（二进制） | 0，1 | 是 |
| Is working | 妻子现在是否工作（二进制） | 0，1 | 是 |
| occupation | 丈夫的职业（分类） | 1，2，3，4 | 是 |
| index | 生活水平指数（分类） | 1，2，3，4 | 是 |
| Media exposure | 媒体曝光（二进制） | 0，1 | 是 |

只有妻子年龄和孩子数量为连续值且数值较大，可能会影响最终的结果，因为数据不可能符合高斯分布，所以我们使用min-max进行标准化处理。

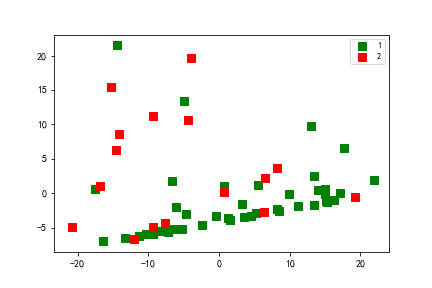
对于1，2，3，4等取值范围的特征，因为1代表low,2,3,4代表high,因此可以对这些数据行进行处理，处理为1，2的布尔值。

## 3、Haberman

**数据介绍：**

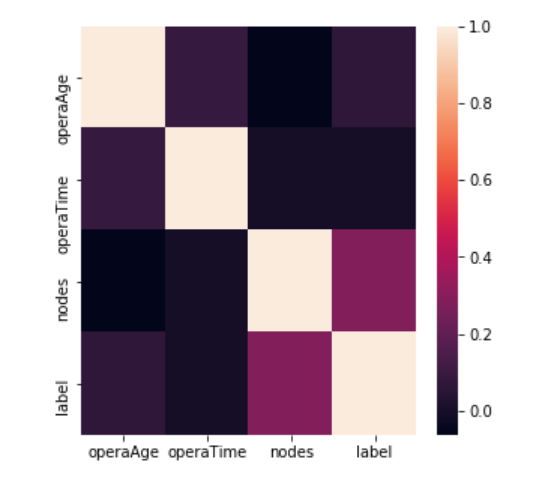
数据集主要是1958年到1970年医院进行的一项乳腺癌患者术后存活率的调查，通过收集患者年龄，患者的手术时间，检测到的阳性腋窝结节数，来判断患者的存活时间，数据的标签为1能活5年以上，2表示患者在5年内死去。

**数据二维点图：**



**数据预处理**

先画出数据项的热力图：



数据特征联系并不紧密，数据的所有特征都为连续值，因此我们使用z-score和min-max进行数据的预处理。

数据说明如下：

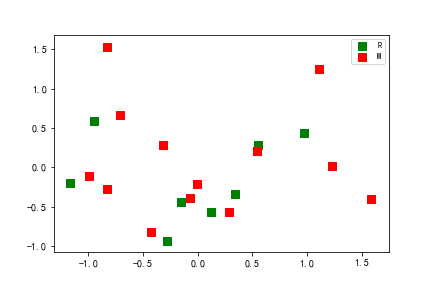
|  |  |  |  |
| --- | --- | --- | --- |
| 数据项名称 | 数据项解释 | 数据取值 | 数据是否为离散值 |
| operaAge | 手术时患者年龄（数字） | 30——83 | 否 |
| operaTime | 病人的手术年份（数字） | 58——69 | 否 |
| nodes | 腋窝淋巴结阳性数（数字） | 0——52 | 否 |

## 4、Sonar

**数据介绍：**

sonar数据集主要是通过声呐获得的大约111种模式，每个模式包括60个数字，范围在0~1之间，数据的标签为R表示岩石，M表示金属矿物。

**数据二维点图：**



**数据预处理：**

因为数据的范围基本在0~1之间，所以不需要进行预处理方法。比较lda降维后和lda降维前的结果。

# 二、模型介绍

## 1、决策树

决策树是通过一系列规则对数据进行分类的过程。它提供一种在什么条件下会得到什么值的类似规则的方法。决策树分为分类树和回归树两种，分类树对离散变量做决策树，回归树对连续变量做决策树。

## 2、KNN

kNN算法的核心思想是如果一个样本在特征空间中的k个最相邻的样本中的大多数属于某一个类别，则该样本也属于这个类别，并具有这个类别上样本的特性。该方法在确定分类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。

## 3、贝叶斯

朴素贝叶斯分类模型要做的事就是在先验概率的基础上将数据集归为n个标签中后验概率最大的标签(基于最小错误率贝叶斯决策原则).

由于数据集的特征个数m和分类标签总数n是固定不变的, 即贝叶斯定理中分母不变, 所以要求最大值, 只需求出分子中的最大值, 即下式中的最大值

https://img2018.cnblogs.com/blog/1469712/201811/1469712-20181111203747763-1403326197.png

## 4、CNN卷积神经网络

积神经网络是一类包含卷积计算且具有深度结构的前馈神经网络，是深度学习的代表算法之一。卷积神经网络具有表征学习能力，能够按其阶层结构对输入信息进行平移不变分类，因此也被称为“平移不变人工神经网络。

这次大作业主要是将CNN引用与手写数字集的分类上，手写数字数据集中有一个数据就是pix，为15\*16的像素信息。利用CNN可以让模型学习到图片中某些空间结构，不同的数字它们之间的空间结构肯定是不一样的，而这样的空间结构就是由像素点与像素点之间的关系形成。数据由输入层经过卷积层之后，就使得模型能够考虑到像素点与像素点之间的关系。

## 5、BP神经网络

BP(back propagation)神经网络是1986年由Rumelhart和McClelland为首的科学家提出的概念，是一种按照误差逆向传播算法训练的多层前馈神经网络，是应用最广泛的神经网络。

## 6、线性回归

线性回归在所有的统计方法中绝对占有不可忽视的一席之地，其用途之广泛毋庸置疑，更重要的是它是整个回归家族中最为简单、也最容易理解的方法，几乎所有的统计学教材，线性回归绝对会有独立的章节，而其他的回归方法则很少有这种待遇。线性回归大致可分为单因素回归和多因素回归，这里的“单”和“多”是针对自变量的(也叫原因变量)。“线性”这一词实际上还说明了一个问题，即如果二者的关系不是直线的，用线性回归可能不一定适合，最好就换别的方法。

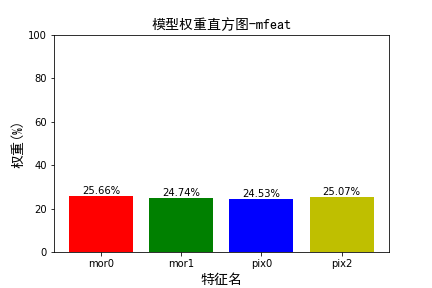
此次大作业中我们多个数据集进行线性回归，只发现pix存在一定的线性关系，其他数据集几乎没有线性关系。

# 三、训练过程

## 1、决策树、KNN和贝叶斯

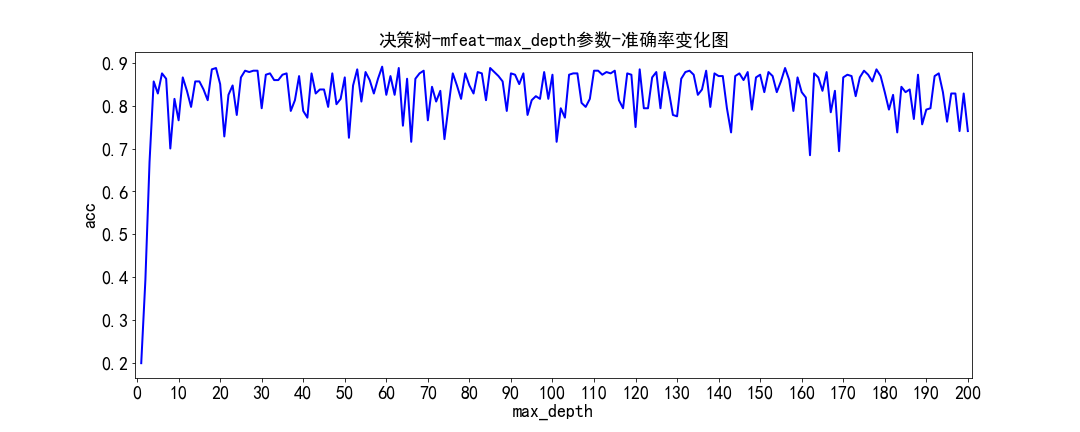
1. **mfeat数据集**

查看在模型中各特征的重要性：可以看到特征mor0，mor1，pix0和pix2对模型的影响相差不大，比较平均。

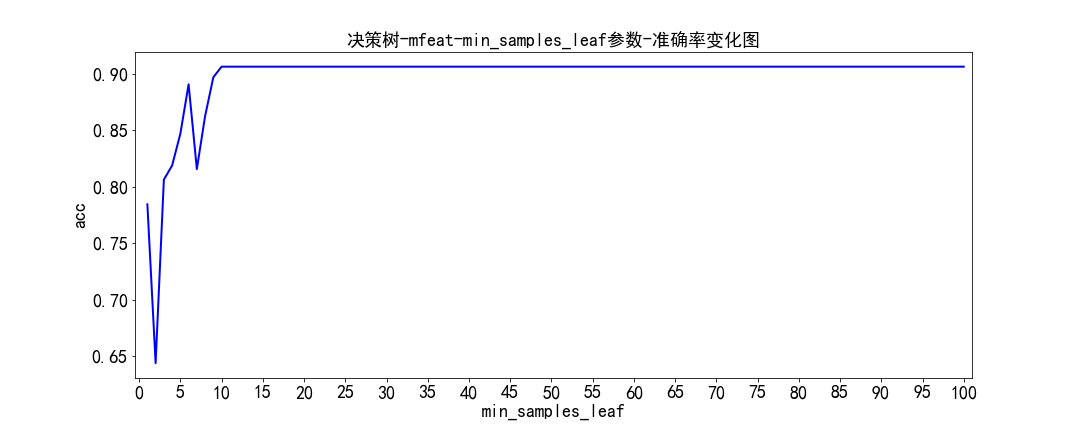


1. 决策树

选择决策树的树的深度为1-200，叶子节点的最少样本数为1时，在测试集上测试模型的性能。得到的结果如下所示，准确率随树的最大深度增加波动很大，可以看到在该范围内，当树的深度59时，预测效果最好，在测试集上预测的准确率为0.8906。

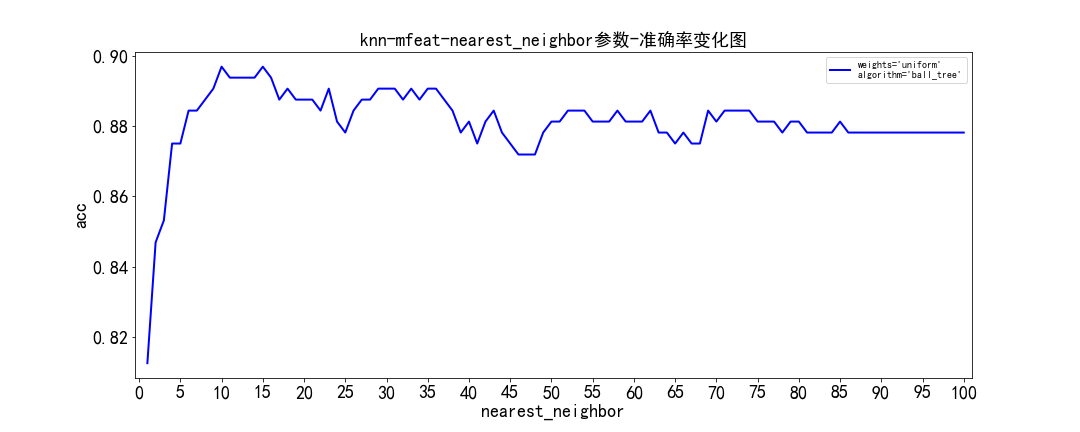


选择决策树的叶子节点的最少样本数为 1-100，树的最大深度为None，在测试集上测试模型的性能。得到的结果如下所示，准确率随叶子节点的最少样本数的增加而呈减小趋势。可以看到在该范围内，当叶子节点的最少样本数为10时，预测效果最好，在测试集上预测的准确率为0.9063。



1. KNN

设置 k 值为 1-100，模型效果如下图所示，模型性能随 k 值的增大而减小， 当 k=10时，预测的准确率最高，为0.8969，预测结果略低于决策树。

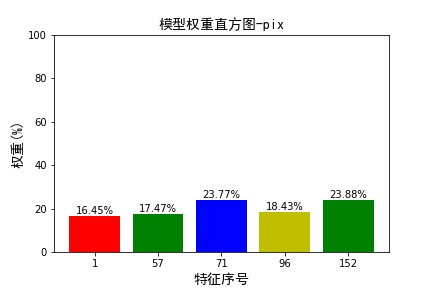


1. 贝叶斯（高斯贝叶斯）

模型预测的准确率为 0.925，性能比决策树和knn好，并且算法的运行时间比决策树短很多，该数据集满足独立分布的前提，所以贝叶斯模型适合用于该数据集的预测任务。

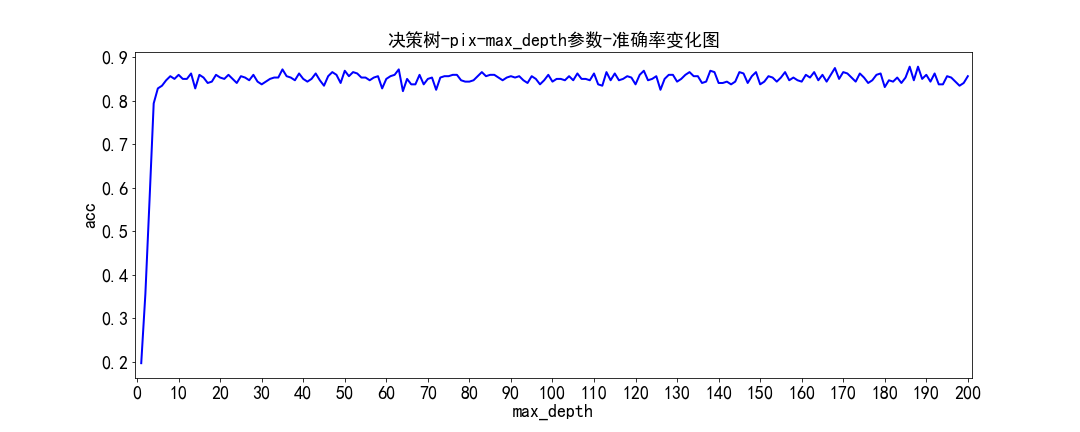
1. **pix数据集**

查看在模型中各特征的重要性：可以看到特征1，57，71，96，152对模型的影响较大，其中特征152对模型影响最大。

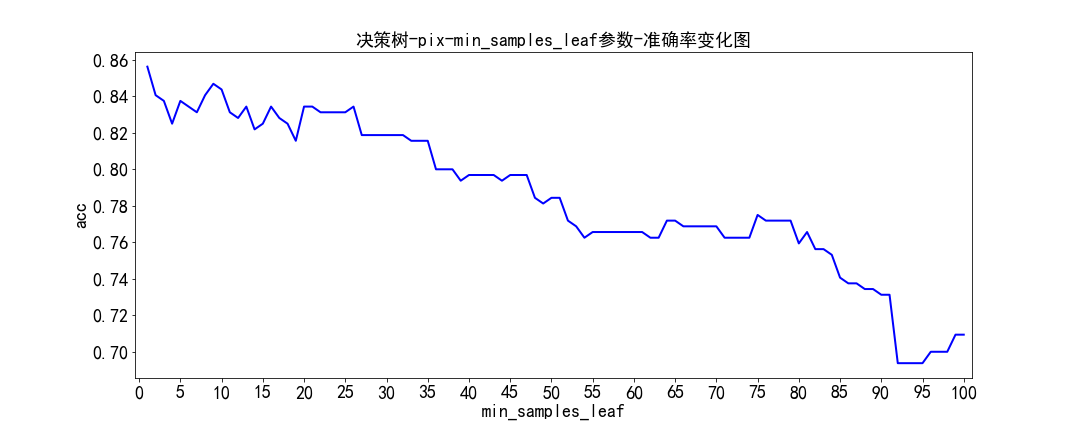


* 1. 决策树

选择决策树的树的深度为1-200，叶子节点的最少样本数为1时，在测试集上测试模型的性能。得到的结果如下所示，准确率随树的最大深度的增加上下波动，可以看到在该范围内，当树的深度为186时，预测效果最好，在测试集上预测的准确率为0.8781。

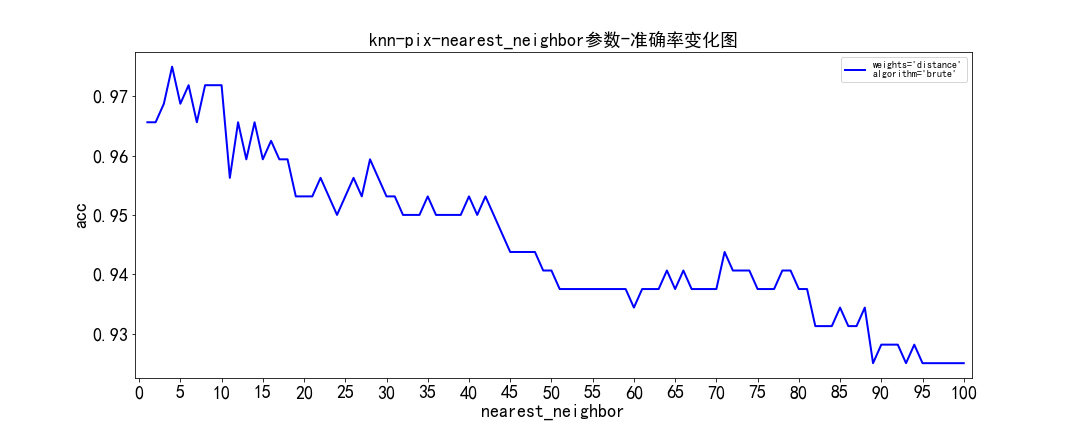


选择决策树的叶子节点的最少样本数为 1-100，树的最大深度为None，在测试集上测试模型的性能。得到的结果如下所示，准确率随叶子节点的最少样本数的增加而呈减小趋势。可以看到在该范围内，当叶子节点的最少样本数为1时，预测效果最好，在测试集上预测的准确率为0.8563。



* 1. KNN

设置 k 值为 1-100，模型效果如下图所示，模型性能随 k 值的增大而减小， 当 k=4时，预测的准确率最高，为0.975，预测结果高于决策树。

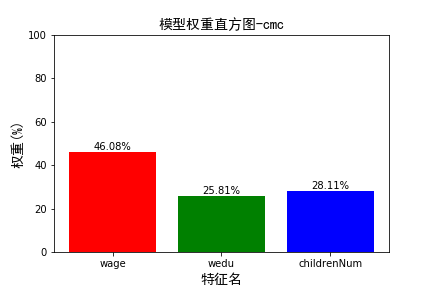


* 1. 贝叶斯（高斯贝叶斯分类器）

模型预测的准确率为 0.925，性能略高于决策树但低于KNN，并且算法的运行时间比决策树短很多，该数据集满足独立分布的前提，所以贝叶斯模型适合用于该数据集的预测任务。

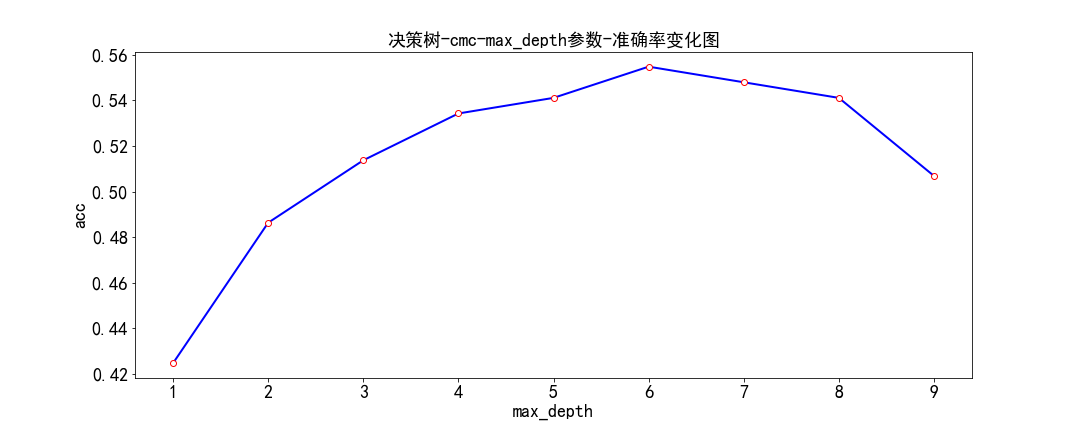
1. **cmc**

查看在模型中各特征的重要性：可以看到特征wage，wedu和childrenNum对模型的影响较大，其中影响最大的是wage。

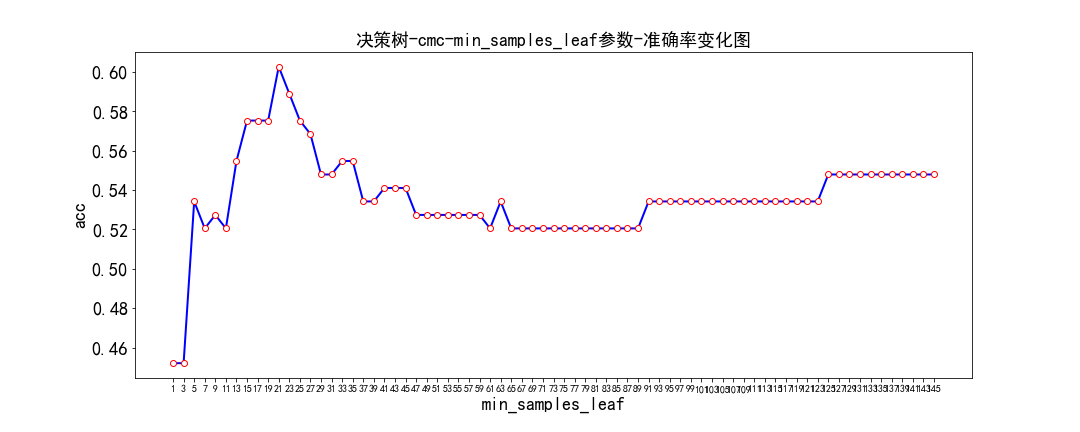


1）决策树

选择决策树的树的深度为1-9，叶子节点的最少样本数为1时，在测试集上测试模型的性能。得到的结果如下所示，可以看到在该范围内，当树的深度为6时，预测效果最好，在测试集上预测的准确率为0.5479。

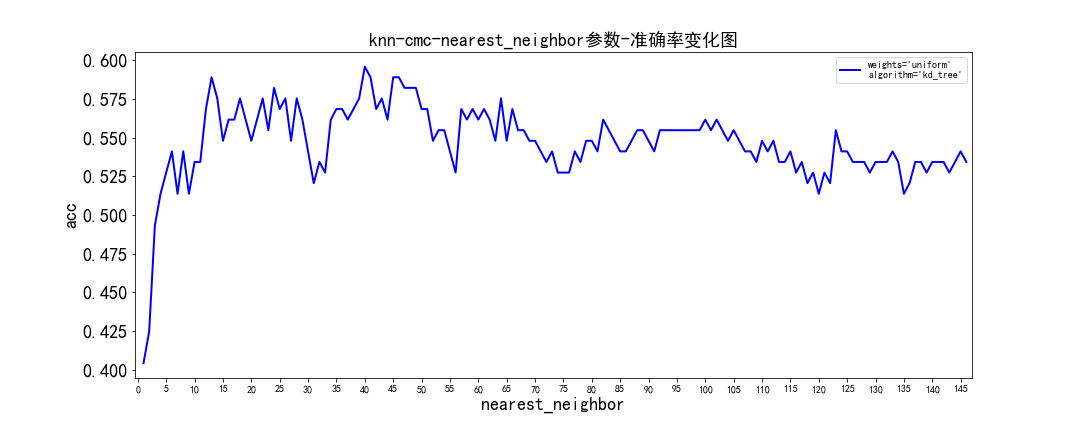


选择决策树的叶子节点的最少样本数为 1-145，树的最大深度为None，在测试集上测试模型的性能。得到的结果如下所示，准确率随叶子节点的最少样本数的增加整体呈减小趋势。可以看到在该范围内，当叶子节点的最少样本数为21时，预测效果最好，在测试集上预测的准确率为0.6027。



1. KNN

设置 k 值为 1-145，模型效果如下图所示，模型性能随 k 值的增大而上下波动， 当 k=40 时，预测的准确率最高，为0.5959，预测结果略低于决策树。

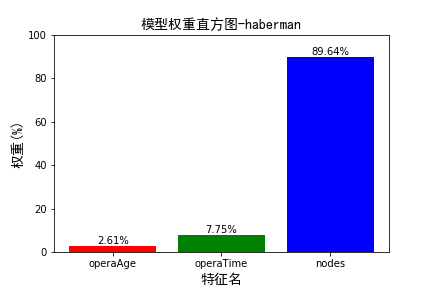


3）贝叶斯（多项式贝叶斯分类器）

模型预测的准确率为 0.4931，性能比决策树和KNN低很多，甚至低于了随机猜测概率0.5，所以贝叶斯模型不适合用于该数据集的预测任务。

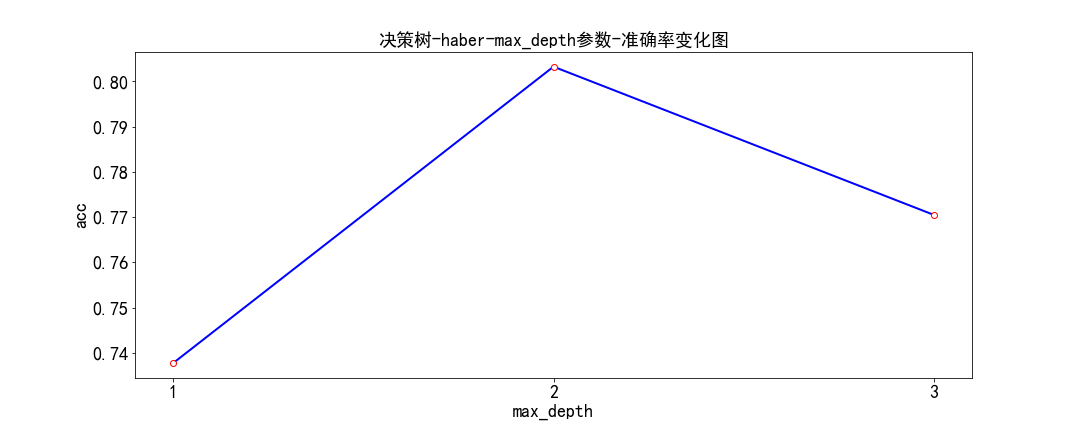
1. **haberman数据集（minmax）**

查看在模型中各特征的重要性：可以看到特征operaAge，operaTime和nodes对模型的影响较大，其中特征nodes对模型影响最大。

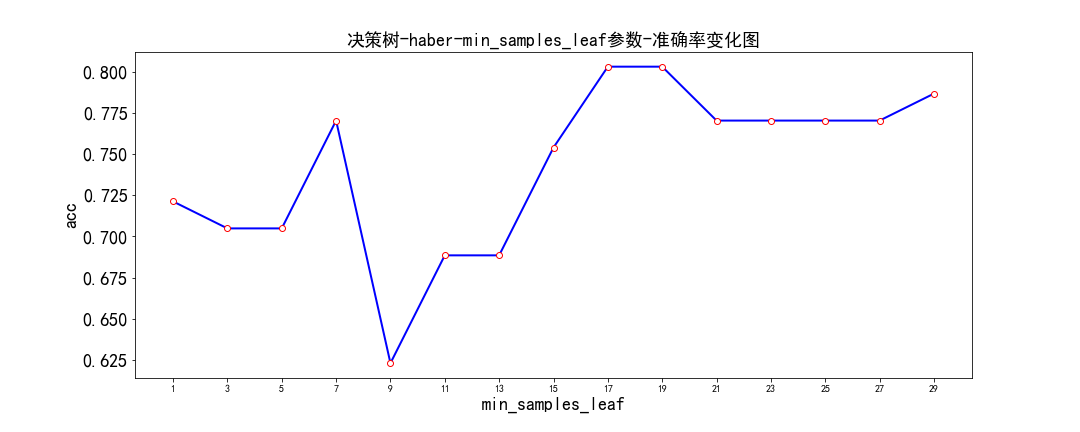


1. 决策树

选择决策树的树的深度为1-3，叶子节点的最少样本数为1时，在测试集上测试模型的性能。得到的结果如下所示，可以看到在该范围内，当树的深度为2时，预测效果最好，在测试集上预测的准确率为0.8033。

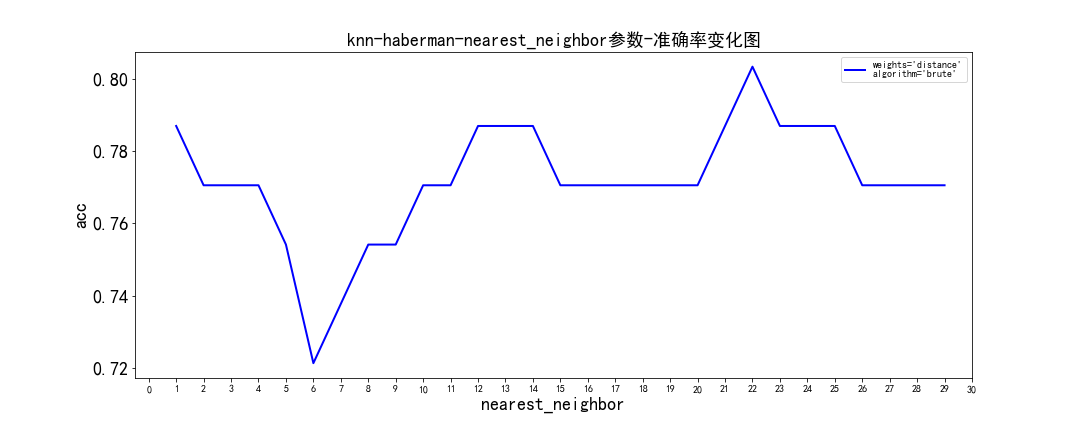


选择决策树的叶子节点的最少样本数为 1-29，树的最大深度为None，在测试集上测试模型的性能。得到的结果如下所示，准确率随叶子节点的最少样本数的增加整体呈上升趋势。可以看到在该范围内，当叶子节点的最少样本数为17和19时，预测效果最好，在测试集上预测的准确率为0.8033。



1. KNN

设置 k 值为 1-30，模型效果如下图所示，模型性能随 k 值的增大而上下波动，当 k=22 时，预测的准确率最高，为0.8033，预测结果等于决策树。



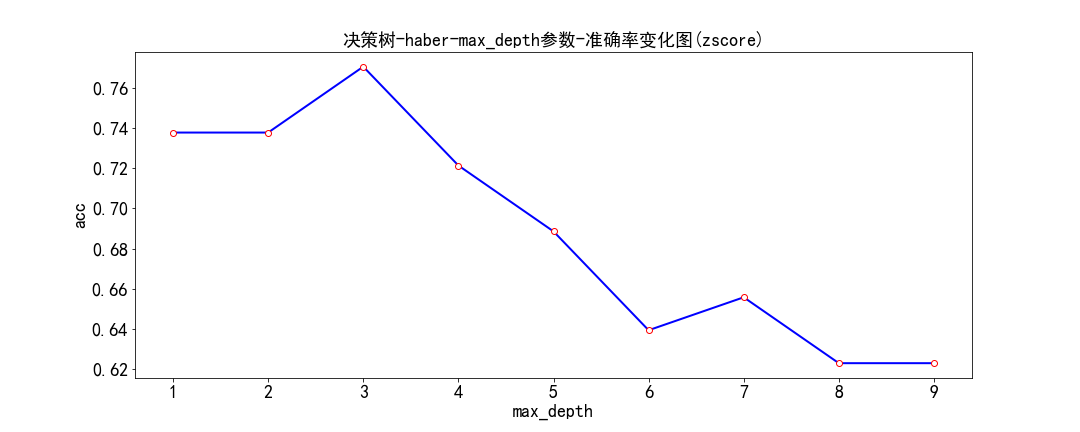
3）贝叶斯（高斯贝叶斯分类器）

模型预测的准确率为 0.7869，性能略低于决策树和KNN，但是算法的运行时间比决策树短很多，该数据集满足独立分布的前提，所以贝叶斯模型适合用于该数据集的预测任务。

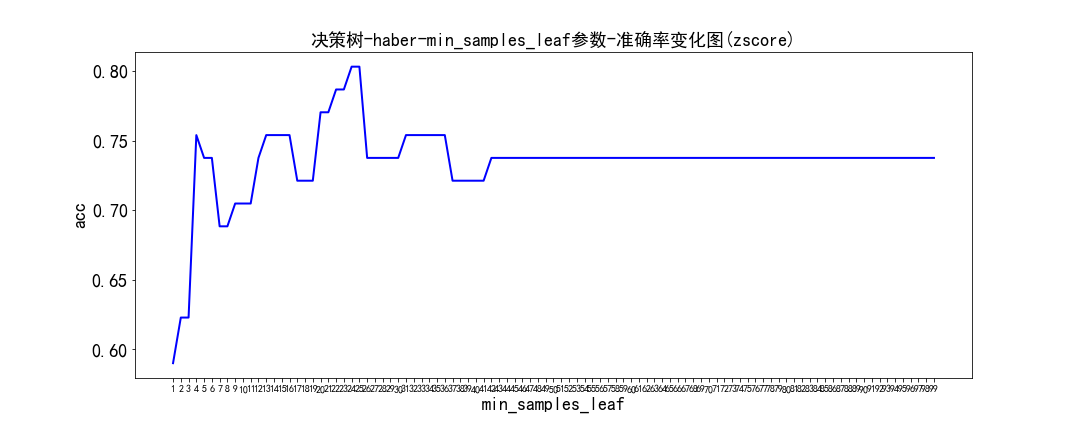
1. **haberman数据集（zscore）**

1）决策树

选择决策树的树的深度为1-9，叶子节点的最少样本数为1时，在测试集上测试模型的性能。得到的结果如下所示，可以看到在该范围内，当树的深度为3时，预测效果最好，在测试集上预测的准确率为0.7705。

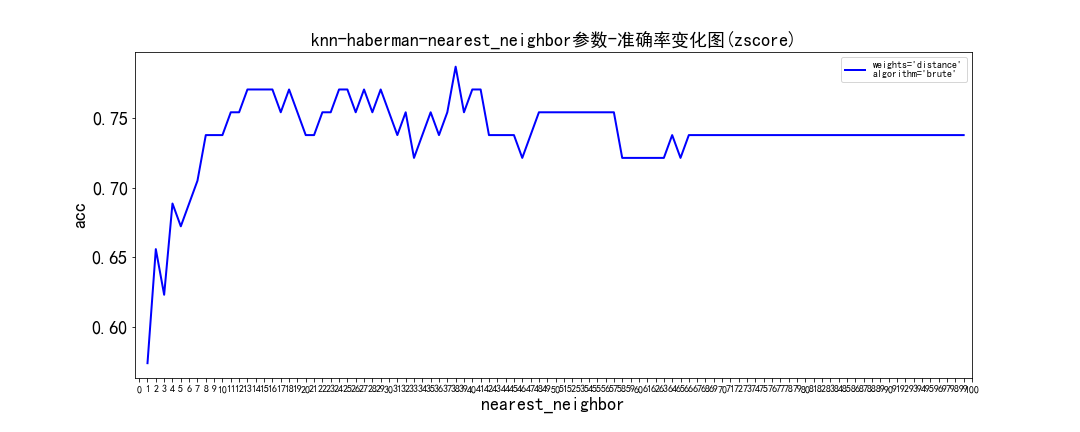


选择决策树的叶子节点的最少样本数为 1-100，树的最大深度为None，在测试集上测试模型的性能。得到的结果如下所示，准确率随叶子节点的最少样本数的增加整体呈上升趋势。可以看到在该范围内，当叶子节点的最少样本数为24和25时，预测效果最好，在测试集上预测的准确率为0.8033。



2）KNN

设置 k 值为 1-100，模型效果如下图所示，模型性能随 k 值的增大而上下波动，后趋于平稳，当 k=38 时，预测的准确率最高，为0.7869，预测结果低于minmax数据集的knn方法。

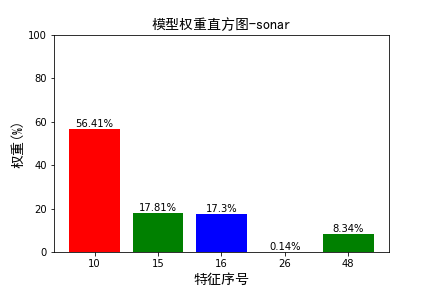


3）贝叶斯（高斯贝叶斯分类器）

模型预测的准确率为 0.7541，性能略低于zscore的决策树和KNN，也低于minmax的贝叶斯，zscore的总体准确率要低于minmax。

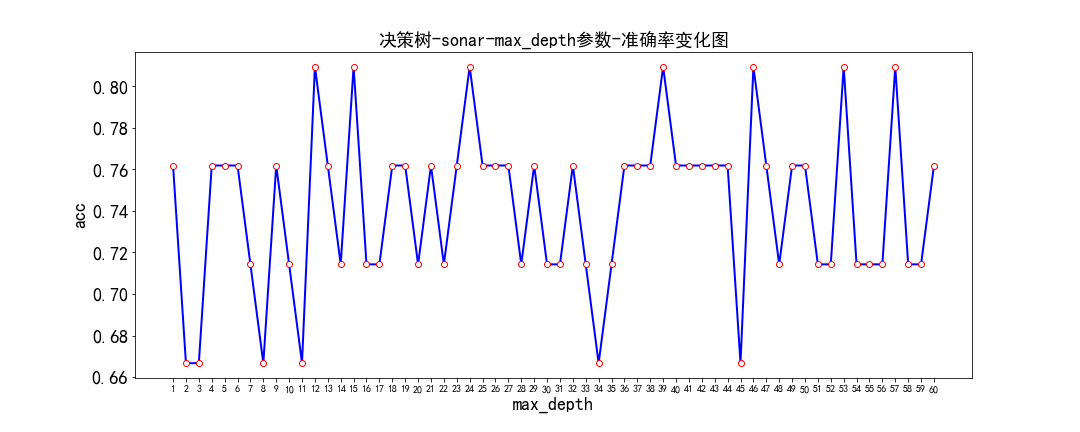
1. **sonar数据集**

查看在模型中各特征的重要性：可以看到特征10，15，,16，26和48对模型的影响较大，其中特征10对模型影响最大。

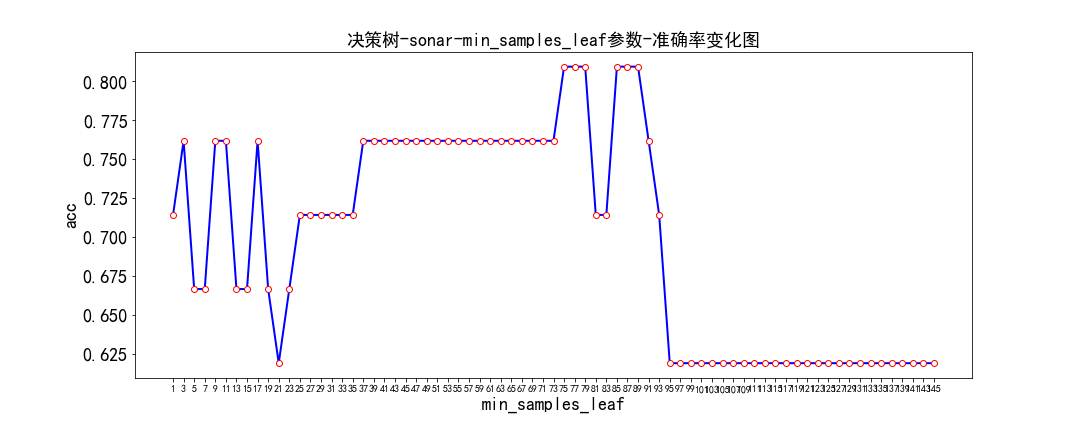


1. 决策树

选择决策树的树的深度为1-60，叶子节点的最少样本数为1时，在测试集上测试模型的性能。得到的结果如下所示，可以看到在该范围内，当树的深度为12，15，24，39，46，53，57时，预测效果最好，在测试集上预测的准确率为0.8095。

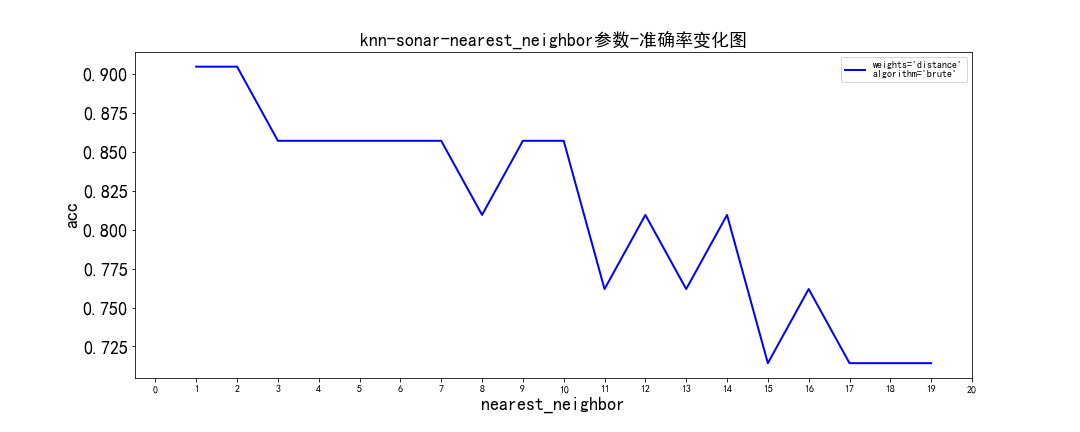


选择决策树的叶子节点的最少样本数为 1-29，树的最大深度为None，在测试集上测试模型的性能。得到的结果如下所示，准确率随叶子节点的最少样本数的增加整体呈上升趋势。可以看到在该范围内，当叶子节点的最少样本数为75，77，79，85，87，89时，预测效果最好，在测试集上预测的准确率为0.8095。



1. KNN

设置 k 值为 1-20，模型效果如下图所示，模型性能随 k 值的增大而呈整体下降趋势，当 k=1或2 时，预测的准确率最高，为0.9048，预测结果大于决策树。



3）贝叶斯（高斯贝叶斯分类器）

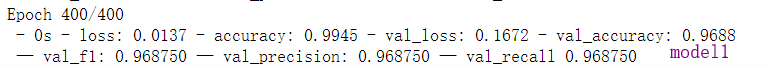
模型预测的准确率为 0.8095，性能等于决策树，低于KNN，但是算法的运行时间比决策树短很多，该数据集满足独立分布的前提，所以贝叶斯模型适合用于该数据集的预测任务。

## 2、CNN卷积神经网络

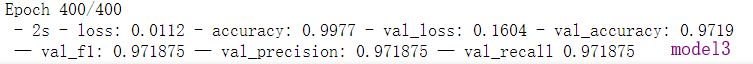
**Pix数据集：**

分别构建以下三个CNN神经网络进行训练。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **第一层卷积核** | **第一层池化** | **第二层卷积核** | **第二层池化** | **epochs** | **batch\_size** |
| model1 | 5\*5 | 2\*2 | 5\*5 | 2\*2\* | 400 | 300 |
| model2 | 4\*4 | 2\*2 | 4\*4 | 2\*2 | 400 | 300 |
| model3 | 3\*3 | 2\*2 | 3\*3 | 2\*2 | 400 | 300 |



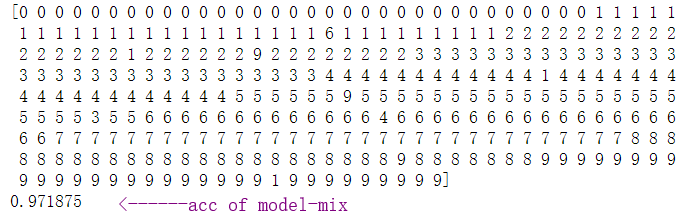




以上分别为三个模型跑出的最终结果，可以看出三者的效果都很好，最小的f1值都达到了0.965625，最高的甚至为0.971875。接下来为了使得模型更加准确，尝试进行将三个模型进行融合。由于卷积输出的结果是一个10\*1的矩阵，index为0的项代表改输入数据代表的手写数字为0的概率，范围为0.0-1.0，同理index为2的项代表改输入数据代表的手写数字为2的概率，index为i的项代表改输入数据代表的手写数字为i的概率。我们最终取的预测结果是这十项中概率值最大的，即为我们预测的数字。

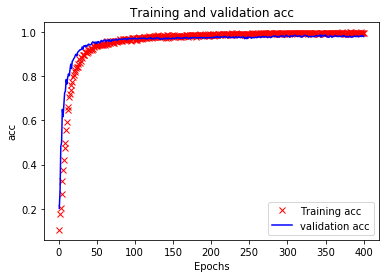
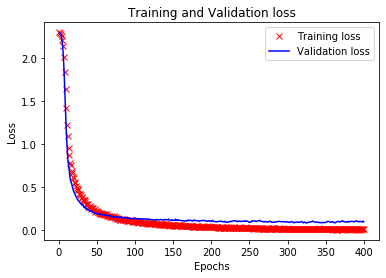
故从这个角度我们可以进行简单的模型融合，即将三个模型出的10\*1矩阵结果，进行加权平均操作。

上式中为model1模型输出10\*1结果index为0的项，为在求加权平均结果时，对model1乘以的权重系数。

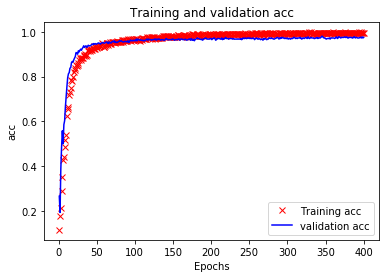
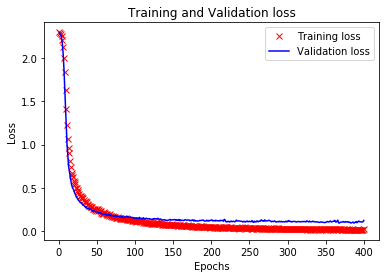


1. 训练结果

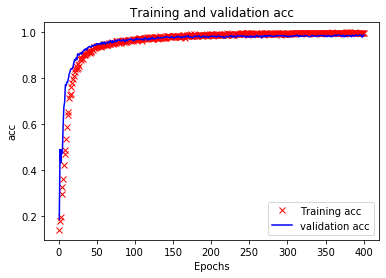
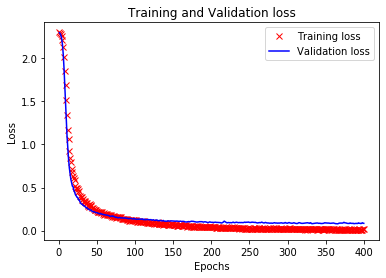
画出每个模型的训练集与验证集的loss随着Epochs变化的图像以及训练集与验证集准确率随着Epochs变化的图像。



*图：model1的相关图像*

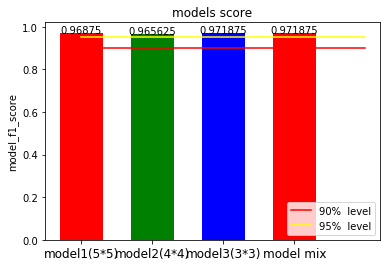


*图：model2的相关图像*



*图：model3的相关图像*

然后画出四个模型f1-score的条状图，从而得到直观的比较。

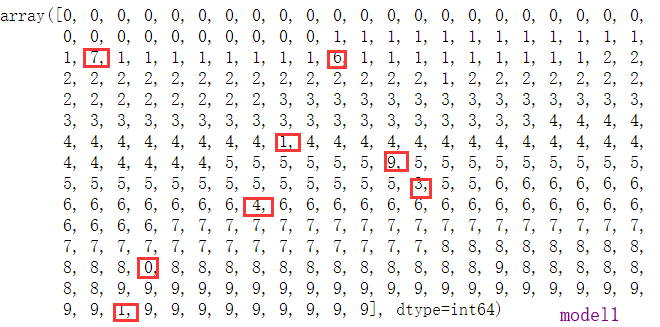


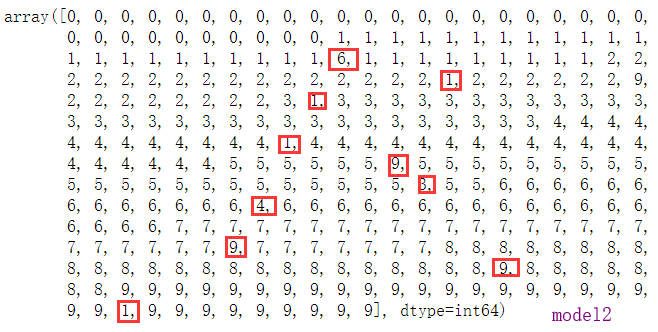
*图：四个模型f1-score条形图*

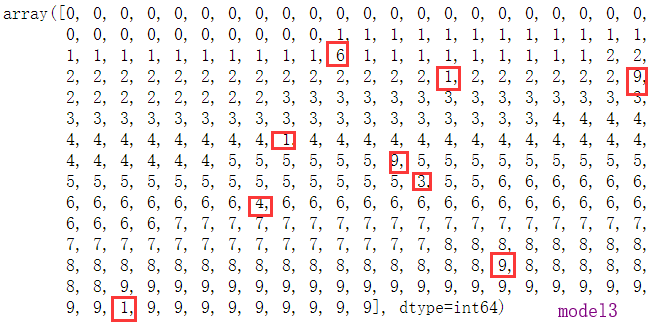
1. 结果分析

上面各个模型loss的变化图像可以看出，每个模型的大致趋势走向以及形状都相似，每个所示的都是一个比较“完美”的损失曲线变化图，在训练开始阶段损失值下降幅度很大，说明学习率合适且进行梯度下降过程，在学习到一定阶段后，损失曲线趋于平稳，损失变化没有一开始那么明显。曲线中的毛刺也没有特别地大，说明batch-size设置较为合理，因为batch-size设置越大，毛刺越小，毕竟每个batch-size的数据相当于不同的个体。

以下是三个独立的模型对测试集的预测情况。







最终的四个模型，模型融合后的模型f1-score值仍然为0.971875，竟然没有比3\*3卷积核来的高。这里就很值得思考，为什么会出现这样的情况。经过思考大概得到以下两个原因：

1. 加权平均值不能使得真实值的概率上升。

举个例子三个模型中前两个模型中把1认成了7，而最后一个模型是正确的。不妨设模型融合时权重为0.3、0.3、0.4

当三个模型的训练10\*1矩阵结果result前三大的概率是以下情况时：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1th | 2nd | 3rd |
| Model1 | result[7]=0.8 | result[1]=0.7 | result[2]=0.3 |
| Model2 | result[7]=0.8 | result[1]=0.7 | result[2]=0.3 |
| Model3 | result[1]=0.8 | result[2]=0.3 | result[7]=0.2 |

此时进行模型融合，融合之后的前三大的概率是以下结果：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1th | 2nd | 3rd |
| Model mix | result[1]=0.74 | result[7]=0.56 | result[2]=0.3 |

此时得到的结果中index为1的概率值最大，故模型会认为该数字为1，完成正确分类。

但是当三个模型的训练10\*1矩阵结果result前三大的概率是以下情况时：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1th | 2nd | 3rd |
| Model1 | result[7]=0.8 | result[2]=0.7 | result[1]=0.3 |
| Model2 | result[7]=0.8 | result[2]=0.7 | result[1]=0.3 |
| Model3 | result[1]=0.8 | result[2]=0.5 | result[7]=0.2 |

此时进行模型融合，融合之后的前三大的概率是以下结果：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1th | 2nd | 3rd |
| Model mix | result[2]=0.62 | result[7]=0.56 | result[1]=0.5 |

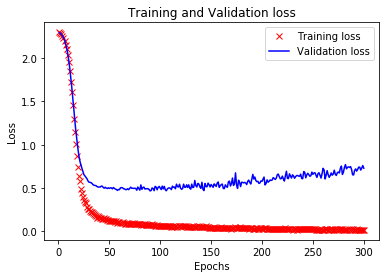
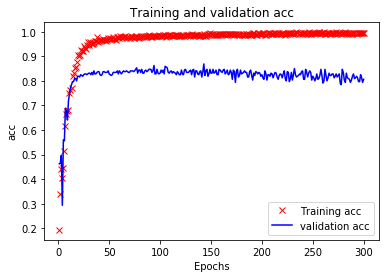
此时得到的结果中index为2的概率值最大，故模型会认为该数字为2，仍然是错误的分类。

从上面的例子可以看出。如果错误的识别结果10\*1概率矩阵中，概率值排在第二位的index若不是正确值，即使使用模型融合，也不能正确识别。很遗憾，通过打印各个模型预测结果中的10\*1概率矩阵发现，大多数的错误识别中，模型预测为正确值得概率很多都不是次大值。所以这就解释了为什么使用模型融合，并没有带来更好得结果，反而比原来最有效的模型还要表现的差。

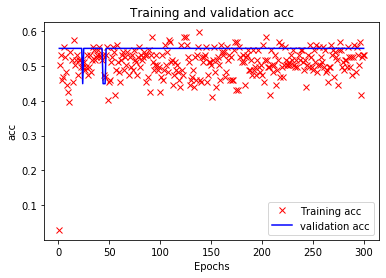
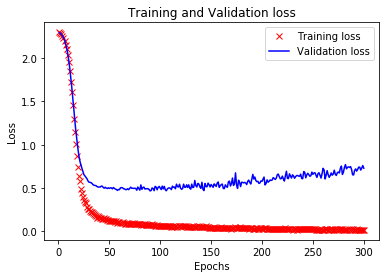
1. 准确率高的模型错误实例其实是准确率低模型错误实例的近似子集。这就意味着模型融合的结果很难超过最好的模型的效果。这一点观察上文中三个独立的模型对测试集的预测情况图可以看出。
2. 其他数据集情况：

结果：

|  |  |  |
| --- | --- | --- |
| 数据集 | 准确率 | F1-score |
| Mfeat(将三个数据集融合) | 0.8062 | 0.805687 |
| Sonar | 0.5507 | 0.550725 |

*图：mfeat(数据降维融合的新数据集)模型loss与acc变化图*



*图：sonar数据集模型loss与acc变化图*

分析：

Mfeat数据集训练的图像表面典型的过拟合。训练集loss趋向稳定,而验证集却没有趋同。可以通过drop out或者删减节点来进行解决。分析之所以过拟合的原因是原来的神经网络输入有240个节点，将三个数据集进行降维融合后，输入节点大大减小，也就不用再需要那麽多节点，自然会造成过拟合。所以一定程度也说明降维融合后的数据集不适合使用CNN进行训练识别，效果必然没有单独使用pix的数据集训练效果好，所以就无需进一步改进。

Sonar数据集只有loss的变化曲线正常，不存在欠拟合或者是过拟合。Acc变化曲线不太正常。整体在一个水平线上。训练集的acc陡升。最终的准确率却很低。说明数据集存在问题。CNN 一般都是对像素和音频进行学习，而Sonar这类数据可能不适合使用CNN进行训练。

## 3、BP神经网络

**Pix数据集：**

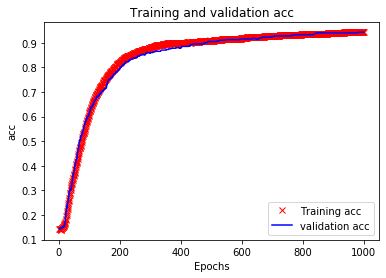
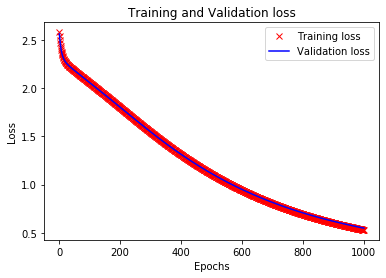
构建神经网络，总共四层，分别为输入一层，输出一层，隐藏两层，并设置各层的节点数目如下，然后进行训练。

|  |  |  |  |
| --- | --- | --- | --- |
| 输入层 | 隐藏层1 | 隐藏层2 | 输出层 |
| 240 | 256 | 128 | 10 |

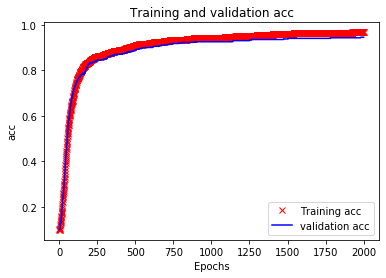
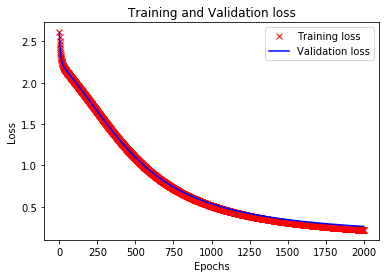
我们分别设置训练总批次为1000、2000以及3000，然后分别训练。

1. 训练结果

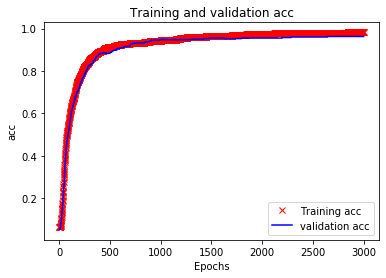
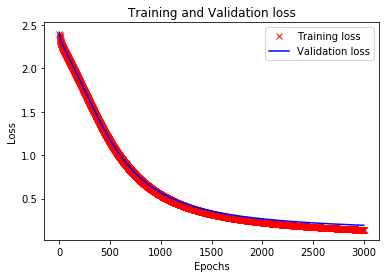
画出训练总批次不同的不同模型的训练集与验证集的loss随着Epochs变化的图像以及训练集与验证集准确率随着Epochs变化的图像。



*图：1000times模型的相关图像*



*图：2000times模型的相关图像*



*图：3000times模型的相关图像*

|  |  |  |
| --- | --- | --- |
| 训练总批次 | 准确率 | F1\_score |
| 1000 | 0.942 | 0.9425 |
| 2000 | 0.945 | 0.9450 |
| 3000 | 0.956 | 0.9563 |

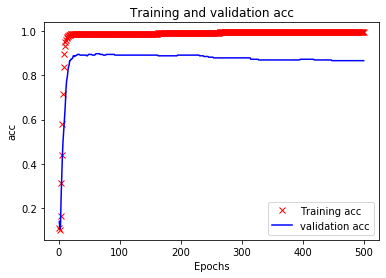
1. 结果分析

在第一次1000times的训练中结合loss下降图可以看出，损失值没有趋于平稳，很有可能是没有训练够；所以随即进行2000times以及3000times的训练，发现准确率和f1-score都有提升，而且损失值也逐步趋于平稳。我们有理由相信继续增加训练次数，应该可以进一步提升模型。知道最终图像和上文中CNN的损失值变化图像类似时，会得到更好的结果。

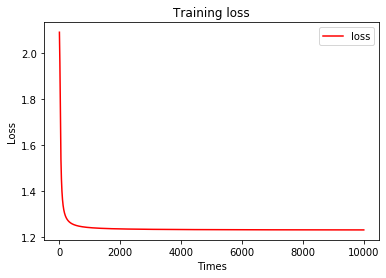
此外bp神经网络的训练，相比CNN要低效，需要更多时间与计算能力才能逼近CNN的效果。

1. 其他数据集情况：

|  |  |  |
| --- | --- | --- |
| 数据集 | 正确率 | F1-score |
| Mfeat(将三个数据集融合) | 0.866 | 0.865625000 |
| Sonar | 0.841 | 0.840600000 |
| Haber | 0.738 | 0.737704918 |
| cmc | 0.423 | 0.422997947 |



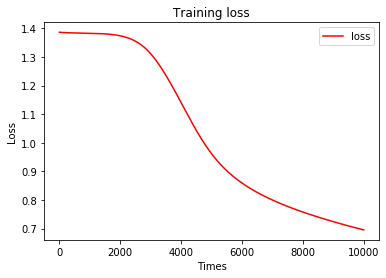
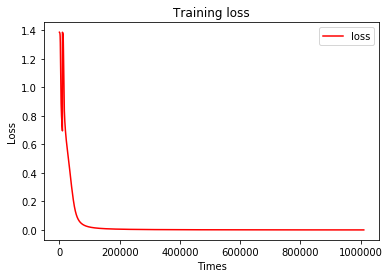
*图：Mfeat(将三个数据集融合)数据集模型的acc变化图像*



*图：cmc数据集模型的loss变化图像*

Cmc数据集使用BP训练时loss变化如图，此外 haber数据集也是类似的图像。所以这两个数据集在使用BP神经网络时并没用很好的效果。

以下为sonar数据集的训练情况：

****

|  |  |
| --- | --- |
| Times | acc |
| 10000 | 79.71% |
| 1000000 | 84.06% |

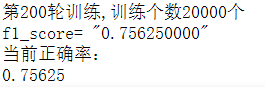
可以发现在训练次数为10000次时，只有79.71%的正确率，而成损失值没有最终收敛，故考虑增大训练次数到1000000次，从得到的loss变化图像来说，效果很明显，损失值达到了收敛，且准确率有显著提升，为84.06%。

## 4、线性回归

**Pix数据集：**

|  |  |
| --- | --- |
| 梯度下降率 | 正确率 |
| 0.001 | 0.1 |
| 0.00001 | 0.1 |
| 0.000001 | 0.3975 |
| 0.0000001 | 0.75 |
| 0.000000092 | 0.75625 |

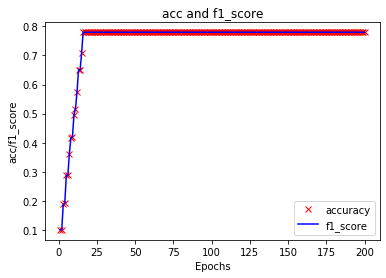
最终取梯度下降率为0.000000092



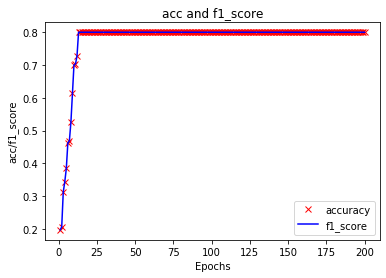
其他数据集也进行相似调参。

1.训练结果

|  |  |  |
| --- | --- | --- |
| 数据集 | 正确率 | F1-score |
| Pix | 0.75625 | 0.756250000 |
| Mfeat(将三个数据集融合) | 0.8 | 0.800000000 |
| Sonar | 0.5507246 | 0.550724638 |
| Haber | 0.73770493 | 0.737704918 |
| cmc | 0.42299795 | 0.422997947 |



*图：pix数据集f1-score与准确率变化图*



*图：mfeat数据集(降维融合后)f1-score与准确率变化图*

2.结果分析

可以看出pix数据集是存在一定的线性关系的，在开始做之前，在网上收集资料的时候就发现有人在用使用线性回归识别手写数字，虽然用的是MNIST数据集，但是这个数据集和本次大作业提供的UCI数据集中的pix还是有相似之处的，MNIST为28\*28像素点数据，而UCI中的pix中为15\*16的像素点数据。故可以推测pix出也可以找出一定的线性关系，事实上在未公布最终测试集，使用最初大的数字手写体数据进行线性回归时就可以达到0.8以上甚至0.9的准确率。值得注意的时将UCI数据集中数字手写体数据集中另外两个数据集和pix一起进行降维后得到新数据集，并以此进行训练发现线性的关系进一步增强。

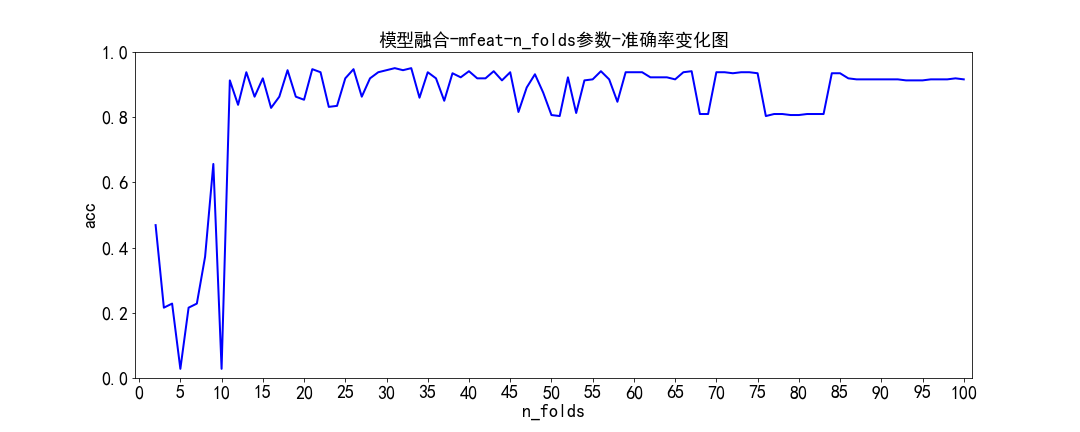
其他数据集大多都没有很强烈的线性关系，所以使用线性回归的效果并不是非常的好。因此也印证了前文所说的，“线性”这一词实际上还说明了一个问题，即如果二者的关系不是直线的，用线性回归可能不一定适合，最好就换别的方法。

## 5、模型融合

1、决策树、KNN和贝叶斯模型融合

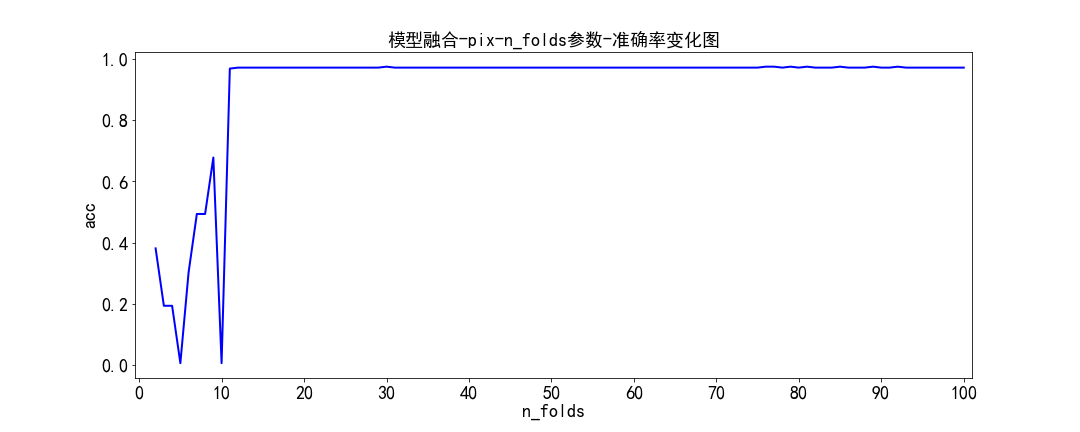
1）mfeat数据集

设置 n\_folds参数为 2-100，模型效果如下图所示，模型性能随 n\_folds 值的增大先增大后平稳， 当 n\_folds=31 时，预测的准确率最高，为0.95，预测结果大于knn。



2）pix数据集

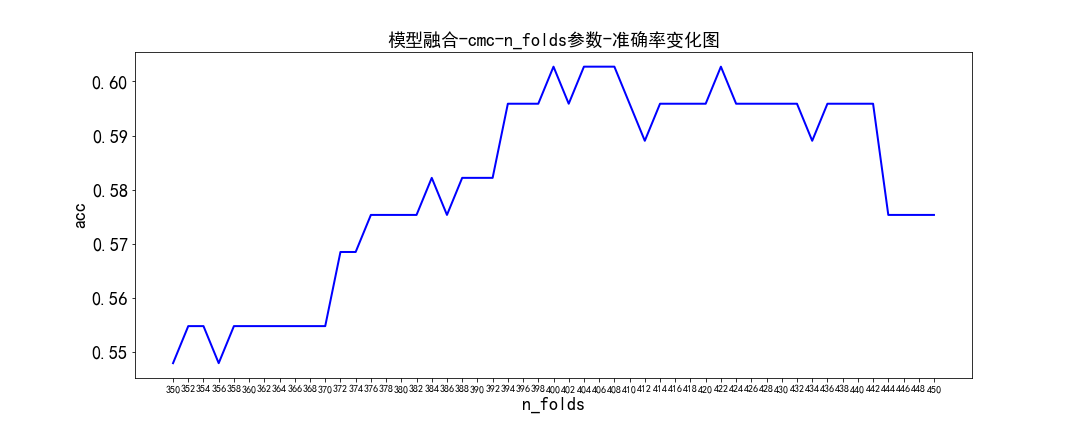
设置 n\_folds参数为 2-100，模型效果如下图所示，模型性能随 n\_folds 值的增大先增大后平稳， 当 n\_folds=30时，预测的准确率最高，为0.975，预测结果等于knn。



3）cmc数据集

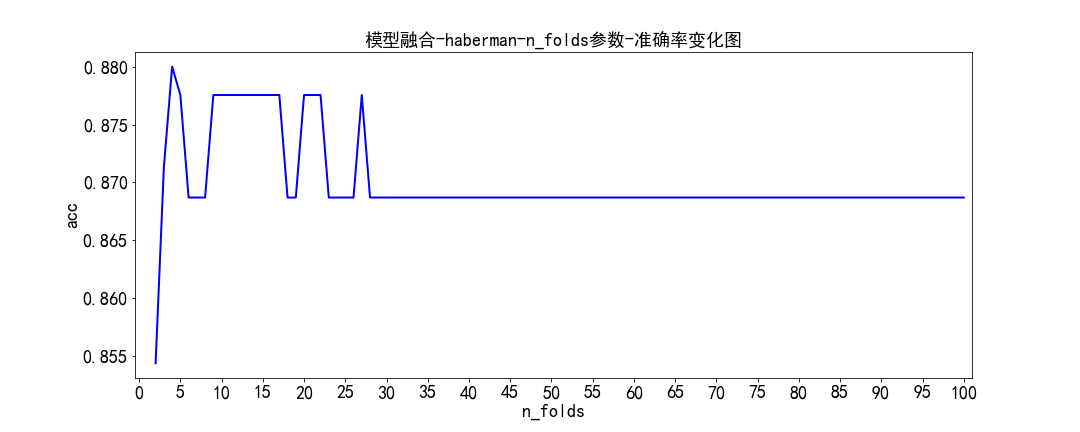
模型融合（knn、决策树和贝叶斯融合）

设置 n\_folds参数为 350-450，模型效果如下图所示，模型性能随 n\_folds 值的增大而上下波动， 当 n\_folds=400时，预测的准确率最高，为0.6027，预测结果等于决策树。



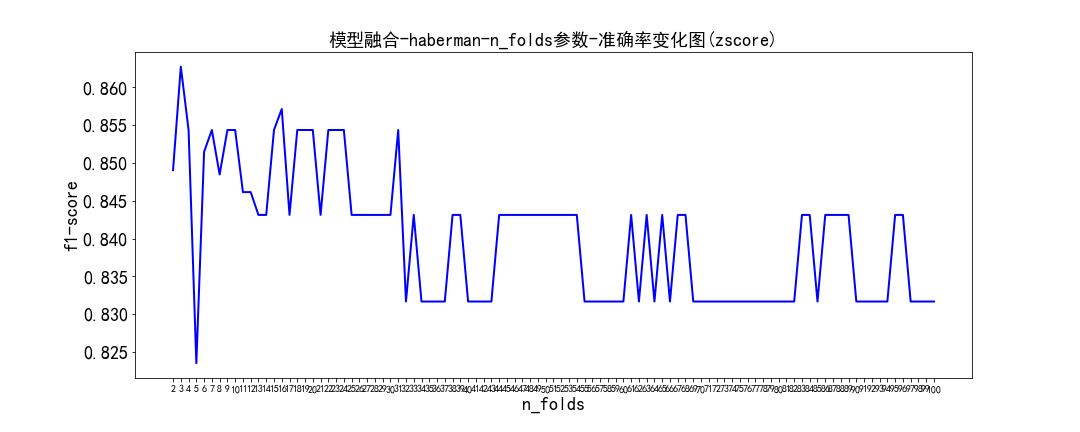
4）haberman数据集（minmax）

设置 n\_folds参数为 2-100，模型效果如下图所示，模型性能随 n\_folds 值的增大而上下波动，整体呈下降趋势 当 n\_folds=4时，预测的准确率最高，为0.88，预测结果大于knn、决策树和贝叶斯。



4）haberman数据集（zscore）

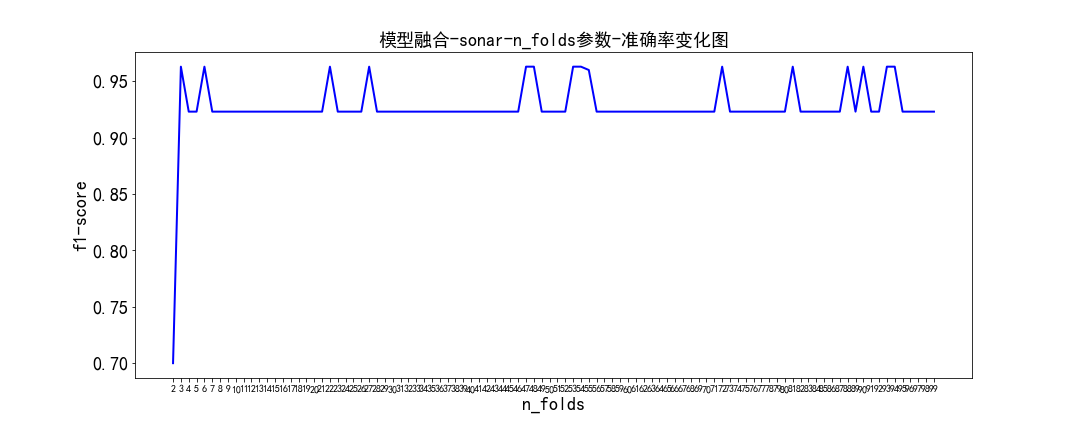
设置 n\_folds参数为 2-100，模型效果如下图所示，模型性能随 n\_folds 值的增大而上下波动，整体呈下降趋势 当 n\_folds=3时，预测的准确率最高，为0.8627，预测结果大于zscore的knn、决策树和贝叶斯，但低于minmax的模型融合。



5）sonar数据集

模型融合（knn、决策树和贝叶斯融合）

设置 n\_folds参数为 2-100，模型效果如下图所示，模型性能随 n\_folds 值的增大而上下波动，整体比较稳定 当 n\_folds=3时，预测的准确率最高，为0.9630，预测结果大于knn、决策树和贝叶斯。



2.逻辑回归、随机森林、SVM、GBDT、AdaBoost、XgBoost模型融合：

各个数据集表现如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Mfeat | Pix | Cmc | Haberman(minmax) | Haberman(zscore) | Sonar | Sonar |
| 逻辑回归 | 0.72 | 0.94 | 0.48 | 0.77 | 0.77 | 0.8 | 0.44 |
| SVM | 0.76 | 0.93 | 0.44 | 0.74 | 0.74 | 0.64 | 0.54 |
| 随机森林 | 0.95 | 0.91 | 0.52 | 0.77 | 0.72 | 0.86 | 0.44 |
| GBDT | 0.93 | 0.96 | 0.48 | 0.7 | 0.59 | 0.84 | 0.44 |
| AdaBoost | 0.97 | 0.96 | 0.52 | 0.75 | 0.64 | 0.9 | 0.44 |
| XgBoost | 0.88 | 0.93 | 0.5 | 0.72 | 0.7 | 0.8 | 0.44 |
| 模型融合（逻辑回归、SVM、GBDT、AdaBoost、XgBoost） | 0.984375 | 0.965625 | 0.52 | 0.75 | 0.75 | 0.9 | 0.54 |

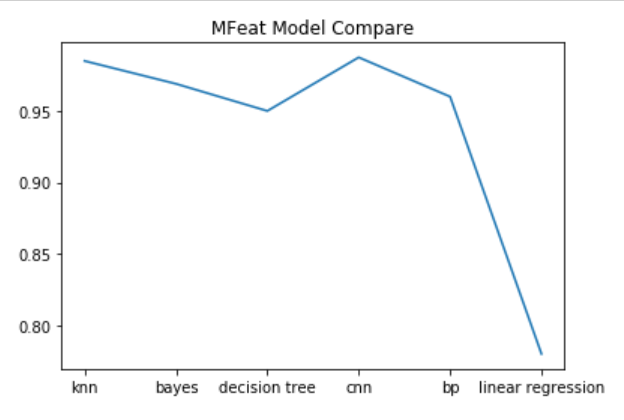
# 四、结果分析

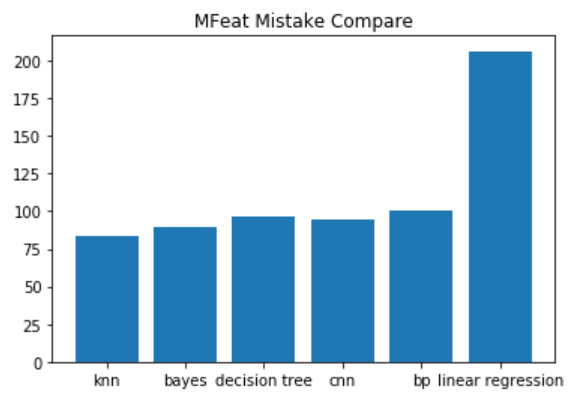
经过对上述数据的处理，现在我们逐一对数据进行分析。

## 1、mfeat数据集

在对mfeat数据集处理的过程中，由于后面加入测试集，集成预处理的结果较差，我们采用只训练pix数据的方法。

得到以下结果：



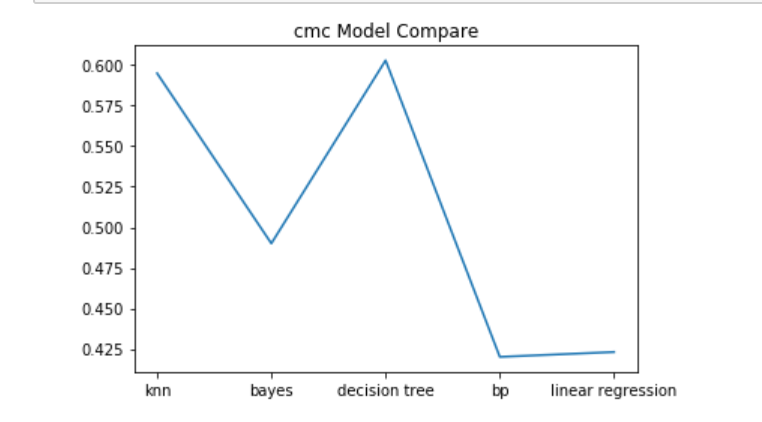


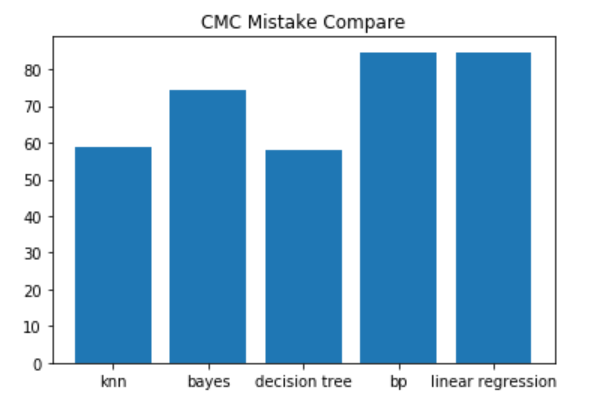
可以看出对于mfeat训练集，knn模型的准确率最高，线性模型的准确率较差，可以得出数据不具有线性特征，机器学习的模型准确率普遍高出深度学习。

而mfeat本身数据特征较分散，适合使用knn这样使用向量特征进行学习的模型。

## 2、cmc数据集

对cmc数据集模型训练得到以下结果：

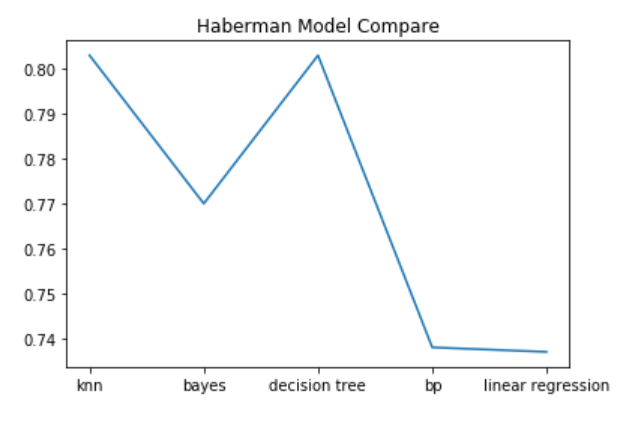


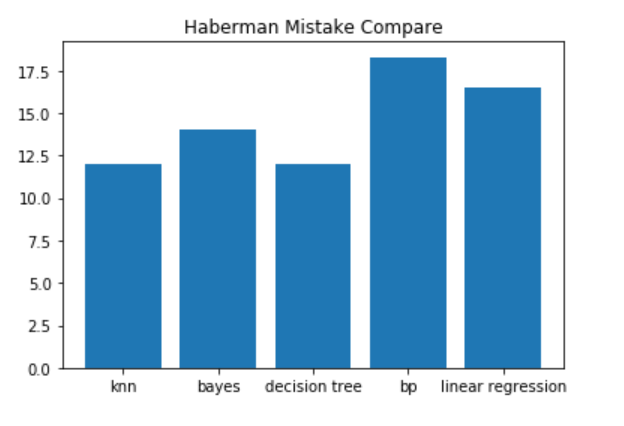


cmc本身并不具有分散的特征，训练结果较差，最高只能达到60%，在所有模型中knn和决策树表现较好，由于数据特征之间的关联度较低，所以贝叶斯预测值也偏低。bp和线性模型表现较差，数据没有线性特征。

## 3、Haberman数据集

在对各个模型的比较中，我们得到以下结果：

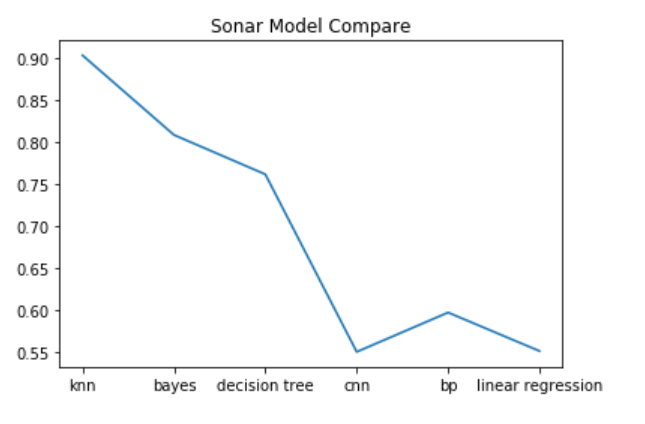


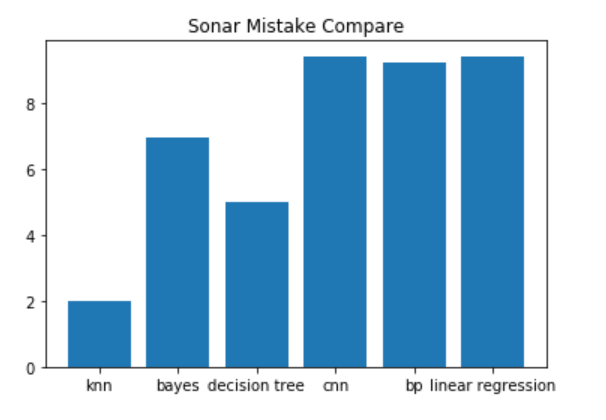


可以看出knn和决策树的结果较好，由于数据量较小，bp和线性回归表现较差。Haberman整体数据离散程度一般，最高准确率达到0.803.

## 4、Sonar数据集

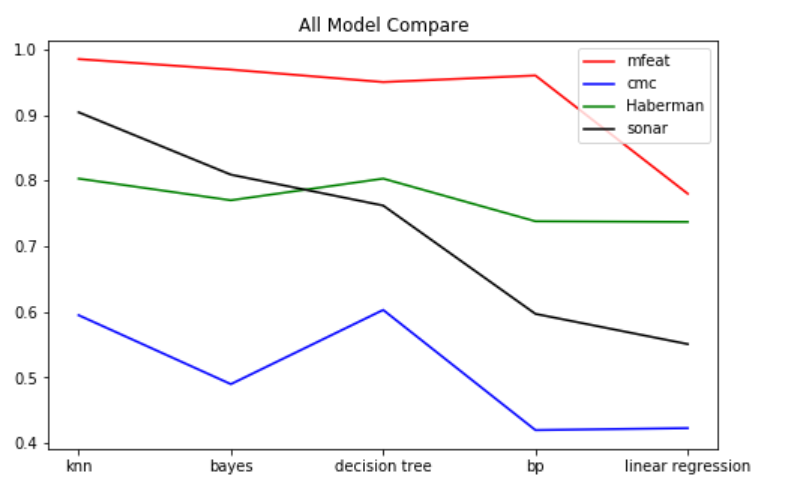
在对各个模型的比较中，我们得到了以下的结果：





可以看出knn结果极好，由于数据有比较明显的向量离散特征，噪声点较少，造成knn的准确率较其他模型有较大的提升，由于数据量较小，深度学习方法表现较差。数据没有明显的线性特征。

最后，我们将数据集放在一起进行比较：



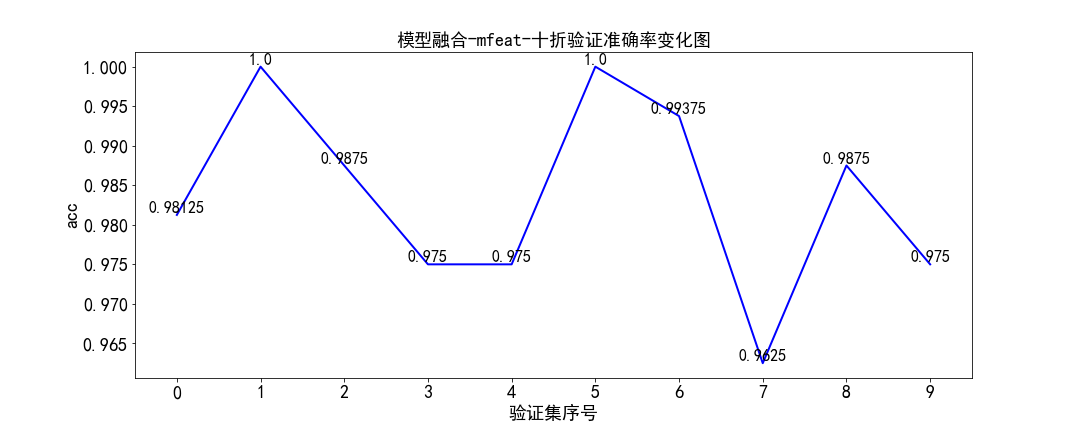
从这一张图可以看出mfeat相较于其他几个数据结果比较好，特征比较分散，可以很容易的使用分类器分类，Haberman总体几个模型较平稳，而sonar模型波动较大，因为不具有明显的线性规律，因此深度学习方法下降较大。Cmc数据特征较相近，训练结果较差。

算法和不同数据集f1-score的关系：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | mfeat | Pix | Cmc | Haberman(minmax) | Haberman(zscore) | Sonar | Sonar  (降维) |
| 决策树 | 0.9063 | 0.8925 | **0.603** | 0.8033 | 0.8033 | 0.8095 | 0.9283 |
| Knn | 0.8969 | **0.975** | 0.596 | 0.8033 | 0.7869 | 0.9048 | 0.9024 |
| 贝叶斯 | 0.925 | 0.925 | 0.493 | 0.7869 | 0.7541 | 0.8095 | 0.9024 |
| Cnn | 0.8057 | 0.97 | \ | \ | \ | 0.55 | \ |
| 线性回归 | 0.8 | 0.76 | 0.42 | 0.74 | 0.75 | 0.55 | 0.44 |
| Bp | 0.8656 | 0.96 | 0.423 | 0.7377 | 0.7377 | 0.8406 | \ |
| 模型融合（knn、决策树和贝叶斯） | 0.95 | **0.975** | **0.603** | **0.88** | **0.8627** | **0.9630** | **0.9524** |
| 逻辑回归 | 0.72 | 0.94 | 0.48 | 0.77 | 0.77 | 0.8 | 0.44 |
| SVM | 0.76 | 0.93 | 0.44 | 0.74 | 0.74 | 0.64 | 0.54 |
| 随机森林 | 0.95 | 0.91 | 0.52 | 0.77 | 0.72 | 0.86 | 0.44 |
| GBDT | 0.93 | 0.96 | 0.48 | 0.7 | 0.59 | 0.84 | 0.44 |
| AdaBoost | 0.97 | 0.96 | 0.52 | 0.75 | 0.64 | 0.9 | 0.44 |
| XgBoost | 0.88 | 0.93 | 0.5 | 0.72 | 0.7 | 0.8 | 0.44 |
| 模型融合（逻辑回归、SVM、GBDT、AdaBoost、XgBoost） | **0.98** | 0.97 | 0.52 | 0.75 | 0.75 | 0.9 | 0.54 |

## 5、模型融合十折交叉验证

从上表中可以看到，knn、决策树和贝叶斯的模型融合准确率最高，为了避免数据的偶然性，我们对准确率最高的模型融合进行了十折交叉验证，以下是mfeat数据集十折交叉验证的结果：



十折交叉验证的平均准确率为：0.98375