智能数据处理二分类问题实验报告

11749178 张涛

1、报告摘要：

面对非平衡的实验数据，通过数据的预处理，设计相应的调节数据平衡的算法，提高对于少类样本的召回率和准确率。本文主要采用 PCA 降维结合 smote 算法生成更多的少类样本数据，之后通过决策树、神经网络等多种分类算法模型对于数据进行分类，通过对于各算法模型精确度的对比分析，确定最优算法模型。

2、实验前期分析：

通过对于数据的人工观察可以可知，实验数据共有 10 个特征，最后 1 列为数据的标签（1/-1），且 label 为 1 的数量占比很少，约占总数据的 5.6%，可以认为这是一个 unbalanced 数据集，且进一步发现，第 3 到第 10 列特征数据的变化不明显，基本全为-1 或接近-1。故猜想部分维度对于结果影响很小，可以先进行降维处理。故先使用了 PCA 分析，确定各维度的方差和主成分占比。结果如下：

```
pca.explained_variance_ratio_ [  9.35570156e-01   1.77620293e-02   1.68798799e-02   9.61431975e-03
   9.49341885e-03   5.69470455e-03   2.03877122e-03   1.57229522e-03
   1.36806420e-03   6.36114094e-06]
pca.explained_variance_ [  2.02642466e-01   3.84721701e-03   3.65614537e-03   2.08244080e-03
   2.05625393e-03   1.23346065e-03   4.41593423e-04   3.40555734e-04
   2.96319738e-04   1.37780933e-06]
```

图 1 特征主成分分析

通过 PCA 分析可知，第一个维度占主成分比例 93.6%，其余特征的重要性几乎可以忽略不计，故为了数据可视化的便利和尽量保留有效信息，此处保留两个特征，以下是通过 PCA 降维之后并通过 smote 生成少类样本之后的数据可视化结果。

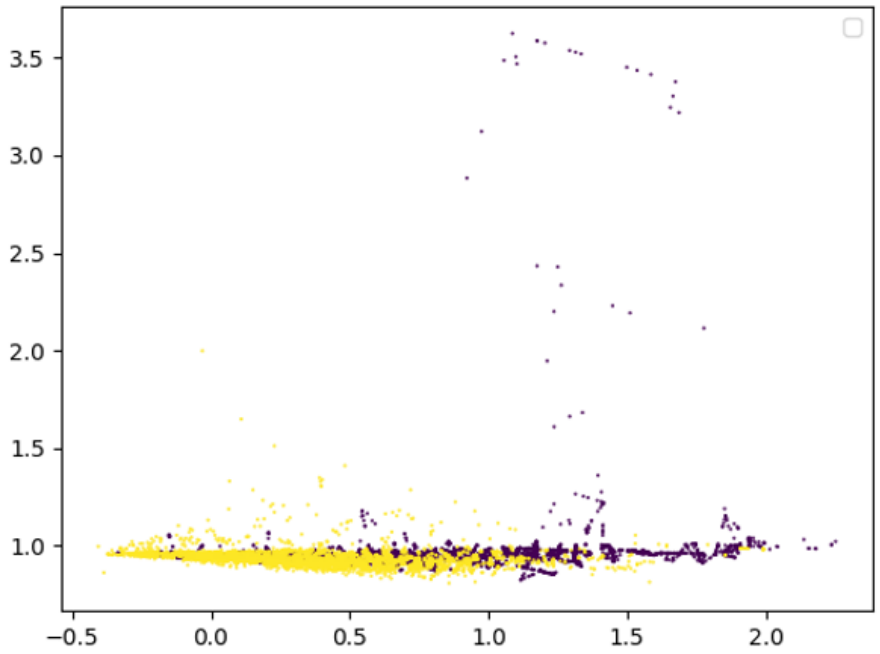

图 2 PCA+Smote 处理后的训练集

如果直接取前两个维度，不经过降维处理，直接进行了可视化分析，结果如下：

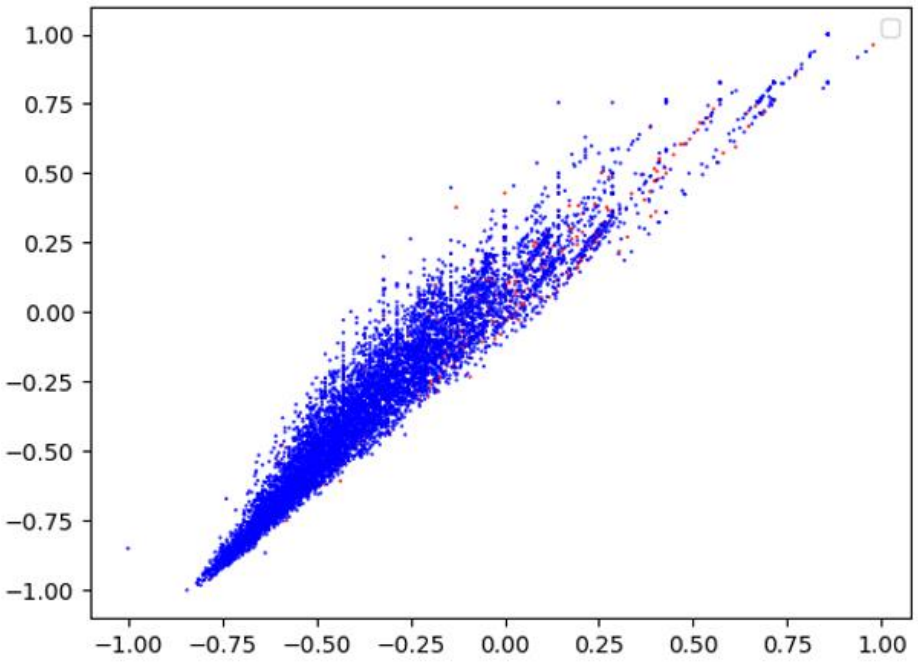

图 3 取前两个特征维度训练集数据可视化结果

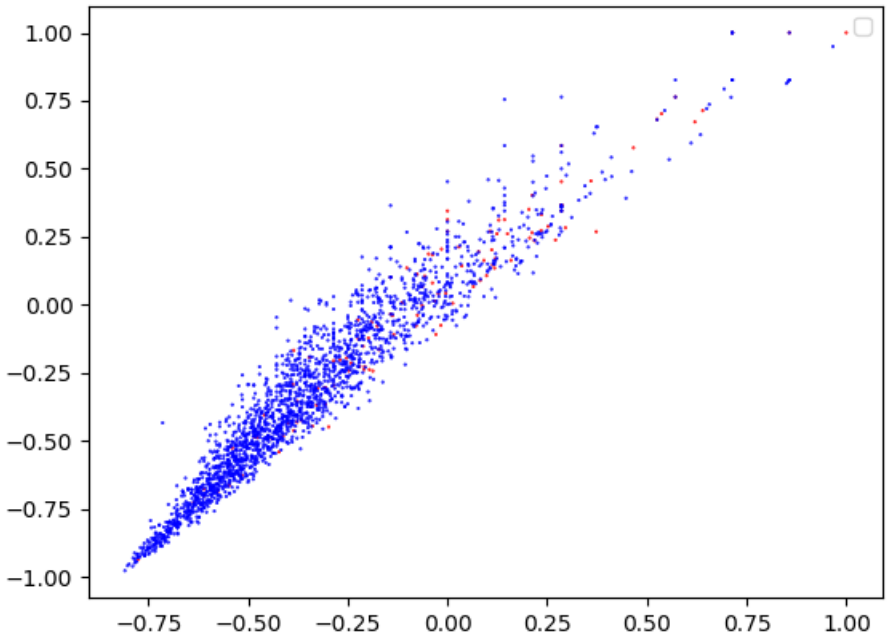

图 4 取前两个特征维度测试集数据可视化结果

通过可视化结果大致分析，数据应该是为**两个高斯分布且互有重叠的散点图**数据集。测试数据的可视化结果分布与训练集非常相似，只是点的散点的数量不同。

3、算法模型设计：

实验主要应用 BP 神经网络实现对于数据的分类，神经网络结构为 2-X-2 结构，X 为隐藏层神经元，学习率初始设置为 0.05，及其迭代次数等均为实验当中不断调节的参数之一。决策树算法为对比算法模型。

4、实验过程：

实验思路：

对于各种算法模型先尝试使用最简单的不降维进行分类，之后采用 PCA 与 smote 结合，进行降维和数据平衡，并使用多种分类模型，如此可以通过实验结果观察

模型改进的效果。

实验一：不降维直接进行 BP 神经网络分类（算法代码文件 BPNN.py）

首先进行了不降维直接通过 BPNN 算法进行二分类，结果如图。

	precision	recall	f1-score	support
-1	0.94	1.00	0.97	1955
1	0.00	0.00	0.00	117
avg / total	0.89	0.94	0.92	2072

图 5 BPNN 结果

实际运行结果很差，分析可能是因为数据的非平衡问题。

实验二：数据欠抽取 BP 神经网络分类 （算法代码文件 BPNN1.py）

之后采用了数据的欠抽取，使正负类样本数基本保持相同比例。网络结构为 2-5-2 BP 神经网络，为经过多轮调参后结果，学习率 0.005，得到训练结果为：

	precision	recall	f1-score	support
-1	0.94	0.71	0.81	817
1	0.25	0.67	0.36	117
avg / total	0.85	0.71	0.75	934

图 6 欠抽取 BPNN 结果

相比较于实验一有了明显改善。

实验三：PCA-Smote 处理后 BP 神经网络分类（算法代码 BPNN_PCA_smote.py）

步骤 1：采用了 PCA-Smote 对于数据预处理，训练集数据处理后可视化结果为：

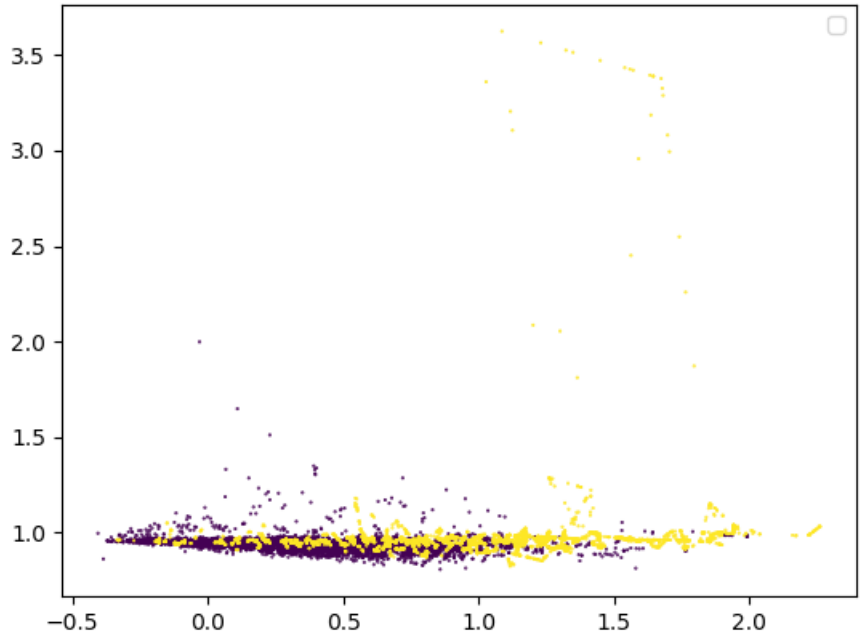

图 7 训练集 PCA-smote 处理后结果

测试集经过了 PCA 处理，得到可视化结果：

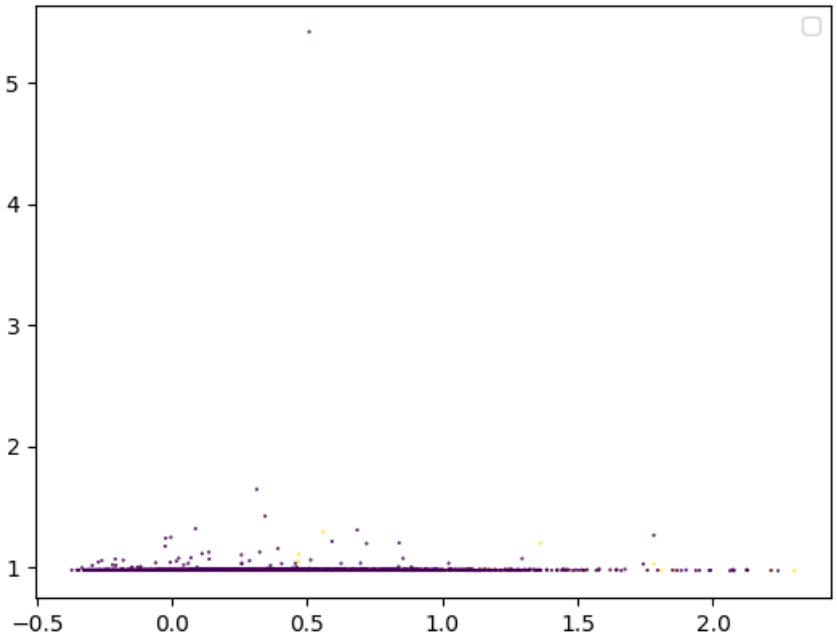

图 8 测试集 PCA-smote 处理后结果

可以明显发现，测试集也是一个非平衡的数据集，与训练集分布相似。

通过 PCA 降维和 smote 生成少类样本之后利用 BP 神经网络运行结果如图 9：

	precision	recall	f1-score	support
-1	0.98	0.64	0.77	1955
1	0.11	0.76	0.20	117
avg / total	0.93	0.65	0.74	2072

图 9 BPNN_PCA_smote 结果

运行结果正常，提高了少类样本的召回率。

实验四：不进行数据平衡处理的决策树算法（算法代码 Decision_tree.py）

利用决策树算法得到的结果，并且经过调节最大树深度等参数没有明显结果，分析决策树算法对于非平衡数据敏感。

	precision	recall	f1-score	support
-1	0.94	1.00	0.97	1955
1	0.50	0.01	0.02	117
avg / total	0.92	0.94	0.92	2072

图 10 决策树算法分类结果

决策树对于非平衡数据敏感，难以得到有效结果。

实验五：数据平衡处理的决策树算法（算法代码 Decision_tree_balanced.py）

采用了数据的平衡处理，并且调节树深度，最大深度为 1/2/3 时结果基本一致。

	precision	recall	f1-score	support
-1	0.98	0.72	0.83	1955
1	0.13	0.69	0.22	117
avg / total	0.93	0.72	0.79	2072

图 11 数据平衡处理决策树算法结果

能够有效将正类样本区分，基本达到实验要求。

实验六：利用以上五个实验当中三个有效实验算法模型共同预测（算法代码

total_predict.py)

	precision	recall	f1-score	support
-1	0.97	0.72	0.83	1955
1	0.13	0.69	0.22	117
avg / total	0.93	0.71	0.79	2072

图 12 多算法混合预测结果

从结果可以看出，多算法混合预测，最终结果和各算法结果近似。

5、实验结论:

采用了 precision recall f1-score 指标对于算法模型结果进行评价。

表 1 算法模型结果评价指数表

	1			-1			avg/total			有效性
	precision	recall	f1-score	precision	recall	f1-score	precision	recall	f1-score	
BPNN	0	0	0	0.94	1	0.97	0.89	0.94	0.92	无
BPNN1	0.13	0.66	0.21	0.97	0.73	0.83	0.93	0.73	0.8	有
BPNN_PCA_smote	0.11	0.76	0.2	0.98	0.64	0.77	0.93	0.65	0.74	有
Decision_tree	0.5	0.01	0.02	0.94	1	0.97	0.92	0.94	0.92	无
Decision_tree_balanced	0.13	0.69	0.22	0.98	0.72	0.83	0.93	0.72	0.79	有
total_predict	0.13	0.69	0.22	0.97	0.72	0.83	0.93	0.71	0.79	有

通过对于以上的实验结果分析，本次的实验主要在于对于处理数据的非平衡问题，正类样本相对很少，如果直接通过算法模型进行训练，得到的结果很差。所以需要处理数据的非平衡问题。通过实验结果也可以知道，数据预处理解决了数据非

平衡问题之后，神经网络、决策树的训练结果均有了明显的提升。同时为了进一步提高训练速度，采用了降维处理，去除无关特征，能够加快训练。最后将三种有效算法模型混合共同对于测试集数据进行预测，采用投票机制，使得预测结果更为准确稳定。

两种算法模型的比较上，都对于非平衡数据集很敏感。在训练的效率上 BP 神经网络相比较决策树算法需要花费更多的时间，在测试阶段，两种算法速度都很快。神经网络有更多的参数可供不断调节，能够寻找最优的结果，但这既是一个优点也是一个缺点，也意味着需要花更多的时间用于参数的调节选择上。

从结果看，最终的测试结果，神经网络和决策树结果基本一致，通过不同算法得到了一致的结果也印证了实验结果的有效性。

6、代码文件列表说明

文件	文件说明
BPNN_1.py	欠抽取 BP 神经网络算法代码
BPNN.py	不经过数据非平衡处理的 BP 神经网络代码
BPNN_PCA_smote.py	采用了 PCA 降维 smote 生成少类数据之后的 BP 神经网络算法
Decision_tree.py	不经过数据非平衡处理的决策树算法
Decision_tree_balanced.py	采用了数据平衡处理之后的决策树算法
Smote_pca.py	用于对于数据降维和生成少类样本的模块，以供其他算法调用。
Total_predict.py	结合三种有效算法模型的投票预测算法代码
\result	存放各算法模型运行的测试结果

BP 神经网络为自行编写的算法，对比算法决策树为调用 sklearn 包。