一、 算法原理

1.1 LR 分类算法

逻辑斯蒂回归（Logistic Regression）是一种用于分类问题的统计模型。它通过对输入变量的线性组合进行逻辑函数（Sigmoid 函数）变换，输出一个介于 0 和 1 之间的概率值。基本原理如下：

假设有输入变量向量$\mathbf{X}=(x_1,x_2,...,x_m)$，权重向量$\mathbf{W}=(w_1,w_2,...,w_m)$，偏置$b$，模型的输出为

$$z=\mathbf{W}^{\mathrm{T}}\cdot\mathbf{X}+b$$

接下来，使用 Sigmoid 函数将线性模型的输出转换为概率值：

$$\sigma(z)=\frac{1}{1+e^{-z}}$$

如果为二分类模型，如果概率值大于 0.5，则预测类别为 1，否则为 0。

1.2 BP 神经网络

BP 神经网络是一种多层前馈神经网络，通过反向传播（误差逆传播）算法进行训练。网络结构：由输入层、一个或多个隐藏层和输出层组成，每层由多个神经元构成。训练过程如下：

（1）初始化神经网络的权W和偏置θ。

（2）对实例中的每个训练样本：

①计算每个隐含层节点的输入：

$$I_j=\sum_i X_i W_{ij}-\theta_j$$

②计算每个隐含层节点的输出：

$$O_j=1/\left(1+e^{-I_j}\right)$$

③计算每个输出层节点的输入：

$$I_k=\sum_j O_j W_{jk}-\theta_k$$

④计算每个输出层节点的输出：

$$O_k=1/\left(1+e^{-I_k}\right)$$

⑤计算每个输出层节点的误差：

$$Err_k=O_k(1-O_k)\left(T_k-O_k\right)$$

⑥计算每个隐含层节点的误差：

$$Err_j=O_j(1-O_j)\sum_k W_{jk}Err_k$$

⑦更新输入层到隐含层的权值：

$$W_{ij} = W_{ij} + (l) X_i Err_j$$

⑧更新隐含层到输出层的权值：

$$W_{jk} = W_{jk} + (l) O_j Err_k$$

⑨更新每个隐含层节点的偏置：

$$\theta_j = \theta_j - (l) Err_j$$

⑩更新每个输出层节点的偏置：

$$\theta_k = \theta_k - (l) Err_k$$

（3）判断终止条件是否满足，若满足停止，否则重复执行步骤（2）。

1.3 朴素贝叶斯（Naïve Bayes）

朴素贝叶斯是一种基于贝叶斯定理的简单但高效的分类算法。基本原理如下：

根据贝叶斯定理，计算后验概率$P(c|\boldsymbol{x})$：

$$P(c|\boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})}$$

其中，$P(c)$是类“先验”概率；$P(\boldsymbol{x}|c)$是样本$\boldsymbol{x}$相对于类标记c的类条件概率。

基于属性条件独立性假设，后验概率$P(c|\boldsymbol{x})$可重写为

$$P(c|\boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})} = \frac{P(c)}{P(\boldsymbol{x})}\prod_{i=1}^{m} P(x_i|c)$$

其中m为属性数目，x_i为$\boldsymbol{x}$在第i个属性上的取值。

由于对所有类别来说$P(\boldsymbol{x})$相同，因此可以得到

$$c(x) = \arg\max_{c \in C} P(c)\prod_{i=1}^{m} P(x_i|c)$$

这就是朴素贝叶斯分类器的表达式。

1.4 ID3 决策树

ID3 决策树是一种基于信息增益的决策树算法，用于分类任务。基本原理如下：

在构建决策树时，选择能最大化信息增益的属性作为节点。信息增益基于熵的减少，熵是度量不确定性的一个指标。假定样本集合D中第k类样本所占的比例为$p_k(k=1,2,...,c)$，则D的信息熵定义为

$$\mathrm{Ent}(D) = -\sum_{k=1}^{c} p_k \log_2 p_k$$

假定离散属性a有V个可能的取值，记D_v是在属性a上取值为v的子集，则可计算出用属性a对样本集D进行划分所获得的“信息增益”

$$\mathrm{Gain}(D,a) = \mathrm{Ent}(D) - \sum_{v=1}^{V} \frac{|D_v|}{|D|} \mathrm{Ent}(D_v)$$

递归地选择属性分裂数据集，直到所有属性都使用完或子集中的样本全属于同一类别。

当不能再分裂时，当前节点作为叶子节点，叶子节点上的类标记为数据集中出现次数最多的类别。

1.5 KNN

KNN 是一种基于实例的学习算法，用于分类和回归任务。基本原理如下：

对于给定的测试样本，计算其与训练集中每个样本的距离，选择距离最近的k个邻居。距离度量常用欧氏距离（或者曼哈顿距离）

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

在分类任务中，对k个邻居的类别进行投票，选择票数最多的类别作为预测结果。在回归任务中，取k个邻居的均值作为预测值。

k的取值对算法性能有重要影响，通常通过交叉验证选择合适的k值。

二、 对比试验

2.1 LR 与 BP 算法比较检验

使用 Weka 平台的 Experiment 模块进行实验。分类数据集为 iris.arrf、car.arrf、flags.arrf。实验结果如下图 1。

```
Tester:     weka.experiment.PairedCorrectedTTester -G
Analysing:  Percent_correct
Datasets:   3
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       24-7-1 下午4:08


Dataset                   (1) function | (2) ly.BP
------------------------------------------------
iris                     (100)   97.13 |   84.67 *
car                      (100)   93.28 |   88.87 *
flags                    (100)   45.33 |   33.83 *
------------------------------------------------
                                 (v/ /*) |   (0/0/3)
```

图 1　LR 与 BP 算法对比

从上述结果可以看出，在 iris 和 car 数据集上，逻辑斯蒂回归的表现优于 BP 神经网络，分别达到了更高的准确率。在 flags 数据集上，两者的表现都不理想，但逻辑斯蒂回归仍然略优于 BP 神经网络。

在置信水平 0.05 下的双边 t 检验结果为（0/0/3），表示有 3 个数据集中没有观察到显著性差异（v），并且没有数据集中观察到显著性差异（*）。

逻辑斯蒂回归优缺点：

优点：

- 简单而高效，实现容易，适合作为基准模型。
- 输出结果可以解释为概率，对于需要知道每个类别的概率的任务有用。
- 不容易过拟合，在数据量不大的情况下表现稳定。

缺点：

- 假设数据线性可分，对于非线性数据分类效果不佳。
- 对特征之间的相关性较为敏感，需要事先进行特征工程处理。

BP 神经网络优缺点：

优点：

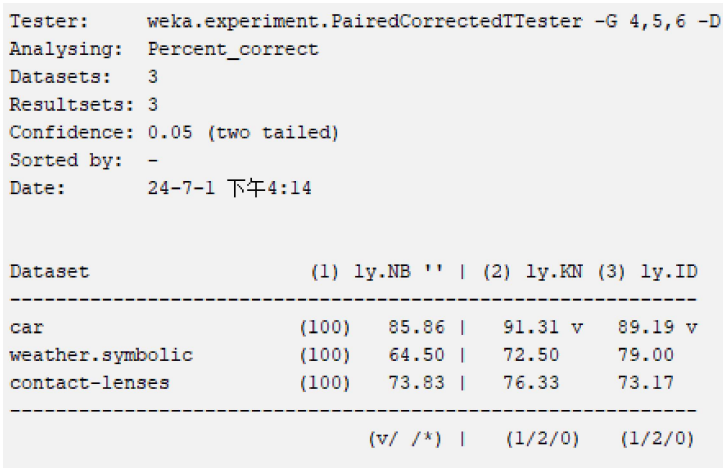
- 能够学习复杂的非线性关系，适合处理复杂的分类问题。
- 可以通过调整网络结构和超参数来优化性能，灵活性较高。

缺点：

- 容易过拟合，特别是在数据量较少的情况下。
- 训练时间较长，计算资源消耗较大。
- 对初始权重的选择比较敏感，可能会影响训练结果的稳定性。

2.2 NB、KNN、ID3 分类算法比较检验

使用 Weka 平台的 Experiment 模块进行实验。分类数据集为 car.arrf、weather.symbolic.arrf、contact-lenses.arrf。实验结果如下图 2。



```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D
Analysing:  Percent_correct
Datasets:   3
Resultsets: 3
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       24-7-1 下午4:14


Dataset                   (1) ly.NB '' | (2) ly.KN (3) ly.ID
------------------------------------------------------------
car                     (100)   85.86 |   91.31 v   89.19 v
weather.symbolic        (100)   64.50 |   72.50     79.00
contact-lenses          (100)   73.83 |   76.33     73.17
------------------------------------------------------------
                              (v/ /*) |   (1/2/0)   (1/2/0)
```

图 2　NB、KNN、ID3 分类算法对比

从上述结果可以看出，在 car 数据集上，KNN 的分类效果最好，其次是 ID3 决策树，NB 表现稍逊。在 weather.symbolic 数据集上，ID3 决策树的表现最好，NB 最差。在 contact-lenses 数据集上，KNN 和 ID3 决策树表现相对接近，NB 略逊一筹。

在置信水平 0.05 下的双边 t 检验结果为（1/2/0），表示对于 NB 和 KNN 之间的比较，有 1 个数据集中观察到显著性差异（v），对于 KNN 和 ID3 之间的比较也有 1 个数据集中观察到显著性差异（v），而 NB 和 ID3 之间的比较没有观察到显著性差异（*）。

NB 优缺点：

优点：

- 算法简单，容易实现。
- 对小规模数据表现良好，处理速度快。
- 在数据较为符合假设的情况下，分类效果不错。

缺点：

- 假设特征之间条件独立，这在实际数据中往往不成立，可能导致分类精度下降。
- 对输入数据的分布偏差较为敏感。

KNN 优缺点：

优点：

- 简单直观，易于理解和实现。
- 适用于复杂的决策边界和非线性数据。
- 不需要训练过程，新数据可以直接加入模型。

缺点：

- 对于高维数据和大数据集计算开销大，预测速度慢。
- 需要选择合适的 k 值，对参数敏感，选择不当可能影响分类结果。
- 对数据中噪声和冗余特征敏感，可能导致过拟合。

ID3 决策树优缺点：

优点：

- 可以处理多类别问题。
- 输出结果易于理解，可以生成清晰的规则。

缺点：

- 对噪声数据和缺失值敏感，可能导致过拟合。
- 容易在训练集上过度匹配，泛化能力较弱。
- 对数据的顺序敏感，不稳定，小的变化可能导致完全不同的树结构。