# 一、 提出问题

掌握线性回归、朴素贝叶斯和K均值聚类算法的基本原理和具体实现步骤，利用 Weka 平台，测试 3 种算法，整理实验结果，完成比较检验，分析比较算法优缺点。

# 二、 算法原理

## 2.1 线性回归算法

线性回归是一种用于预测连续数值输出的统计模型，其基本原理如下：

假设有样本$\boldsymbol{x}=(x_1,x_2,...,x_m)$，其中$x_i$是$\boldsymbol{x}$在第$i$个属性上的取值，权重向量$\boldsymbol{w}=(w_1,w_2,...,w_m)$，偏置$b$，则模型的输出为

$$y=\boldsymbol{w}^{\mathrm{T}}\cdot\boldsymbol{x}+b$$

通常使用最小二乘法来估计$\boldsymbol{w}$和$b$

$$w=\frac{\sum_{i=1}^{n}y_i\left(x_i-\overline{x}\right)}{\sum_{i=1}^{n}x_i^2-\frac{1}{n}\left(\sum_{i=1}^{n}x_i\right)^2}$$

$$b=\frac{1}{n}\sum_{i=1}^{n}(y_i-wx_i)$$

或者求解以下正规方程

$$(w;b)=(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\boldsymbol{y}$$

其中，

$$\mathbf{X}=\begin{pmatrix}x_1^{\mathrm{T}} & 1\\ x_2^{\mathrm{T}} & 1\\ \vdots & \vdots\\ x_m^{\mathrm{T}} & 1\end{pmatrix}$$

## 2.2 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

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

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。

### 2.3 K 均值聚类

k 均值聚类是一种常用的无监督学习算法，用于将数据集划分为 k 个不重叠的子集（簇），其基本原理如下：

- 初始化：随机选择 k 个初始聚类中心（centroid），每个聚类中心用于代表一个簇的中心点。
- 分配数据点：对于每个数据点，根据其与各个聚类中心的距离，将其分配到距离最近的聚类中心所属的簇中。
- 更新聚类中心：重新计算每个簇的中心点（聚类中心），方法是取该簇所有数据点的均值。
- 迭代：重复步骤 2 和步骤 3，直到达到停止条件，通常是聚类中心不再变化或达到最大迭代次数。
- 输出结果：最终得到 k 个簇，每个数据点都被分配到一个簇中，形成聚类结果。

## 三、 实验

### 3.1 线性回归算法

使用 Weka 平台的 Experiment 模块进行实验。回归数据集为 cpu.arrf、elevator.arrf、meta.arrf。属性选择策略分别为：M5 方法、贪心策略、无。比较领域为相对绝对误差。实验结果如下图 1。

从图 1 可以看出，采用不同属性选择策略做回归对结果影响不大。但算法在有的数据集上回归结果不太理想。

在显著水平 0.05 下的双边 t 检验结果为(0/3/0) 和 (1/2/0)，表示当线性回归属性选择策略为 M5 方法和贪心方法时，回归结果没有观察到显著性差异；当没有属性选择策略时，回归结果在一个数据集上与前两者方法有显著差异。

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1
Analysing:  Relative_absolute_error
Datasets:   3
Resultsets: 3
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       24-7-1 下午4:56


Dataset                   (1) functions |  (2) functi (3) functi
-------------------------------------------------------------------
cpu                      (100)    42.10 |    42.63      41.69
elevators                (100)    43.52 |    43.51      43.51
meta                     (100)   130.48 |   138.89     150.88 v
-------------------------------------------------------------------
                               (v/ /*) |    (0/3/0)     (1/2/0)
```

**图 1　线性回归算法实验结果**

线性回归优缺点：

优点：简单和易于理解、计算效率高、可解释性强、适用性广泛。

缺点：对非线性关系拟合能力有限、容易受异常值影响、可能存在欠拟合问题、无法处理多重共线性。

### 3.2 NB 算法

使用 Weka 平台的 Experiment 模块进行实验。分类数据集为 iris.arrf、car.arrf、glass.arrf、zoo.arrf、flags.arrf。实验结果如下图 2。

```
Tester:     weka.experiment.PairedCorrectedTT
Analysing:  Percent_correct
Datasets:   5
Resultsets: 1
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       24-7-1 下午5:16


Dataset                   (1) bayes.Naiv
----------------------------------------
iris                     (100)   95.53 |
car                      (100)   85.46 |
Glass                    (100)   49.45 |
zoo                      (100)   94.97 |
flags                    (100)   52.49 |
----------------------------------------
(v/ /*)                                |
```

**图 2　NB 算法实验结果**

从上述结果可以看出，NB 算法分类性能在一些简单小规模数据集上表现良好，但如果数据集规模偏大（glass、flags），分类精度则会严重下降。

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。

### 3.3 K 均值聚类

使用 Weka 平台的 Explore 模块进行实验。聚类数据集为 glass.arrf、wine.arrf。聚类数设置为数据集属性数。实验结果如下图 3 和图 4。

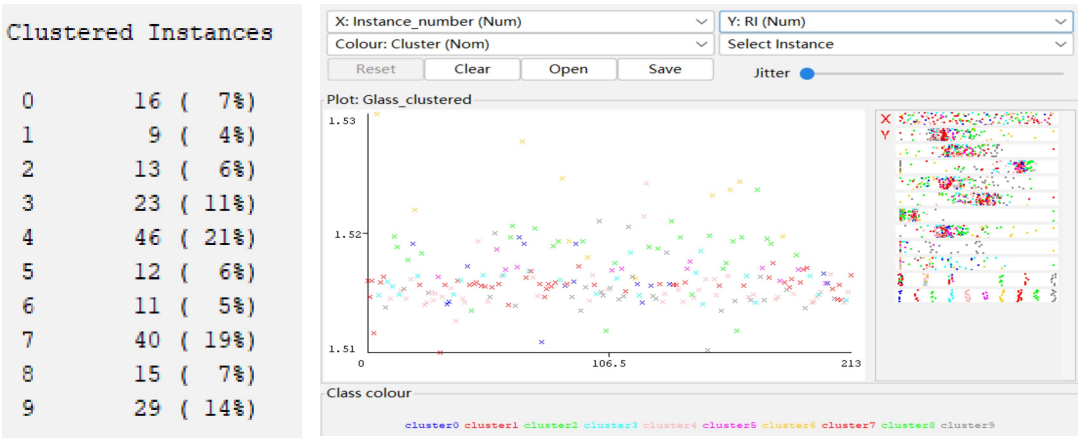


**图 3　K 均值聚类实验结果（glass）**

在 glass 数据集上，迭代次数和误差平方和为

```
Number of iterations: 16
Within cluster sum of squared errors: 38.47923197081721
```

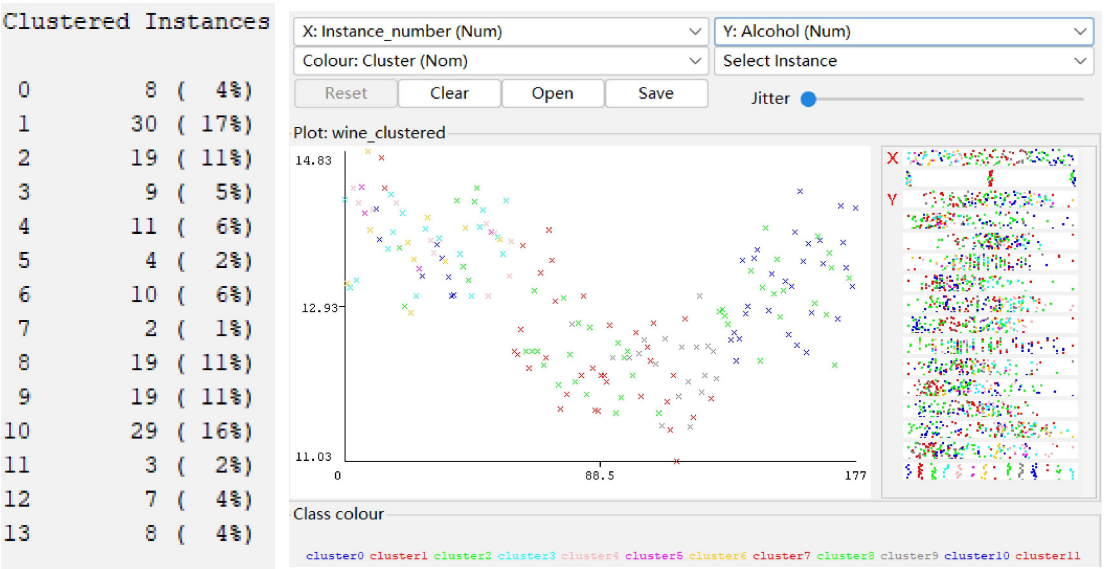


**图 4　K 均值聚类实验结果（wine）**

在 wine 数据集上，迭代次数和误差平方和为

```
Number of iterations: 9
Within cluster sum of squared errors: 33.803527454410165
```

误差平方和越小，表示聚类效果越好。对以上两个数据集的聚类结果表明 k 值的设置是合理的。

K 均值算法的优缺点：

优点：

- K 均值算法相对简单，易于实现和理解。
- 对大数据集有较好的可伸缩性和高效性。
- 在处理大型数据集和常规数据集时表现良好。

缺点：

- 初始中心点的选择对聚类结果影响较大，可能导致得到不同的聚类结果。
- 对非凸形状或者非球形状的簇效果不佳，容易导致簇之间的重叠或者不均匀分布。
- 对于 K 的选择敏感，且 K 均值算法无法自动确定最优的 K 值，需要先验地知道簇的数量。