哈爾濱工業大學（深圳）
Harbin Institute of Technology, Shenzhen

哈尔滨工业大学（深圳）课程论文

基于多元统计的大米图像特征数据分析

学期	2024 春季学期
课程名称	多元统计分析与软件
指导老师	冯峥晖
学院	理学院
专业	数据科学与大数据技术
队长	石金强 210810122
小组成员 1	程敬东 210810106
小组成员 2	梁祖瑜 210810112
小组成员 3	赵吴宇 210810101

小组分工

石金强 210810122

寻找数据集；编程实现 2 描述性统计，3.3 tSNE 降维方法，4.2 逻辑回归分类；整理其他成员的输出结果并进行分析；报告的写作。

程敬东 210810106

编程实现 3.1 主成分分析降维方法

梁祖瑜 210810112

编程实现 3.2 因子分析降维方法

赵吴宇 210810101

编程实现 4.1 判别分析分类

摘 要

大米是世界上生产和消费最广泛的谷类作物之一，也是我国的主要粮食作物之一。通过分析大米图像的特征数据，提出一个行之有效的大米分类方法，可作为大米的自动化标准分类的基础或用于市场商品真伪检测等。同时，可将这种方法平移到其他类似农产品上，如葡萄干、小番茄等。

小组利用主成分分析、因子分析和 tSNE 等数据降维方法先对特征数据进行降维，然后使用判别分析和逻辑回归方法对降维数据进行分类。上述降维和分类方法的结合均表现良好，其中以主成分分析和逻辑回归的结合表现最优。

【关键词】大米分类；主成分分析；因子分析；tSNE；判别分析；逻辑回归

目 录

1. 绪论

1.1 背景介绍

在全世界的谷物产品中，大米是继小麦和玉米之后最重要的产品。大米富含碳水化合物和淀粉，营养丰富，经济实惠，在我国乃至全世界的人类营养中都具有重要地位。此外，大米也有各种各样的用途，例如工业领域中酒、醋类的生产，作为养殖业的饲料等。

我国是全球大米的第一生产国，大米产量稳定在 21000 万吨左右，在我国主要谷物中的产量仅次于玉米。

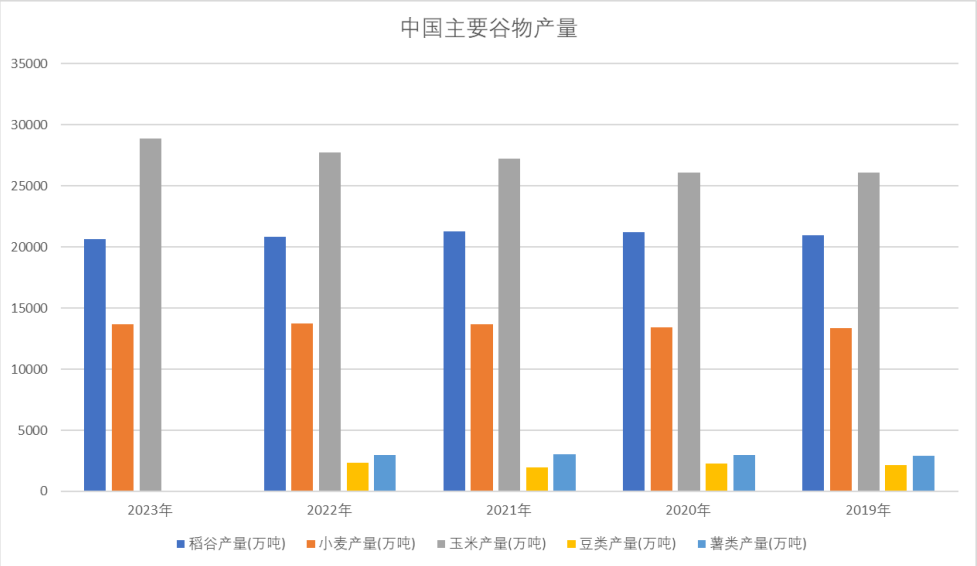

图 1：中国主要谷物产量[1]

在由国家市场监督管理总局和中国国家标准化管理委员会发布的 GB/T 1354-2018 大米标准中，包括物理外观、品尝评分、色泽和气味等各类标准。而从日常消费的角度来看，对于在市场上出售的大米，消费者最关注的是物理外观标准，例如籼米外观是又细又长，常见的是泰国香米；粳米外观是又短又圆，常见的是东北大米。对此，我们可以使用机器视觉系统和图像处理技术对大米的物理外观进行分析，从而实现自动化的大米标准判断。除此以外，我们还可以将这种外观判断流程直接平移到其他农产品上，例如葡萄干等。同时，这种方法也可以用于商品真伪判断等。

1.2 研究内容

大米类别也是大米标准判断的重要因素，我们希望通过对大米的物理外观进行分析，从而对大米进行分类。

如果对每一粒大米都进行物理外观的实际测量，例如测量长度和宽度等，工作量实在是太大。因此我们希望通过机器视觉系统和图像处理技术能够一次性且大量地获得大米的物理外观信息。

如下图所示，Ilkay CINAR 和 Murat KOKLU 利用机器视觉系统一次性采集了大量大米的图像，然后通过图像处理技术，最终提取了 7 个特征值。

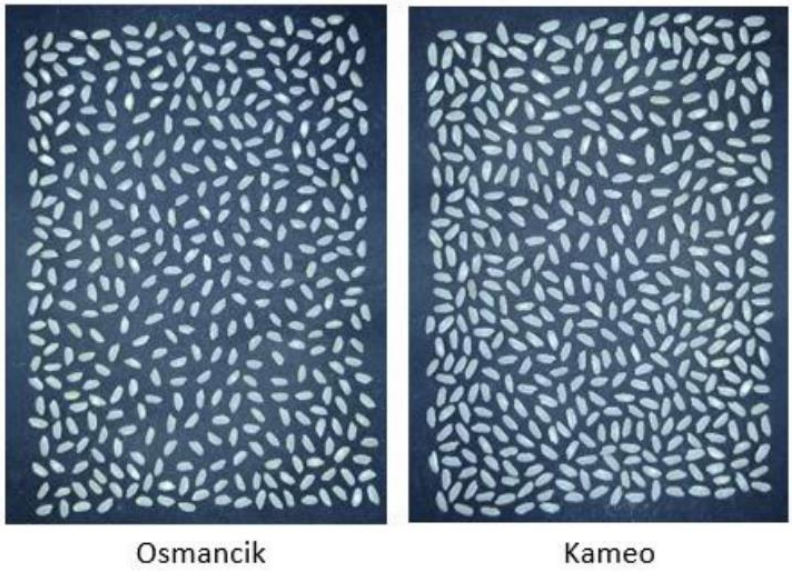

图 2：大米图像示例[2]

得益于他们的研究，我们直接得到了由大米图像提取出的用于描述大米物理外观的特征数据。因此，我们的研究任务就是根据这些特征数据提出一种对能够对大米做一个有效的分类的方法。

1.3 研究方法

首先，我们将对这些特征数据进行一些描述性统计。其次，我们将利用数据降维方法或模型，例如：主成分分析，tSNE 等，对特征数据进行降维。最终，我们对降维后的数据构建分类模型，例如：判别分析，逻辑回归等。

1.4 数据集说明

本次课程论文的数据集来源于 UCI 数据库[3]，其描述了在土耳其种植的 Cammeo 和 Osmancik 两种大米的信息，每条数据记录了由大米图像提取出的 7 个

特征属性和 1 个类别属性，共有 3810 条数据，其中 Cammeo 大米有 1630 条数据，Osmancik 大米有 2180 条数据。下面给出两种大米的数据示例：

表 1：数据集示例

Area	Perimeter	Major_Axis_Length	Minor_Axis_Length
15231	525.5789795	229.7498779	85.09378815
11434	404.7099915	161.0792694	90.86819458
Eccentricity	Convex_Area	Extent	Class
0.928882003	15617	0.572895527	Cammeo
0.825692177	11591	0.802949429	Osmancik

其中，各变量含义如下：

表 2：数据集变量含义

变量	含义
Area	米粒面积（区域中的实际像素数）
Perimeter	米粒周长[4]（区域边界周围的距离）
Major_Axis_Length	主轴长度（米粒上可以画出的最长直线）
Minor_Axis_Length	小轴长度（米粒上可以画出的最短直线）
Eccentricity	离心率（近似于米粒的椭圆的离心率）
Convex_Area	凸包面积（米粒的最小凸包的像素数）
Extent	米粒区域与分割边界框的像素比例
Class	米粒品种（Cammeo 和 Osmancik）

2. 描述性统计

2.1 基本统计量

计算整个数据集的最小值、均值、最大值、标准差、偏度和峰度等基本统计量。

表 3：数据集的基本统计量

属性	最小值	均值	最大值	标准差	偏度	峰度
Area	7551	12667.73	18913	1723.37	0.3252	-0.4311

Perimeter	359.10	454.24	548.45	35.60	0.2214	-0.8402
Major_Axis_Length	145.26	188.78	239.01	17.45	0.2602	-0.9518
Minor_Axis_Length	59.53	86.31	107.54	5.73	-0.1349	0.5261
Eccentricity	0.7772	0.8869	0.9480	0.0208	-0.4492	0.0711
Convex_Area	7723	12952.50	19099	1776.97	0.3198	-0.4658
Extent	0.4974	0.6619	0.8611	0.0772	0.3438	-1.0301

观察上表结果，可以看到各属性的数据数值差异巨大，这是因为它们之间的量纲不同，所以在后续的分析处理中要对数据进行去量纲化或标准化。

2.2 相关性分析

这里用 R 语言中 GGally 库的 ggpairs()函数对属性间的相关性进行分析和可视化。

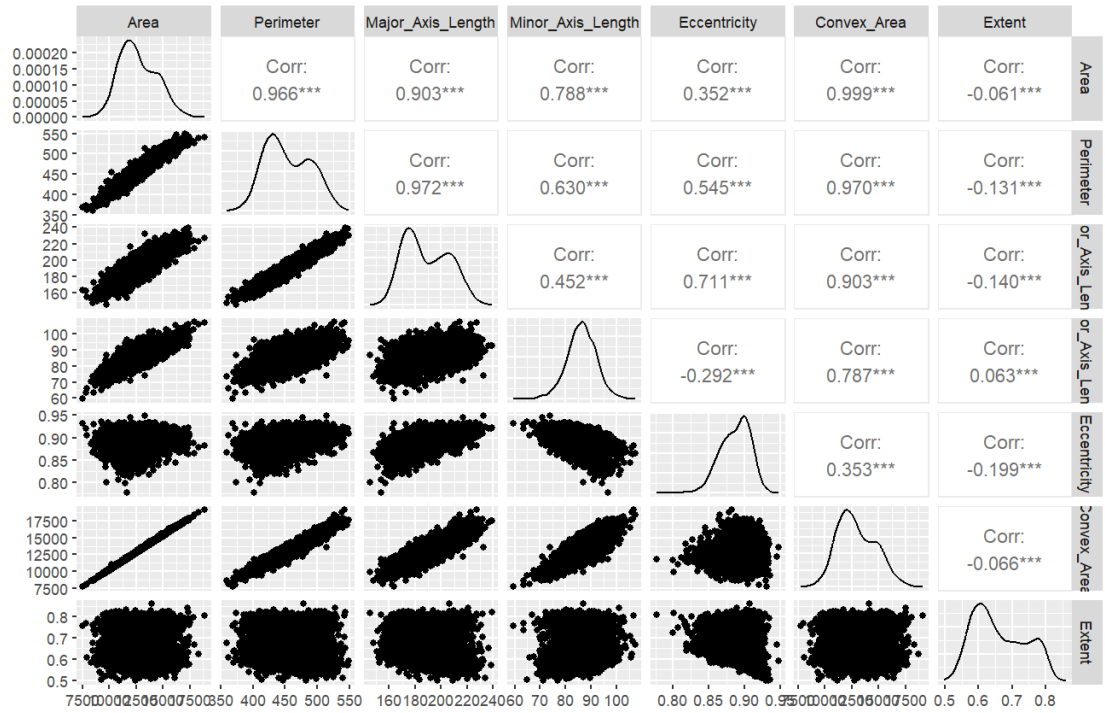

图 3：属性间相关性的可视化

观察上图结果，属性之间呈现有高度的相关性，并且都是显著的：例如 Perimeter 属性和 Convex_Area 属性之间的相关性达到了 0.999***，说明这两个属性之间几乎就是线性相关。这些变量间的高相关性，给了我们充足的理由对数据进行降维处理。

3. 特征数据降维

3.1 主成分分析[5]

主成分分析（Principal component analysis， PCA）是由 Pearson（1901）提出，后来被 Hotelling （1933）进一步系统发展的。主成分分析是利用降维的思想，在损失很少信息的前提下，将多个指标化为少数几个综合指标的一种统计分析方法，通常把转化生成的综合指标称为主成分，其中每个主成分都是原始变量的线性组合，且各个主成分之间互不相关，使得主成分集中了原始变量绝大部分信息，并且具有某些更优越的性能，在实际问题的研究中，研究多指标的问题是经常遇到的问题，这样利用主成分分析方法就可以只考虑少数几个主成分而不至于损失太多信息，从而抓住主要矛盾，揭示事物内部变量之间的规律性，同时使问题得到简化，提高分析效率。

3.1.1 总体主成分

3.1.1.1 主成分的定义

设$X=(X_1,X_2,\cdots,X_p)'$为p维随机向量，称$Z_i=a_i'X$为X的第i主成分，$i=1,2,\cdots,p$，如果：

（1） $a_i'a_i=1$，$i=1,2,\cdots,p$；

（2） 当$i>1$时，$a_i'\Sigma a_j=0$，$j=1,\cdots,i-1$；

（3） $Var(Z_i)=\max\limits_{a'a=1,a'\Sigma a_j=0(j=1,\cdots,i-1)} Var(a'X)$

3.1.1.2 主成分的计算方法

令Σ是p维随机向量$X=(X_1,\cdots,X_p)'$的协方差矩阵，具有特征值和单位特征向量序列$(\lambda_1,a_1),(\lambda_2,a_2),\cdots,(\lambda_p,a_p)$，其中$\lambda_1\geq\lambda_2\geq\cdots\geq\lambda_p\geq0$和$a_i=(a_{i1},a_{i2},\cdots,a_{ip})'$，则X的第i个主成分为：

$$Z_i=a_i'X=a_{i1}X_1+a_{i2}X_2+\cdots+a_{ip}X_p,\qquad i=1,\cdots,p$$

并进一步有：

$$Var(Z_i)=a_i'\Sigma a_i=\lambda_i,\qquad i=1,\cdots,p$$

$$\mathrm{Cov}(Z_i, Z_k) = a_i' \Sigma a_k = 0, \qquad i \neq k$$

3.1.1.3 主成分个数选择依据

当原变量X的维数p很大时，主成分分析的目的是为了简化数据结构（即减少变量的个数），即在不损失较多信息的情况下，用很少的几个主成分去代表原变量X。因此在实际应用中一般不用p个主成分，而选用前$m(m < p)$个主成分来代替。为了解决这个问题，下面介绍贡献率和累计贡献率的定义：

第i个主成分方差在总方差中所占的比例，称为第i个主成分的贡献率，定义为：

$$\frac{\lambda_i}{\lambda_1 + \lambda_2 + \cdots + \lambda_p}, \qquad i = 1, \cdots, p$$

把前m个主成分方差在总方差所占的比例，称为累计贡献率，定义为：

$$f_m = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{p} \lambda_i}, \qquad m < p$$

贡献率表示的是主成分综合$X_1, \cdots, X_p$信息的能力，贡献率越大，能力越强。累计贡献率f_m表示的就是前m个主成分在$X_1, \cdots, X_p$所占信息的比重。在实际应用中，通常取m使得累计贡献率f_m达到 85%以上，表面前m个主成分基本包含了全部变量的信息。

3.1.2 基于标准化的总体主成分

在实际应用中，当变量的测量尺度不同时，有不同的量纲，通过协方差矩阵Σ来计算主成分时总是优先考虑方差大的变量，导致不合理的结果。为了消除量纲带来的影响，有必要对变量X进行标准化：

$$X_i^* = \frac{X_i - E(X_i)}{\sqrt{\mathrm{Var}(X_i)}} = \frac{X_i - \mu_i}{\sqrt{\sigma_{(ii)}}}$$

那么标准化后的变量X^*的协方差矩阵Σ^*其实就是原始变量X的相关系数矩阵R。因此，标准化后变量X^*的主成分可以通过原始变量X的相关系数矩阵R出发进行计算。对$i = 1, \cdots, p$，若(λ_i^*, a_i^*)表示R的特征值和特征向量序列，则基于X^*的主成分为：

$$Z_i^* = a_i^{*\prime} X^*, \qquad i = 1, \cdots, p$$

3.1.3 样本主成分分析

上面介绍的是总体主成分分析，而样本主成分分析就是将总体主成分的内容平移到样本主成分上。具体来说，样本主成分就是用样本协方差矩阵S和样本相关系数矩阵$\hat{R}$来代替总体主成分中的协方差矩阵Σ和相关系数矩阵R。

3.2 因子分析[5]

因子分析是一种数据简化或降维的技术，它通过研究众多变量之间的内部依赖关系，探求观测数据中的基本结构，并用少数几个假想变量来表示其基本的数据结构。

3.2.1 正交因子分析模型

首先引入一些记号，令：

$$\mathbf{X} = (\mathbf{X_1}, \cdots, \mathbf{X_p})' \qquad \boldsymbol{\mu} = (\mu_1, \cdots, \mu_p)'$$

$$\mathbf{F} = (\mathbf{F_1}, \cdots, \mathbf{F_k})' \qquad \boldsymbol{\varepsilon} = (\varepsilon_1, \cdots, \varepsilon_p)'$$

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & \ddots & \vdots \\ a_{p1} & \cdots & a_{pk} \end{pmatrix} \qquad \Psi = \begin{pmatrix} \psi_1 & & \\ & \ddots & \\ & & \psi_p \end{pmatrix}$$

下面给出正交因子分析模型的定义：

设$\mathbf{X}$为p维随机向量，$\boldsymbol{\mu}$为其均值向量。若$\mathbf{X}$可随机表示为：

$$\mathbf{X} = \mathbf{AF} + \boldsymbol{\mu} + \boldsymbol{\varepsilon}$$

其中$\mathbf{A}$是一个$p \times k$的常数矩阵($k < p$)，$\mathbf{F}$和$\boldsymbol{\varepsilon}$分别为k维和p维随机向量，且

$$E(\mathbf{F}) = 0, \mathrm{Cov}(\mathbf{F}) = \mathbf{I_k}$$

$$E(\boldsymbol{\varepsilon}) = 0, \mathrm{Cov}(\boldsymbol{\varepsilon}) = \Psi$$

$$\mathrm{Cov}(\mathbf{F}, \boldsymbol{\varepsilon}) = 0$$

则称上述表达式定义的模型为$\mathbf{X}$的正交因子分析模型，称$\mathbf{A}$为因子载荷矩阵，称$\mathbf{F}$为$\mathbf{X}$的公共因子向量，称$\boldsymbol{\varepsilon}$为$\mathbf{X}$的特殊因子向量。

3.2.2 因子载荷矩阵的估计方法

因子载荷矩阵的估计方法有：主成分法、主因子法和极大似然法。

3.2.2.1 主成分法

设随机向量$\mathbf{X}$的协方差矩阵Σ，$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$为$\Sigma$的特征值，$\mathbf{u_1}, \cdots, \mathbf{u}_p$为对应的标准化特征向量，则

$$\Sigma = AA' + \Psi = \mathbf{U\Lambda U}' = \sum_{i=1}^{p} \lambda_i \mathbf{u_i u_i'}$$

这里，$\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_p)$，$\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_p)$。

若前k个特征值$\lambda_1, \cdots, \lambda_k$的累计贡献率很高，令：

$$\mathbf{U_1} = (\mathbf{u}_1, \cdots, \mathbf{u}_k) \qquad \mathbf{U_2} = (\mathbf{u}_{k+1}, \cdots, \mathbf{u}_p)$$

$$\mathbf{\Lambda_1} = \mathrm{diag}(\lambda_1, \cdots, \lambda_k) \qquad \mathbf{\Lambda_2} = \mathrm{diag}(\lambda_{k+1}, \cdots, \lambda_p)$$

于是：

$$\Sigma = \mathbf{U_1 \Lambda_1 U_1}' + \mathbf{U_2 \Lambda_2 U_2}'$$

由于$\mathbf{U_1 \Lambda_1 U_1}'$解释了$\mathbf{X}$的主要相关关系，故因子载荷矩阵$\mathbf{A}$的估计为：

$$\widehat{\mathbf{A}} = \mathbf{U_1 \Lambda_1^{1/2}}$$

特殊因子的方差ψ_i可用$\mathbf{U_2 \Lambda_2 U_2}'$的第i个对角元素来估计。

3.2.2.2 主因子法

主因子法是从相关系数矩阵$\mathbf{R}$出发来估计因子载荷矩阵$\mathbf{A}$的。主因子法的迭代算法如下：

1. 给定特殊因子方差一个初值$\Psi(0)$；
2. 对$\mathbf{R}^* = \mathbf{R} - \Psi(0)$进行主成分分解，并获得估计$\widehat{\mathbf{A}}$；
3. 令$\Psi(0) = \mathrm{diag}(\mathbf{R} - \widehat{\mathbf{A}}\widehat{\mathbf{A}}')$，重复步骤 2，直到收敛为止。

3.2.2.3 极大似然法

极大似然法要求$\mathbf{X}$来自p元正态分布$N_p(\mu, \Sigma)$。

极大似然法的估计结果如下：

$$\begin{cases} \Psi = S - \mathbf{AA}' \\ S[\mathbf{AA}' + \Psi]^{-1}\mathbf{A} = \mathbf{A} \end{cases}$$

其中S为样本协方差矩阵。该方程组没有显示解，需要用迭代算法完成。

3.2.3 因子旋转

由于因子载荷矩阵是不唯一的，所以可对因子载荷矩阵进行旋转。旋转的目的是使因子载荷矩阵的结构简化，使载荷矩阵每列或行的元素平方值向 0 和 1 两级分化。常用的旋转法有：方法最大（varimax）法、四次方最大法和等量最大法。

3.2.4 因子得分

在实际应用中，需要估计公共因子$\mathbf{F}$，给出公共因子的值，即因子得分。主要有两种估计因子的方法，分别是 Thomson 因子得分和 Bartlett 因子得分。

3.2.4.1 Thomson 因子得分

Thomson 因子得分假设因子分析模型中的$\mathbf{X}$来自p元正态分布$\mathrm{N_p}(\boldsymbol{\mu},\boldsymbol{\Sigma})$，公共因子$\mathbf{F}$的先验分布为$\mathrm{N_k}(0,\mathbf{I_k})$，则 Thomson 因子得分为：

$$\mathbf{F}^{*}=\mathbf{A}'(\mathbf{A}\mathbf{A}'+\Psi)^{-1}(\mathbf{X}-\boldsymbol{\mu})$$

3.2.4.2 Bartlett 因子得分

Thomson 因子得分假设因子分析模型中的$\mathbf{X}$来自p元正态分布$\mathbf{X}-\boldsymbol{\mu}\sim\mathrm{N_p}(\mathbf{A}\mathbf{F},\boldsymbol{\Sigma})$，则 Bartlett 因子得分：

$$\hat{\mathbf{F}}=(\mathbf{A}'\Psi^{-1}\mathbf{A})^{-1}\mathbf{A}'\Psi^{-1}(\mathbf{X}-\boldsymbol{\mu})$$

3.3 tSNE

tSNE 方法是数据挖掘中常用的数据降维方法，主要用于高维数据的低维可视化（平面、立体图像）。tSNE 应用广泛，性能优秀，例如在 MINST 手写数据集的聚类分析中，tSNE 数据降维的效果要远远优于主成分分析方法。因此，考虑将其应用到本次课程论文。

3.3.1 tSNE 介绍[6]

tSNE 的目的是把X（高维原始数据）转换成Z（指定低维数据）。

在给定D维的高维数据点$\mathrm{X_1},\mathrm{X_2},\cdots,\mathrm{X_N}$，将距离转换为条件概率来表达点与点之间的相似度：

$$p_{j|i} = \begin{cases} \dfrac{\exp\left(-\left||X_i - X_j|\right|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\left||X_j - X_k|\right|^2 / 2\sigma_i^2\right)}, & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

考虑在指定d维的低维数据$z_1, z_2, \cdots, z_N$，我们同样考虑将距离转换为条件概率:

$$q_{ij} = \frac{\left(1 + \left||z_i - z_j|\right|^2\right)^{-1}}{\sum_k \sum_{l \neq k} \left(1 + \left||z_k - z_l|\right|^2\right)^{-1}}$$

接下来，我们的目的是令p_{ij}的分布（记为 P）和q_{ij}的分布（记为 Q）接近，也就是最小化如下的 KL 度量：

$$L = KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

可以使用基于梯度的优化方法解上述问题：

$$\frac{\partial L}{\partial z_i} = \sum_j (p_{ij} - q_{ij})(z_i - z_j)\left(1 + \left||z_i - z_j|\right|^2\right)^{-1}, i = 1, \cdots, N$$

3.4 降维结果及分析

3.4.1 主成分分析结果

使用 R 语言对数据的特征属性进行标准化的样本主成分分析，结果如下：

表 4：Importance of components

Comp	标准差	贡献率	累计贡献率
Comp.1	2.1398550	0.6541399	0.6541399
Comp.2	1.2246463	0.2142512	0.8683911
Comp.3	0.9491077	0.1286865	0.9970776
Comp.4	0.108424219	0.001679402	0.998757020
Comp.5	0.0788457673	0.0008880936	0.9996451140
Comp.6	0.0453063366	0.0002932377	0.9999383518

Comp.7	0.02077348	0.00006164823	1

观察累计贡献率，取前 2 个主成分时，贡献率就已经达到 86.8%，因此，我们选择 2 个主成分就已经足够用于后续的分析。但取前 3 个主成分时，贡献率就达到了 99.7%，这表明取前 3 个主成分就几乎完全囊括了数据集的信息。因此在后续的分类建模中，我们可以比对 2 个主成分和 3 个主成分哪个效果更好。下面给出前 3 个主成分的载荷，并加以解释：

表 5：主成分载荷矩阵

属性	Comp.1	Comp.2	Comp.3
Area	0.461	0.124	
Perimeter	0.464		
Major_Axis_Length	0.447	-0.213	-0.122
Minor_Axis_Length	0.322	0.567	0.213
Eccentricity	0.227	-0.673	-0.298
Convex_Area	0.462	0.123	
Extent		0.382	-0.922

以$X_1, X_2, \cdots, X_7$简记属性$\text{Area}, \text{Perimeter}, \cdots, \text{Extent}$，则前三个主成分为：

$$Z_1^* = 0.461X_1^* + 0.464X_2^* + 0.447X_3^* + 0.322X_4^* + 0.227X_5^* + 0.462X_6^*$$

$$Z_2^* = 0.124X_1^* - 0.213X_3^* + 0.567X_4^* - 0.673X_5^* + 0.123X_6^* + 0.382X_7^*$$

$$Z_3^* = -0.122X_3^* + 0.213X_4^* - 0.298X_5^* - 0.922X_7^*$$

Z_1^*各个分量的系数值符号相同且系数值相差不大，它反映了米粒整体的综合指标：米粒越大，它$X_1^*, \cdots, X_6^*$的特征数据就比较大。因此称第一主成分为大小因子。

Z_2^*中X_3^*和X_5 $*$对应的系数值为负，其他为正，它反映了米粒的形态：米粒越细越长，拟合椭圆的离心率就越大，它X_3^*和X_5^*的特征数据就比较大，进而Z_2^*就越小。因此称第二主成分为形态因子。

Z_3^*中X_7^*对应的系数占了大部分，它描述了米粒的完整程度：如果米粒光滑圆润，那么在图像分割的边界框中，米粒应该占据大部分，X_7^*特征数据应该接近于 1；反之，若米粒破损，边角崎岖，那么在图像分割的边界框，米粒占据的部分就变小了，X_7^*特征数据也变小了。因此称第三主成分为完整因子。

同时，我们给出取前两个主成分的可视化：

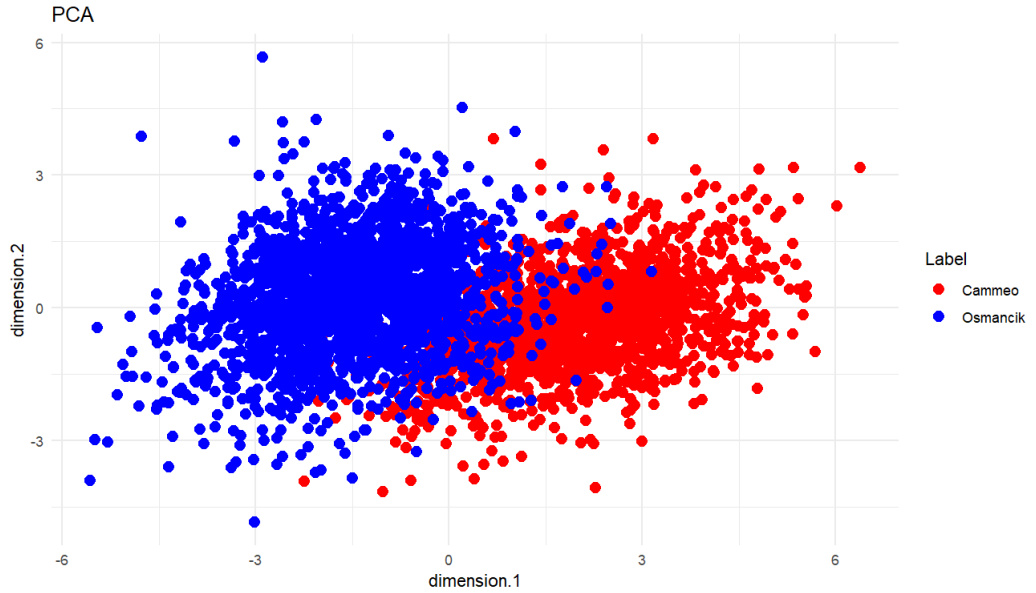

图 4：主成分降维可视化

3.4.2 因子分析结果

使用 R 语言对数据的特征属性进行因子分析，将数据维度降至 2 维，结果如下：

表 6：因子分析载荷矩阵

属性	因子 1	因子 2
Area	0.977	0.208
Perimeter	0.903	0.418
Major_Axis_Length	0.796	0.603
Minor_Axis_Length	0.898	-0.434
Eccentricity	0.151	0.981
Convex_Area	0.977	0.209
Extent		-0.201

第一公共因子除了属性 Eccentricity 和 Extent 外，在其他属性上的载荷都很大，体现了米粒的大小程度。

第二公共因子在属性 Eccentricity 上的载荷很大，体现了米粒的细长程度。

计算其 Thomson 因子得分，并进行可视化，结果如下：

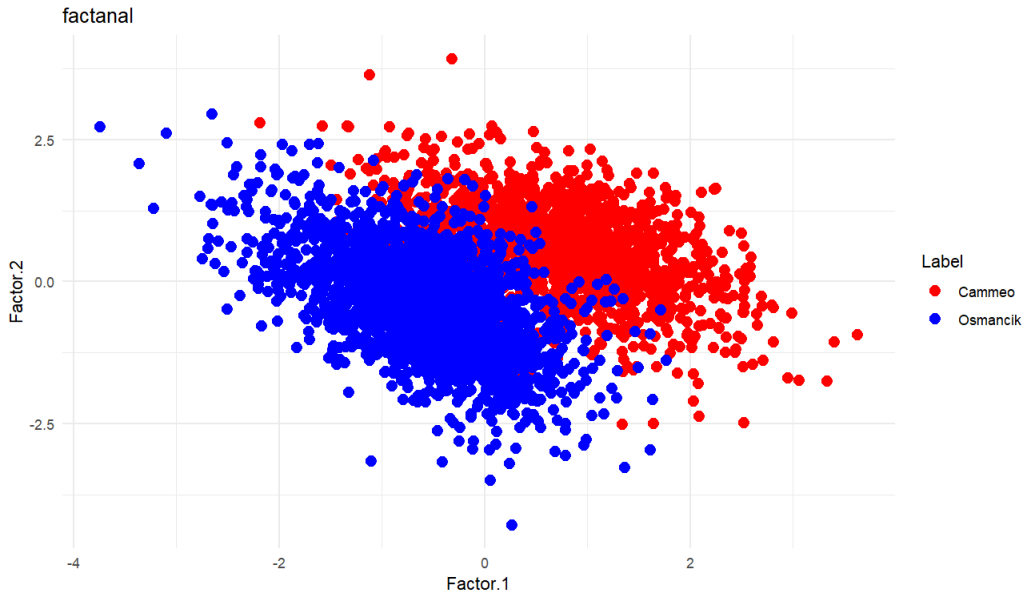

图 5：因子分析结果可视化

3.4.3 tSNE 结果

使用 Python 中 sklearn 库的 TSNE 类对标准化的特征数据进行 tSNE 降维，并进行可视化，结果如下：

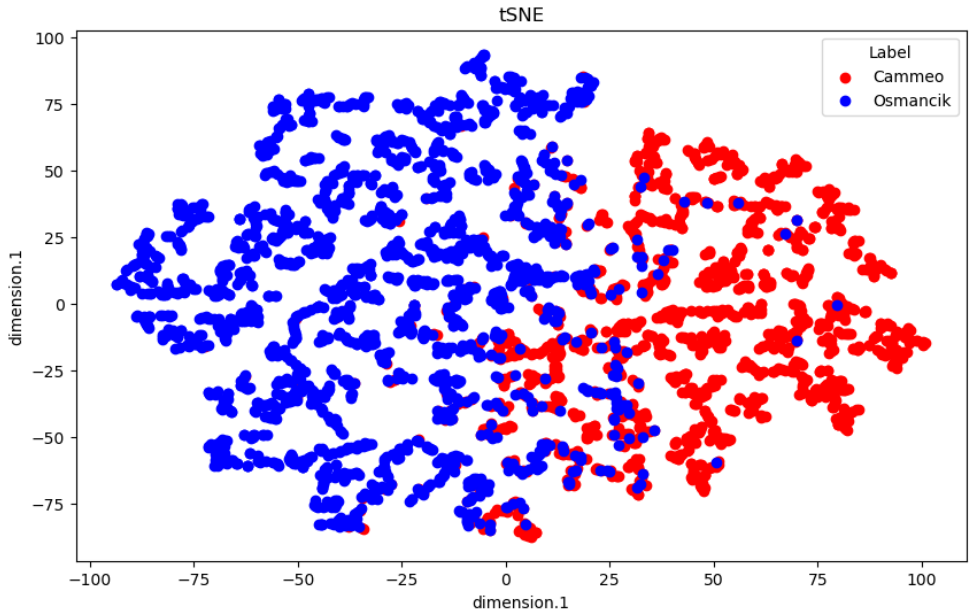

图 6：tSNE 降维可视化

3.4.4 降维方法比较

以降维方法的可视化结果进行比较，可以明显看到 tSNE 相比于主成分分析和因子分析能够将数据分得更开，而因子分析的结果又优于主成分分析。但

对于具体的分类性能，还需要进行真正的分类尝试。

4. 降维数据分类

4.1 判别分析

判别分析是指使用具有类别信息的观测数据，建立一个分类器或分类法则，可以最大可能的区分事先定义的类。例如，假设有总体π_1和π_2，给定一个新个体$\mathbf{x} = (x_1, \cdots, x_p)$，应该将其分为哪个总体？判别分析给出的解决方案是，寻找一个判别方法把$\mathbb{R}^p$空间分为两个区域R_1和R_2，如果新个体$\mathbf{x} \in \mathbf{R_1}$，就把它分为总体$\pi_1$，否则分为总体$\pi_2$。

4.1.1 两总体判别分析准则

假设总体π_1服从$f_1(X)$分布，总体π_2服从$f_2(X)$分布。称先验概率q_1, q_2为：q_1是观测值来自总体π_1的概率，q_2是观测值来自总体π_2的概率，它们满足$q_1 + q_2 = 1$。在判别过程中会发生两种错误：①个体来自总体π_1，却被误判为总体π_2；②个体来自总体π_2，却被误判为总体π_1。称错误损失$C(2|1), C(1|2)$为：$C(2|1)$是发生错误①带来的损失，$C(1|2)$是发生错误②带来的损失。一个好的判别方法的标准就是令错判带来的损失最小。

4.1.2 两总体判别分析情形

若个体来自总体π_1，其被正确判别的概率为：$Pr(1|1, R) = \int_{R_1} f_1(\mathbf{x})d\mathbf{x}$。

若个体来自总体π_1，其被错误判别的概率为：$Pr(2|1, R) = \int_{R_2} f_1(\mathbf{x})d\mathbf{x}$。

若个体来自总体π_2，其被正确判别的概率为：$Pr(2|2, R) = \int_{R_2} f_2(\mathbf{x})d\mathbf{x}$。

若个体来自总体π_2，其被错误判别的概率为：$Pr(1|2, R) = \int_{R_1} f_2(\mathbf{x})d\mathbf{x}$。

我们将上述判别概率与先验概率结合：

若观测值来自总体π_1，且被正确判别的概率为：$q_1 Pr(1|1, R)$。

若观测值来自总体π_2，且被正确判别的概率为：$q_2 Pr(2|2, R)$。

若观测值来自总体π_1，但被错误判别的概率为：$q_1 Pr(2|1, R)$。

若观测值来自总体π_2，但被错误判别的概率为：$q_2 Pr(1|2, R)$。

再结合错判损失，则错判的平均损失为：

$$\mathrm{ECM}(R_1, R_2) = C(2|1)\Pr(2|1, R)q_1 + C(1|2)\Pr(1|2, R)q_2$$

而判别分析的目的就是找最优判别方法，把空间分成R_1和R_2，是得错判的平均损失达到最小。

4.1.3 两个多元正态总体的判别

当总体π_1和π_2就是多元正态总体$N_p(\mu_1, \Sigma_1)$和$N_p(\mu_2, \Sigma_2)$时，我们有如下两个判别准则。

4.1.3.1 线性判别准则（LDA）

当$\Sigma_1 = \Sigma_2 = \Sigma$时，有线性判别准则（LDA）如下：

定义线性判别函数：

$$x'\Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 + \mu_2)'\Sigma^{-1}(\mu_1 - \mu_2)$$

其给出的最好的判别区域为：

$$R_1: x'\Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 + \mu_2)'\Sigma^{-1}(\mu_1 - \mu_2) \geq \ln k$$

$$R_2: x'\Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 + \mu_2)'\Sigma^{-1}(\mu_1 - \mu_2) < \ln k$$

若先验概率q_1和q_2已知，则$k = \frac{q_2 c(1|2)}{q_1 c(2|1)}$。

4.1.3.2 二次判别准则（QDA）

当$\Sigma_1 \neq \Sigma_2$时，有二次判别准则（QDA）如下：

定义二次判别函数：

$$\delta(\mathbf{x}) = -\frac{1}{2}\mathbf{x}'(\Sigma_1^{-1} - \Sigma_2^{-1})\mathbf{x} + (\boldsymbol{\mu}_1'\Sigma_1^{-1} - \boldsymbol{\mu}_2'\Sigma_2^{-1})\mathbf{x} - \xi$$

$$\xi = \frac{1}{2}\ln\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right) + \frac{1}{2}(\boldsymbol{\mu}_1'\Sigma_1^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2'\Sigma_2^{-1}\boldsymbol{\mu}_2)$$

其给出的最好的判别区域为：

$$R_1 = \{\mathbf{x}:\ \delta(\mathbf{x}) \geq \ln k\}$$

$$R_1 = \{\mathbf{x}:\ \delta(\mathbf{x}) < \ln k\}$$

若先验概率q_1和q_2已知，则$k = \frac{q_2 c(1|2)}{q_1 c(2|1)}$。

4.2 逻辑回归

逻辑回归(Logistic Regression)，是一种广义的线性回归分析模型，属于机器学习中的监督学习，其常用于解决二分类问题，例如：预测借贷人是否违约，判断邮件是否为垃圾邮件等。逻辑回归简单、实用、高效、应用广泛。因此，考虑在本次课程论文中使用。

4.2.1 逻辑回归介绍

逻辑回归的思想来自于线性回归，但不同于线性回归。在线性回归中，我们试图拟合一个线性方程来预测一个连续的输出值。而在逻辑回归中，我们不是直接预测输出值，而是预测输出值属于某一特定类别的概率。

对于二分类问题，不妨记其中类C_1的标签为 0，类C_2的标签为 1。在给定一条数据$\mathbf{x}$后，通过以下的逻辑回归模型计算它标签为 1 的条件概率：

$$\Pr(1|\mathbf{x}) = \sigma(\mathbf{w}'\mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}'\mathbf{x}+b)}}$$

其中，$\mathbf{w}$是权重，b是偏置。下面给出计算$\mathbf{w}$和b的方法。

假设现有数据集$\{\mathbf{x}_i, y_i\}, i = 1, \cdots, N$，其中$y_i \in \{0,1\}$，则有似然函数：

$$p(\mathbf{y}|\mathbf{w}, b) = \prod_{i=1}^{N} f_i^{y_i}\{1 - f_i\}^{1-y_i}$$

其中$\mathbf{y} = (y_1, \cdots, y_N)'$，$f_i = \sigma(\mathbf{w}'\mathbf{x_i} + b)$。

接下来的问题就是极大化似然函数求解$\mathbf{w}$和b：

$$\mathbf{w}, b = \text{argmax}_{w,b}\ p(y|w, b)$$

我们其转换为等价的极小化的问题：

$$\mathbf{w}, b = \text{argmin}_{w,b} - \ln p(\mathbf{y}|\mathbf{w}, b) = -\sum_{i=1}^{N}\{y_i \ln f_i + (1 - y_i)\ln(1 - f_i)\}$$

在实际应用中可以通过梯度下降法求解上述极小化问题：

记$E(\mathbf{w}, b) = -\ln p(\mathbf{y}|\mathbf{w}, b)$，则$E(\mathbf{w}, b)$的梯度为：

$$\nabla E(\mathbf{w}, b) = \sum_{i=1}^{N}(f_i - y_i)x_i$$

4.3 分类结果及分析

4.3.1 判别分析结果

以混淆矩阵的形式给出判别分析的结果：

表 7：判别分析结果

降维数据	判别准则	混淆矩阵	
2 个主成分	LDA	1500	130
		159	2021
	QDA	1510	120
		168	2012
3 个主成分	LDA	1514	116
		162	2018
	QDA	1514	116
		171	2009
因子分析	LDA	1512	118
		157	2023
	QDA	1518	112
		170	2010
tSNE	LDA	1502	128
		201	1979
	QDA	1492	138
		184	1996

4.3.2 逻辑回归结果

以混淆矩阵的形式给出逻辑回归的结果：

表 8：逻辑回归结果

降维数据	混淆矩阵	
2 个主成分	1483	147
	135	2045

3 个主成分	1491	139
	136	2044
因子分析	1492	138
	137	2043
tSNE	1438	192
	149	2031

4.3.3 评价指标

上述混淆矩阵的含义如下：

表 9：大米分类的混淆矩阵

		预测结果	
		Cammeo	Osmancik
实际结果	Cammeo	TP	FP
	Osmancik	FN	TN

其中，TP、FP、FN、TN 的含义如下：

TP(True Positive)是指真实值是 positive，模型认为是 positive 的数量。

FP(False Positive)是指真实值是 negative，模型认为是 positive 的数量，对应于统计学上的第一类错误。

FN(False Negative)是指真实值是 positive，模型认为是 negative 的数量，对应于统计学上的第二类错误。

TN(True Negative)是指真实值是 negative，模型认为是 negative 的数量。

混淆矩阵常见的评价指标有：

- 准确率(Accuracy)

  其含义是所有预测对的样本占所有样本的比例

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

- 精确率(Precision)，也称查准率

  其含义是预测对的正样本总数占预测的正样本总数的比例

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- 真阳性率(True Positive Rate, TPR)，也称灵敏度(Sensitivity)、召回率(Recall)、查全率

  其含义是预测对的正样本总数占所有正样本总数的比例

$$\mathrm{TPR} = \mathrm{Sensitivity} = \mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

- 真阴性率(True Negative Rate, TNR)，也称特异度(Specificity)

  其含义是预测对的负样本占全体负样本的比例

$$\mathrm{TNR} = \mathrm{Specificity} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

- 假阳性率(False Positive Rate, FPR)

  其含义是预测错的正样本占全体正样本的比例

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}}$$

在实际中，往往不会用到上述全部指标，通常以下面两种指标组合为主：

- Accuracy + Precision + Recall
- TPR + TNR + FPR

4.3.4 分类方法比较

下面给出各方法的评价指标：

表 10：各方法评价指标

降维数据	分类方法	Accuracy	Precision	Recall/TPR	TNR	FPR
2 个主成分	LDA	0.9241	0.9202	0.9042	0.9396	0.0729
	QDA	0.9244	0.9264	0.8999	0.9437	0.0771
	逻辑回归	0.926	0.9098	0.9166	0.9329	0.0619
3 个主成分	LDA	0.927	0.9288	0.9033	0.9456	0.0743
	QDA	0.9247	0.9288	0.8985	0.9454	0.0784
	逻辑回归	0.9278	0.9147	0.9164	0.9363	0.0624
因子分析	LDA	0.9278	0.9276	0.9059	0.9449	0.072
	QDA	0.926	0.9313	0.8993	0.9472	0.078
	逻辑回归	0.9278	0.9153	0.9159	0.9367	0.0628
tSNE	LDA	0.9136	0.9215	0.882	0.9393	0.0922
	QDA	0.9155	0.9153	0.8902	0.9353	0.0844

逻辑回归	0.9105	0.8822	0.9061	0.9136	0.0683

4.3.4.1 Accuracy + Precision + Recall 评价组合

在 Accuracy + Precision + Recall 的评价组合下，上述方法均表现良好，除了 QDA 判别和建立在 tSNE 上的判别分析的召回率小于 90%外，其余方法准确率、精确度、召回率几乎都大于等于 90%。

如果一定要选出一个最佳方法，首先排除准确率、精确度、召回率中任一指标小于 90%的方法，例如建立在 tSNE 降维数据上的全部方法，建立在主成分降维数据上的全部 QDA 判别。

其次，逐一考虑指标：

若以准确率(Accuracy)为主要指标，则最佳方法为建立在 3 个主成分上的逻辑回归、建立在因子分析上的 LDA 判别和建立在因子分析上的逻辑回归。

若以精确度(Precision)为主要指标，则最佳方法为建立在 3 个主成分上的 LDA 判别。

若以召回率(Recall)为主要指标，则最佳方法为建立在 2 个主成分上的逻辑回归。

总的来说，降维数据为 3 个主成分时要优于其他降维，逻辑回归要优于判别分析。

4.3.4.2 TPR + TNR + FPR 评价组合

在 TPR + TNR + FPR 的评价组合下，上述方法均表现良好，除了 QDA 判别和建立在 tSNE 上的判别分析的真阳性率小于 90%外，其余方法真阳性率、真阴性率几乎都大于等于 90%，假阳性率也能控制得很好。

如果一定要选出一个最佳方法，首先排除真阳性率、真阴性率中指标小于 90%和假阳性率过大的方法，例如 QDA 判别和建立在 tSNE 上的判别分析。

其次，逐一考虑指标：

若以真阳性率(TPR)为主要指标，则最佳方法为建立在 2 个主成分上的逻辑回归模型。

若以真阴性率(TNR)为主要指标，则最佳方法为建立在 3 个主成分上的 LDA 判别。

若以假阳性率(FPR)为主要指标，则最佳方法为建立在 2 个主成分上的逻辑回归模型。

总的来说，降维数据为 2 个主成分时要优于其他降维，逻辑回归模型要优于判别分析。

5. 总结

我们小组使用了 UCI 数据库的大米图像特征数据，利用主成分分析、因子分析和 tSNE 数据降维方法对数据进行降维，然后利用判别分析和逻辑回归分类方法对降维数据进行分类，希望能够从中得到一种有效的能够对大米进行分类的方法。

我们有以下结论：

- 主成分分析、因子分析和 tSNE 降维方法与判别分析（LDA、QDA 准则）、逻辑回归分类方法一共形成了 9 种方法，这 9 种方法均表现良好。
- 严格来说，在 Accuracy + Precision + Recall 的评价组合下，降维数据为 3 个主成分时要优于其他降维，逻辑回归要优于判别分析。
- 严格来说，在 TPR + TNR + FPR 的评价组合下，降维数据为 2 个主成分时要优于其他降维，逻辑回归要优于判别分析。
- 总的来说，对于大米图像特征数据而言，主成分降维要优于其他降维，逻辑回归要优于判别分析。
- 上述结果与 3.4.4 中降维数据可视化的结果相差甚大，这说明了可视化具有局限性，数据的具体分类性能还需要进行真正的分类实践。
- 最终，一种有效的能够对大米图像特征数据进行有效分类的方法是对特征数据先进行主成分数据降维，然后对降维数据建立逻辑回归的分类模型。

参考文献

[1] 国家统计局 https://data.stats.gov.cn/index.htm

[2] Cınar, I., & Koklu, M. (2019). Classification of Rice Varieties Using Artificial Intelligence Methods. International Journal of Intelligent Systems and Applications in Engineering.

[3] Rice (Cammeo and Osmancik). (2019). UCI Machine Learning Repository. https://doi.org/10.24432/C5MW4Z.

[4] 图像区域分析和常规特征提取① - Clock_926 的文章 - 知乎 https://zhuanlan.zhihu.com/p/625915055

[5] 多元统计分析/李高荣，吴密霞编著.—北京：科学出版社，2021.9（统计与数据科学丛书；4）

[6] Maaten, L.V., & Hinton, G.E. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research, 9, 2579-2605.

附 录

2.1 基本统计量(Python)

```python
from ucimlrepo import fetch_ucirepo
rice_cammeo_and_osmancik = fetch_ucirepo(id=545)
X = rice_cammeo_and_osmancik.data.features
y = rice_cammeo_and_osmancik.data.targets
data_stats = X.describe()
data_stats.loc['skew'] = X.skew()
data_stats.loc['kurt'] = X.kurt()
print(data_stats)
```

2.2 相关性分析(R)

```r
library(openxlsx)
library(GGally)
# 读取数据
data = read.xlsx("Rice_Cammeo_Osmancik.xlsx")
# 相关系数可视化
ggpairs(data = data, columns = 1:7)
```

3.1 主成分分析(R)

```r
library(openxlsx)
library(ggplot2)

# 读取数据
data = read.xlsx("Rice_Cammeo_Osmancik.xlsx")

data.pca = princomp(data[1:7], cor = TRUE)

summary(data.pca, loadings = TRUE)

screeplot(data.pca, type="lines", main="Scree Plot")

score = predict(data.pca)

data$comp.1 = score[,1]
data$comp.2 = score[,2]
data$comp.3 = score[,3]

ggplot(data, aes(x = comp.1, y = comp.2, color = Class)) +
```

```
  geom_point(size = 3) +  # 调整点的大小
  labs(title = "PCA",
       x = "dimension.1",
       y = "dimension.2",
       color = "Label") +  # 设置图例标题
  scale_color_manual(values = c("Cammeo" = "red", "Osmancik" = "blue")) +
  theme_minimal()  # 使用简洁的主题

library(openxlsx)
write.xlsx(data[8:11], "pca_data.xlsx")
```

3.2 因子分析(R)

```
library(openxlsx)
library(ggplot2)

# 读取数据
data = read.xlsx("Rice_Cammeo_Osmancik.xlsx")

factanal(data[1:7], factors = 2, rotation = "varimax")

data.Thomson = factanal(data[1:7], factors = 2, scores = "regression")$score
s
data$Factor1 = data.Thomson[,1]
data$Factor2 = data.Thomson[,2]


ggplot(data, aes(x = Factor1, y = Factor2, color = Class)) +
  geom_point(size = 3) +  # 调整点的大小
  labs(title = "factanal",
       x = "Factor.1",
       y = "Factor.2",
       color = "Label") +  # 设置图例标题
  scale_color_manual(values = c("Cammeo" = "red", "Osmancik" = "blue")) +
  theme_minimal()  # 使用简洁的主题

write.xlsx(data[8:10], "factanal_data.xlsx")
```

3.3 tSNE(Python: ipynb)

```
from ucimlrepo import fetch_ucirepo
rice_cammeo_and_osmancik = fetch_ucirepo(id=545)
X = rice_cammeo_and_osmancik.data.features
```

```
y = rice_cammeo_and_osmancik.data.targets


import pandas as pd
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler

def data_scale_copy(X):
    '''''返回数据 X 的标准化'''
    data_copy = X.copy()
    scaler = StandardScaler()
    scaled_data = scaler.fit_transform(data_copy)
    data_copy = pd.DataFrame(scaled_data, columns=X.columns)
    return data_copy

data_copy = data_scale_copy(X)
tsne = TSNE(n_components=2, perplexity=10, random_state=210810122)
tsne.fit_transform(data_copy)
tsne_data = pd.DataFrame(tsne.embedding_, columns=['dimension1', 'dimension2
'])
tsne_data['label'] = y['Class']


tsne_data.to_excel('tsne_data.xlsx', index=False)


from matplotlib import pyplot as plt

colors = {'Cammeo':'red', 'Osmancik':'blue'}

plt.figure(figsize=(10, 6))
for label in tsne_data['label'].unique():
    subset = tsne_data[tsne_data['label'] == label]
    plt.scatter(subset['dimension1'], subset['dimension2'], c=colors[label],
 label=label)

plt.legend(title='Label')
plt.xlabel('dimension.1')
plt.ylabel('dimension.1')
plt.title('tSNE')
plt.show()
```

4.1 判别分析

```
library(openxlsx)
pca_data = read.xlsx("pca_data.xlsx")
tsne_data = read.xlsx("tsne_data.xlsx")
factanal_data = read.xlsx("factanal_data.xlsx")

library(MASS)
# 2 个主成分的 LDA 判别
lda.2comp.fit = lda(Class~comp.1+comp.2, data = pca_data, prior = c(1,1)/2)

lda.2comp.fit.predict = predict(lda.2comp.fit, newdata = pca_data)
# print("2 个主成分的 LDA 判别")
table(pca_data$Class, lda.2comp.fit.predict$class)

# 2 个主成分的 QDA 判别
qda.2comp.fit = qda(Class~comp.1+comp.2, data = pca_data, prior = c(1,1)/2)

qda.2comp.fit.predict = predict(qda.2comp.fit, newdata=pca_data)
# print("2 个主成分的 QDA 判别")
table(pca_data$Class, qda.2comp.fit.predict$class)

# 3 个主成分的 LDA 判别
lda.3comp.fit = lda(Class~comp.1+comp.2+comp.3, data = pca_data, prior = c(1
,1)/2)
lda.3comp.fit.predict = predict(lda.3comp.fit, newdata = pca_data)
# print("3 个主成分的 LDA 判别")
table(pca_data$Class, lda.3comp.fit.predict$class)

# 3 个主成分的 QDA 判别
qda.3comp.fit = qda(Class~comp.1+comp.2+comp.3, data = pca_data, prior = c(1
,1)/2)
qda.3comp.fit.predict = predict(qda.3comp.fit, newdata = pca_data)
# print("3 个主成分的 QDA 判别")
table(pca_data$Class, qda.3comp.fit.predict$class)

# 因子分析的 LDA 判别
lda.factor.fit = lda(Class~Factor1+Factor2, data = factanal_data, prior = c(
1,1)/2)
lda.factor.fit.predict = predict(lda.factor.fit, newdata = factanal_data)
# print("因子分析的 LDA 判别")
table(factanal_data$Class, lda.factor.fit.predict$class)

# 因子分析的 QDA 判别
qda.factor.fit = qda(Class~Factor1+Factor2, data = factanal_data, prior = c(
1,1)/2)
```

```
    qda.factor.fit.predict = predict(qda.factor.fit, newdata = factanal_data)
    # print("因子分析的 QDA 判别")
    table(factanal_data$Class, qda.factor.fit.predict$class)

    # tSNE 的 LDA 判别
    lda.tSNE.fit = lda(label~dimension1+dimension2, data = tsne_data, prior = c(1,1)/2)
    lda.tSNE.fit.predict = predict(lda.tSNE.fit, newdata = tsne_data)
    # print("tSNE 的 LDA 判别")
    table(tsne_data$label, lda.tSNE.fit.predict$class)

    # tSNE 的 QDA 判别
    qda.tSNE.fit = qda(label~dimension1+dimension2, data = tsne_data, prior = c(1,1)/2)
    qda.tSNE.fit.predict = predict(qda.tSNE.fit, newdata = tsne_data)
    # print("tSNE 的 QDA 判别")
    table(tsne_data$label, qda.tSNE.fit.predict$class)
```

4.2 逻辑回归

```
    library(openxlsx)
    pca_data = read.xlsx("pca_data.xlsx")
    tsne_data = read.xlsx("tsne_data.xlsx")
    factanal_data = read.xlsx("factanal_data.xlsx")

    class_mapping = c("Cammeo"=0, "Osmancik"=1)
    pca_data$Class = class_mapping[pca_data$Class]
    tsne_data$label = class_mapping[tsne_data$label]
    factanal_data$Class = class_mapping[factanal_data$Class]

    # 2 个主成分的逻辑回归
    logit.2comp = glm(Class~comp.1+comp.2, data = pca_data, family = "binomial")

    logit.2comp.predict = predict(logit.2comp, type = "response")
    logit.2comp.predict.class = ifelse(logit.2comp.predict < 0.5, 0, 1)
    table(pca_data$Class, logit.2comp.predict.class)

    # 3 个主成分的逻辑回归
    logit.3comp = glm(Class~comp.1+comp.2+comp.3, data = pca_data, family = "binomial")
    logit.3comp.predict = predict(logit.3comp, type = "response")
    logit.3comp.predict.class = ifelse(logit.3comp.predict < 0.5, 0, 1)
    table(pca_data$Class, logit.3comp.predict.class)

```

```
    # 因子分析的逻辑回归
    logit.factor = glm(Class~Factor1+Factor2, data = factanal_data, family = "binomial")
    logit.factor.predict = predict(logit.factor, type = "response")
    logit.factor.predict.class = ifelse(logit.factor.predict < 0.5, 0, 1)
    table(factanal_data$Class, logit.factor.predict.class)

    # tSNE 的逻辑回归
    logit.tSNE = glm(label~dimension1+dimension2, data = tsne_data, family = "binomial")
    logit.tSNE.predict = predict(logit.tSNE, type = "response")
    logit.tSNE.predict.class = ifelse(logit.tSNE.predict < 0.5, 0, 1)
    table(tsne_data$label, logit.tSNE.predict.class)
```