单位代码：11414
学　　号：2016011660

中国石油大学（北京）
CHINA UNIVERSITY OF PETROLEUM , BEIJING

# 本科生毕业设计（论文）

| | |
|---|---|
| 题　　目 | 随机森林算法及其在回归与分类中的应用研究 |
| 学院名称 | 理学院 |
| 专业名称 | 数学与应用数学 |
| 学生姓名 | 张逢源 |
| 指导教师 | 武国宁教授　副教授 |

起止时间：　2020 年 2 月 1 日 至 2020 年 5 月 23 日

# 摘　要

随机森林是一种监督学习算法。它可以用于分类和回归。它也是最灵活和易于使用的算法。森林由树木组成。据说树木越多，森林就越健壮。随机森林在随机选择的数据样本上创建决策树，从每个树中获取预测，并通过投票选择最佳解决方案。它还提供了功能重要性的很好指示。随机森林具有各种应用程序，例如推荐引擎，图像分类和特征选择。它可用于对忠实的贷款申请人进行分类，识别欺诈活动并预测疾病。创建随机森林模型主要有以下几步：从给定的数据集中选择随机样本。为每个样本构造一个决策树，并从每个决策树中获得预测结果。对每个预测结果进行投票。选择投票最多的预测结果作为最终预测结果。由于参与该过程的决策树数量众多，因此随机森林被认为是一种高度准确且健壮的方法。我们利用随机森林算法，它不会遭受过度拟合的问题。主要原因是它取所有预测的平均值，从而消除了偏差影响。随机森林也可以处理缺失值。处理这些问题的方法有两种：使用中位数替换连续变量，以及计算缺失值的近似加权平均值。

**关键词：决策树；集成学习；随机森林；回归；分类**

# Random forest algorithm and its application in regression and classification

## ABSTRACT

Random forest is a supervised learning algorithm. It can be used for classification and regression. It is also the most flexible and easy to use algorithm. A forest is made up of trees. It is said that the more trees there are, the stronger the forest. A random forest creates decision trees on randomly selected data samples, retrieves predictions from each tree, and votes to select the best solution. It also provides a good indication of functional importance. Random forest has a variety of applications, such as recommendation engines, image classification, and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity and predict disease. There are several steps to creating a random forest model: select a random sample from a given data set. Construct a decision tree for each sample and obtain the predicted results from each decision tree. Vote on each prediction. Select the forecast result with the most votes as the final forecast result. Because of the large number of decision trees involved in the process, random forest is considered a highly accurate and robust method. We use the random forest algorithm, which does not suffer from overfitting problems. The main reason is that it takes the average of all the predictions, thus eliminating the effects of bias. Random forests can also handle missing values. There are two ways of dealing with these problems: replacing continuous variables with medians, and calculating an approximate weighted average of missing values.

**Key words: decision tree; Integrated learning; Random forest; Regression; classification**

# 目　录

# 第 1 章　绪论

## 1.1 研究目的和意义

机器学习（Machine Learning）是人工智能（AI）研究和实践的一个领域，它使用计算机学习经验，以此改善他们的认知和行动。机器学习可以从数据和经验中不断改进，以此智能和快捷的处理大量学科中的复杂信息，其宗旨就是利用计算机处理量化问题的高效性，通过多种方式对现象进行估计，为决策提供建议以及指导和纠正。

在处理分类与回归分析中，由于计算机在处理量化数据上有很大的优势，机器学习能够分析大量的数据，发现人类无法发现的特定趋势和模式。并且在机器学习中，我们不需要关注项目中的每一步骤，由于我们赋予机器学习自我学习的能力，它可以自动处理新数据以及根据新数据来改进自身算法。随着处理数据的积累，其精度和效率可以不断的提高，这能够使得机器学习作出更好的判断。随机森林算法作为机器学习的一种，且由于其集成学习的思想，它能够在众多分类器生成的结果之中进行投票，选择最优的结果。并且随机森林算法可以降低单个分类器造成的过拟合风险，从而达到高精度的分类与回归结果。本文的主要目的就是研究随机森林算法的基本原理，适用条件，以及如何利用该算法处理实际分类与回归问题。

## 1.2 国内外研究现状

1996 年，Leo Breiman 提出了 bagging[1]思想，即一种生成多个分类器并对每个分类器得出的结果进行投票，得出最终结果的方法。2001 年，Leo Breiman 在 bagging 思想的基础上，将之与随机子空间方法结合提出了随机森林算法。在 20 年里，众多科学家和学者使用随机森林算法解决了各个领域的分类与回归问题，并且国内外对随机森林算法进行了进一步的改进与研究。

### 1.2.1 随机森林算法在各学科领域中的应用

在环境、生态学领域中，彭潮[3]等人利用随机森林算法的高精度等优点，对煤粉的组成成分进行了回归分析，并预测了不同煤粉的组成对着火温度的影响。胡梦珺[4]等人对随机森林算法中决策树的个数以及每个树节点的变量个数进行优化，通过对 2000 多个特征参数与 500 余训练结果组成的训练集组成的训练样本，分析了兰州市重金属污染的地区特征与时间特征。陈军飞[5]等人利用随机森林算法，对导致洪水发生的各类因素进行分析，用环境和时间因素对洪水发生风

险度建立了完整的评估模型，并且将随机森林算法与其他机器学习算法进行精确度比较，得出随机森林算法精确度更高的结果。杭琦[6]在对决策树特征选择方法进行优化后，利用改进的随机森林算法，对各种影响城市空气质量因素进行分类与回归分析，提出了一种新的评价与预测地区空气质量水平的方法。包青岭[7]等人在新疆地区进行土壤采样，并对土壤进行光谱分析，用对地区已有有机质含量数据进行训练后，建立了一种利用随机森林算法对地区土壤有机质含量的检测模型，大大加快了土壤检测的精确度与速度。潘登[8]等人选取可燃物湿度等因素，对地区林火预测建立了随机算法模型，预测了各因素对火灾发生的影响，并且与其他机器学习算法模型进行比较，得出了一种高精度的预测模型。

在医学领域中，孙凯[9]提出了一种对医学成像进行分析，利用随机森林算法检测图像，判断患者肿瘤位置。由于该算法的高精准度，医生能够获取精准的判断，进行诊断治疗。杨美洁[10]等人将人年龄，性别，胰岛素含量等因素作为特征向量，利用随机森林算法，根据此些因素，对受试者是否患糖尿病建立了预测模型。张英男[11]等人将吸烟情况，饮酒情况，是否进行体育锻炼，食用蔬菜水果习惯等因素作为特征变量，利用随机森林算法建立老年痴呆症的预测模型，很好的计算出特征因素的重要性排序。王平[12]等人将正负样本数据分开处理，讨论随机森林不同决策树之间的相关系数，缩小泛化误差，并利用改进的算法对乳腺癌诊断建立预测模型，方便医生临床诊断。

在金融领域，郭建山[13]等人通过分析年龄，性别，还款情况的 20 余种特征变量，建立基于随机森林算法的信用卡还款违约预测模型，并且与 KNN 等算法比较，发现随机森林算法具有更高的精确性。马晓君[14]等人对决策树权重进行修正，计算最优决策树数量，很好的避免了过拟合的情况，并利用优化的随机森林算法对上市公司信用等级建立了评级模型。朱青[15]通过对机构持股比例，基金经理能力能十余种特征变量进行分析，利用随机森林算法对开放式基金进行评级，并与 SVM 等其余机器学习方法进行比较，得出了一种准确度更高的基金评级模型。张潇[16]等人将市盈率，市净率等 8 项特征变量作为判断股票选取的技术指标，将过去三年内十几只股票进行训练，建造了一种高精确度的股票预测模型。

### 1.2.2 随机森林算法的改进

Jerome H. Friedman[17]在 2001 年提出了一种从函数空间的数值优化角度出发，而不是从参数空间出发，在逐级加性扩展和急剧下降最小化之间建立了联系。在任何拟合准则的基础上，提出了一种广义梯度提升随机森林算法（Gradient boosted decision trees, GBDT），使用梯度下降方法，寻找最优迭代次数即数量最优的决策树，该算法能够很好的避免过拟合的情况。随机森林将多棵

决策树的结果进行投票后得到最终的结果，对不同的树的训练结果也没有做进一步的优化提升，将其称为 bagging 算法。boosting 算法是在迭代的每一步构建弱学习器来弥补原有模型的不足。梯度提升随机森林算法用到的是 boosting 算法，在迭代的每一步构建弱学习器弥补原有模型的不足。梯度提升随机森林算法的提升梯度就是通过每次迭代的时候构建一个沿梯度下降最快的方向的学习器。并且通过设置不同的损失函数可以处理各类学习任务（多分类、回归等）。

GEURTSP[18]等人针对监督分类和回归问题，于 2006 年提出了一种新的基于树的集成方法，称为极端随机森林（Extremely randomized trees）。它本质上是在对树节点进行分割时，对属性和切割点的选择进行强随机化。在极端情况下，它构建完全随机的决策树，其结构独立于学习样本的输出值。在高度随机化的决策树中，随机性在计算分割的方式上进一步向前迈进了一步。像在随机森林中一样，使用候选特征的随机子集，但不是寻找最有区别的阈值，而是为每个候选特征随机绘制阈值，并选择这些随机生成的阈值中的最佳阈值作为划分规则。这通常可以更大程度地减少模型的方差，但要以更大的偏差增加为代价

Quadrianto Novi[19]在 2015 年提出了一种贝叶斯随机森林（Safe-Bayesian Random Forest），这种算法从先验分布中随机抽样许多树，然后执行预测概率的加权集合。使用先验分布，允许在查看数据之前对决策树进行采样，并使用幂次可能性来探索决策树的组合所跨越的空间。贝叶斯随机森林的安全性来自于它具有良好的预测性能，即使底层的概率模型是错误的。贝叶斯随机森林在速度和精度上都优于基于 MCMC 或 SMC 的贝叶斯决策树，并且在与熵或基尼系数优化的随机森林的竞争中表现优异，而且构造起来却非常简单。

## 1.3 主要研究内容

我们在对事物作出判别或者预测中，由于数据的量是庞大的，往往无法作出科学的预测。随机森林算法作为机器学习的一种，能够处理大量的数据，并且进行分类和预测。本文的主要研究内容是如何运用随机森林算法在 python 上处理数据，并且能够利用已有数据，判断特征变量对事物发展的影响权重，以及能够对事物的未来发展作出良好预测，以及给出预测和分类的准确度计算，将之与其他机器学习算法进行比较，总结该算法的优缺点以及适用性。

# 第 2 章　随机森林基本原理

## 2.1 机器学习

### 2.1.1 样本

样本即是待解决问题中数据的特定实例，在各个场景中，过程数据以及产生的结果就成为一个样本。样本分为特征与标签。描述样本的信息被称为特征变量，在机器学习中，特征能够让算法模型理解并描述过程数据，通过特征输入让算法模型理解该样本特征与结果之间的关系。标签即是算法模型需要去预测的结果。无监督学习没有标签。有监督学习中，样本标签作为反馈的结果，再可以分为判别分析与回归分析。

样本集合分为训练集，验证集与测试集。由于算法模型的目的是让其学习数据的规律来进行分析学习，进而对未来作出判断，因此训练集就是让模型学习规律的样本集合。验证集的作用是为了证明该算法的拟合程度，利用算法未进行学习过的样本集合来观察效果，从而进一步对算法进行改进。虽然模型没有学习验证集的数据，但是当我们依靠模型在验证集中的效果来判断模型是否准确时，该模型在训练集与验证集中就是一个持续迭代的过程，随着不断的修改，该模型会逐渐拟合验证集的数据，此时我们需要测试集来测试该模型的精确性。测试集是模型从来没有分析的数据，因此我们用测试集测试模型的准确度时，其反馈的结果会更加客观准确。

### 2.1.2 拟合

模型学习数据的过程称为拟合。当我们拟合数据时，会出现欠拟合与过拟合的情况。欠拟合指的是模型过于简单，不能充分的解释和描述数据的规律，对数据的预测有很大的偏差。过拟合是指模型过于复杂，使得模型对训练集所有数据都能够精确的表达，使得该模型能够符合每一个数据，过拟合会把偶然值当成规律性学习，因此造成精确度低的结果。

### 2.1.3 偏差与方差

偏差是指模型预测的结果与真实结果的差距，当偏差越来越小的情况下，模型会向过拟合发展，此时方差就会越来越大，导致模型的泛化能力很弱，模型不稳定，不能适用于多种样本集合。当偏差很大是，会造成模型欠拟合，此种情况下该模型不能做到预测的功能，此时方差会很小。一个优秀的机器学习模型需要

同时降低偏差与方差，此时模型既能对结果进行精准的预测，又具有很强的泛化能力，在不同场景中都有精确的判断。

### 2.1.4 损失函数

当机器进行学习时，我们定义一种函数关系，叫做损失函数：

$$J(\omega) = \sum_i L\big(m_i(\omega)\big) + \lambda R(\omega) \quad (2.1)$$

在每一次学习过程中，我们将模型结果和真实结果$R(\omega)$比较，得到差值$L\big(m_i(\omega)\big)$，计算机会不断修改以缩小此差值，通过降低损失函数达到学习的目的。

损失函数分为$L$和$R$两部分。$L$是误差项，表达了模型预测结果与真实结果的偏差程度。$R$代表了模型复杂程度，表达了模型预测的泛化能力。我们在修改模型时，需要调整$L$与$R$的权重$\lambda$，以达到最小的$J$，使得模型达到最优。

## 2.2 决策树

决策树是监督学习的一种重要算法，是基于树结构的分类算法。1986 年，J.Ross Quinlan 提出了 ID3[20]决策树算法，在此基础上，1993 年 Quinlan 又提出了 C4.5[21]决策树算法，Leo Breiman 等人又提出了 CART 算法。决策树算法能够处理离散型数据，从而提取数据中蕴含的规律，是机器学习的重要组成。决策树是一种类似于流程图的树结构，如图 2.1 所示。

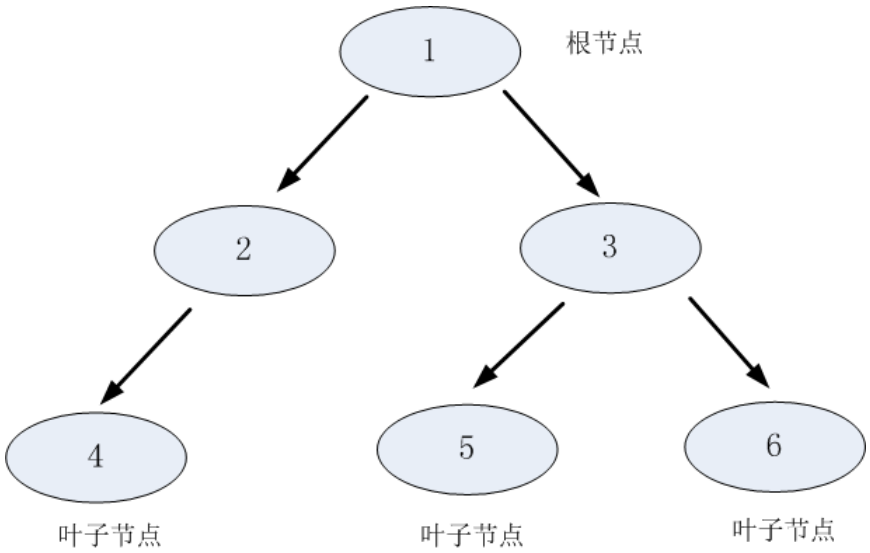


图 2.1 决策树流程图

Fig.2.1 Decision tree flow chart

决策树中包含了多个内部节点，每个内部结点表示在一个属性上的测试，每个分支表示一个属性输出，而每个树叶结点代表类或者类的分布。树的最顶层是根结点。

当决策树运行时，先判定根结点属性，根据不同的取值情况进行下一分支，接着依次进行分支。当前所有两本都属于同一个目标函数的标记时，停止分支，最终所有数据都落在叶子结点上，既可以做分类也可以做回归。

每一个结点相当于相当于将数据集划分，结点越多，数据集就会划分的更细。

从根节点开始，我们需要考虑分支如何确定，特征向量如何划分。对于如何确定划分样本集合的顺序的特征变量，我们需要引入信息和熵的概念。

### 2.2.1 信息熵

假设随机变量$X$有 m 个值，分别为：$V_1, V_2, V_3, \ldots, V_m$; 并且每个值出现的概率分别为： $P_1, P_2, P_3, \ldots, P_m$;那么可以使用下面公式来表示每个变量的信息值。设 X 是一个取有限个值的离散随机变量，其概率分布为：

$$P(X = x_i) = p_i, \quad i = 1,2, \ldots, n \tag{2.2}$$

则随机变量 X 的信息熵的定义为：

$$H(X) = -p_1 \log_2(p_1) - p_2 \log_2(p_2) - \cdots - p_m \log_2(p_m) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{2.3}$$

一个样本或者时间具有的信息被称为信息量，由上式可知，确定性越高的事件概率越大，所蕴含的信息越少；而确定性低的事件概率越小，所蕴含的信息量就越小。从而引入信息熵的概念：当一个事件信息量越小时，整个系统就会变得有序，此时信息熵就越小；当事件的信息量越大时，系统就会变得混乱，此时信息熵越大。信息熵就是一种系统有序程度的度量。

在机器学习中，信息熵表示各随机变量取值的可能性程度大小。高信息熵表示各随机变量不确定程度很高，各种取值情况出现的概率差别不大。低信息熵表示随机变量有序，各事件出现的概率有很大差异。

### 2.2.3 条件熵

设有随机变量（$X, Y$）,其联合概率分布为：

$$P(X = x_i, Y = y_i) = p_{ij}, \qquad i = 1,2,3, \ldots, n, j = 1,2, \ldots, m。 \tag{2.4}$$

条件熵$H(Y|X)$表示在已知随机变量$X$的条件下，随机变量 Y 的不确定性。随机变量 X 给定的条件下随机变量$Y$的条件熵$H(Y|X)$，定义为$X$给定条件下,所有不同$x$的值情况下$Y$的信息熵的平均值叫做条件熵：

$$H(Y|X) = \sum_{i=1}^{n} p_i H(Y|X = x_i) \quad (2.5)$$

其中:

$$p_i = P(X = x_i), \quad i = 1,2, \dots, n$$

另外，事件（$X,Y$）发生所包含的熵，减去事件$X$单独发生的熵，也就是事件 X 发生的前提下，$Y$发生的熵。即条件熵本身的概念，我们可得到以下公式

$$H(Y|X) = H(X,Y) - H(X) \quad (2.6)$$

证明过程如下：

$$\begin{aligned} H(Y|X) &= \sum_{i=1}^{n} p_i H(Y|X = x_i) = \sum_{x} P(x) H(Y|X) \\ &= \sum_{x} p(x) \left\{ -\sum_{y} p(y|x) \log\big(p(y|x)\big) \right\} \\ &= -\sum_{x}\sum_{y} p(X) p(y|x) \log\big(p(y|x)\big) \\ &= \sum_{x}\sum_{y} p(x,y) \log\left(\frac{p(x,y)}{p(x)}\right) \\ &= \sum_{x}\sum_{y} p(x,y) \log\big(p(x,y)\big) - \left[ -\sum_{x} \left( \sum_{y} p(x,y) \right) \log\big(p(x)\big) \right] \\ &= H(X,Y) - \left[ -\sum_{x} p(x) \log\big(p(x)\big) \right] = H(X,Y) - H(X) \end{aligned}$$

### 2.2.4 决策树算法

构造决策树的关键点是如何选择实行判断结点，不同的决策树建造过程需要选择不同结点，我们需要计算如何划分结点来达到最优的分类结果。当前主流的判断结点的方法有三种，分别是 ID3、C4.5、CART。这几种算法都是贪心算法，即自上而下构造决策树。然而属性选择的度量方式不同

（1）ID3 算法

该算法创建多路树，为每个节点（即，以贪婪的方式）找到分类特征，该分类特征将为分类目标产生最大的信息获取量。将树长到最大大小，然后通常应用

修剪步骤以提高树概括未知数据的能力。ID3 算法是依靠信息获取量（Information Gain）来判定结点的属性划分,有如下公式：

$$Gain(A) = info(D) - infor\_A(D)$$ 这个公式再核实一遍

$Gain(A)$表示 A 的信息获取量即是没有 A 的信息时的信息熵和得知 A 的信息之后的条件熵的差。之后，比较每个特征变量的信息获取量大小，选取具有最大的信息获取量的特征向量作为优先划分的结点。

但是，ID3 算法会倾向于取值比较多的样本，因此得出结果不一定最优。而且智能处理离散数据，因此出现了 C4.5 决策树算法。

（2）C4.5 算法

C4.5 是 ID3 的后继版本，它通过动态定义离散属性（基于数字变量）来消除要素必须分类的限制，该离散属性将连续属性值划分为离散的间隔集，不使用信息增益作为结点划分标准，而使用信息增益率评判。增益率即使将输出结点的个数也纳入考量。C4.5 将训练后的树（即 ID3 算法的输出）转换为 if-then 规则集。然后评估每个规则的这些准确性，以确定应该应用它们的顺序。有以下公式：

$$GainRatio(X) = \frac{Gain(X)}{Splitinformation(X)} \tag{2.7}$$

其中，分裂信息度为

$$Splitinformation(X) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{2.8}$$

（3）CART 算法

CART（分类树和回归树）与 C4.5 非常相似，但是区别在于它支持数字目标变量（回归）并且不计算规则集，仅仅构造二叉决策树。CART 使用在每个节点处产生最大信息增益的特征和阈值构造二叉树。

在 CART 决策树算法中，与另外两种算法不同的是，它通过基尼系数大小来判断，有以下公式：

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk}) \tag{2.9}$$

### 2.2.5 决策树算法构造流程

（1）决策树以代表训练集的单个结点开始运行。

（2）所有属性都是分类处理，连续性特征变量必须离散化处理才能作为决策树的划分结点。

（3）如果样本都在同一个类别，把该结点直接转变为叶子结点，并用该类标号。

（4）算法基于熵的度量作为启发信息，选择能够最好的将样本分类的属性作为当前划分结点。该属性称为该结点的测试或判定属性。

（5）对测试属性的每个已知值，创建不同的分支，并借此划分样本。

（6）算法使用相同的过程，递归地形成每个划分熵的样本判定树，一旦一个属性出现在一个结点上，就不必该结点下的任何结点上考虑。

（7）分类过程仅当下列条件之一决策树停止运行：

①给定结点的所有样本属于同一类别，停止运行，不需要继续选择属性划分。

②没有剩余特征变量能够用来进一步划分样本集合，所有的特征变量已经划分为结点。在这种条件下，使用少数服从多数策略，将结果归类为占大多数情况的类别。此种情况下依然有特征变量没有划分为结点， 将给定的结点转变成树叶，并将样本中的多数所在的类别标记它。替换成可以存放结点的样本的类的分布。

③分枝：没有样本的情况下，以样本中的多数类别创建一个树叶。

### 2.2.6 剪枝

当决策树划分了过多结点时，树的深度就会越大，特征数非常多此时分类情况会导致过于细化，造成过拟合情况的发生。每一个数据样本都落到对应的叶子结点上，每一个叶子结点都会无休止的继续划分，以至于划分过于复杂多样，每一个叶子结点过于纯净。因此理论上决策树能够把样本完全分开，此时决策树会非常贴合训练集，而在其他样本集合中不能很好的反应特征规律，此时我们用剪枝来避免过拟合。

（1）先剪枝

先剪枝即使在建造决策树过程当中进行剪枝。我们在建造决策树的时候，应先考虑我们的特征变量数目是否过大，每一个特征变量是否应该都作为结点划分。我们可以通过限制参数的方法来控制特征变量选取的多少，从而避免决策树过深。另外我们需要控制叶子结点里的样本数量，以控制决策树的宽度

①限制深度：限制特征树数量，控制决策树长度，停止特征变量继续划分。

②叶子结点个数和叶子结点样本数：通过控制叶子结点个数来避免样本落在每个结点的数目过小而导致的过于复杂。使得每个叶子结点上的样本个数不会太少，从而达到泛化能力更高的决策树模型。通过直接控制每个叶子结点的样本数量也能够也能够限制决策树的分裂从而控制模型复杂程度。

③信息增益量：通过评判没个特征所带来的信息增益量，使获得信息增益量过小的特征变量停止继续划分为一个结点，从而避免了不重要的特征划分来控制决策树的复杂程度。

（2）后剪枝

后剪枝即使在决策树构造完成之后再进行剪枝。我们可以通过以下公式来进行：

$$C_\alpha(T) = C(T) + \alpha \cdot |T_{leaf}| \tag{2.10}$$

其中，C(T)表示损失量。对于每一个叶子结点样本数乘信息熵值或者基尼系数，与它之后分裂的所有叶子结点的样本数与信息熵或者基尼系数想乘然后累加在一起相比较，就可以得到当前结点的损失值。因此来判断当前结点是否需要剪枝。$T_{leaf}$指的是叶子结点的个数。

### 2.2.7 决策树优点

（1）易于理解和解释。树木可以可视化，通过 python 绘图代码可以清晰的理解决策树每一步的分裂以及每一步的参数。

（2）需要很少的数据准备。其他机器学习模型通常需要数据标准化化，提前创建伪变量并删除空白值。

（3）使用树的成本（即预测数据）与用于训练树的数据点数量成对数，能够帮助我们更好的处理回归与分类问题。

（4）能够处理回归和分类数据。其他分类器通常专用于分析仅具有一种类型的变量的数据集。

（5）能够处理多输出问题。

（6）使用白盒模型。如果模型中可以观察到给定的情况，则可以通过布尔逻辑轻松解释条件。相反，在黑匣子模型中（例如，在人工神经网络中），并不能很好的解释数据。

（7）可以使用统计测试来验证模型。这使得考虑模型的可靠性成为可能，从而验证决策树算法的精确度。

（8）即使生成数据的真实模型在某种程度上违背了它的分析结果，也可以很好的处理各种样本。

### 2.2.8 决策树的缺点

（1）决策树学习者可能会创建过于复杂的树，从而无法很好地概括数据，造成过拟合现象。为避免此问题，必须使用诸如修剪，设置叶节点处所需的最小样本数或设置树的最大深度之类的限制条件。

（2）决策树可能不稳定，因为数据中的细微变化可能会导致生成完全不同的树。通过使用集成决策树学习可以缓解此问题，也就是随机森林算法。

（3）在最优化的几个方面，甚至对于简单的概念，学习最优决策树的问题都被认为是因不能用多项式算法而使问题无法解决的。因此，实用的决策树学习算法基于启发式算法（例如贪婪算法），其中在每个节点上做出局部最优决策。这样的算法不能保证返回全局最优决策树。可以通过在集成学习器中训练多棵树来缓解这种情况，也就是随机森林算法。在该学习器中，特征和样本将通过又放回随机抽样。

（4）有些概念很难进行学习，因为决策树无法量化表达它们，例如 XOR，奇偶校验或多路复用器问题。

（5）如果某些类别占主导地位，决策树学习者会创建有偏见的树。因此，需要在与决策树拟合之前平衡数据集。

## 2.3 集成学习

对于决策树的缺点，我们可以通过创建多种决策树来降低算法的错误率，这里需要引入集成学习的思想。

### 2.3.1 基本理念

通过组合各种弱分类器（如决策树）获得基于集成的系统。此类系统也称为多分类器系统，或称为集成学习系统。在几种情况下，使用基于集成的系统具有统计意义。但是，为了地认识到使用多个分类器系统的重要性，我们可以举几个简单的例子来证明这种集成学习思想在机器学习中的合理性：在我们的日常生活中，在做出决定之前，我们通常会询问一些专家来判断一些事物。例如，在同意医疗程序之前，我们通常会征询多位医生的意见，在购买某项物品（尤其是大件物品）之前，我们会先阅读多个用户的评论，我们通过检查他们的参考资料等来评估未来的使用效果。在每种情况下，我们作出决定之前，并不是依靠一个人或者数据来评判，而是通过不同的人做出的不同的选择。

### 2.3.2 集成学习的特点

（1）能够处理数据太多或太少的情况：当处理大量数据或缺少足够的数据时，集成学习系统能够起到很好的作用。当单个分类器很难处理训练数据量太大的情况时，可以将数据策略性地划分为较小的子集。然后，使用每个分开的样本集合来训练单独的分类器，并且可以使用适当的组合规则进行组合。在另一方面，对于数据样本太少的情况，进行自举可以使用利用不同的分类器来训练不同的分类 bootstrap 样本数据。

（2）能够将样本边界进行细致划分：对于单独的分类器而言，我们很难将样本空间进行平滑划分。实际上，将数据从不同类别中分离出来的决策边界太过复杂。从某种意义上说，分类系统遵循分而治之的方法，将数据空间划分为较小且易于学习的分区，其中每个分类程序仅学习一个较简单的分区。

（3）多元化：集成学习具有纠正某些单个分类器错误的能力，这完全取决于构成集成学习的分类器的多样性。如果所有分类器都提供相同的输出，就不可能纠正可能的错误。因此，集成系统中的各个分类器可能会对于不同的样本集合中出现过拟合与欠拟合的错误。但是，即使每个分类器产生不同的误差，这些分类器的策略性组合就可以减少总误差的大小。具体来说，集成系统需要分类器，其决策边界与它包含的决策器有一定的区别。分类器多样性可以通过多种方式实我们可以使用不同的训练数据集来训练各个分类器。此类数据集通常是通过从整个训练数据中随机抽取训练数据子集，利用重新采样技术（例如 bagging（请参见下文））获得的。为了确保各个边界有足够的差异，尽管使用的训练数据基本相似，但弱点或更多不稳定分类器被用作基本模型，因为即使对于训练参数的微小扰动，它们也可以生成足够不同的决策边界。

### 2.3.3 投票系统

集成学习主要依靠其中每一个弱分类器共同投票的结果来决定输出，当在这里我们让$d_{t,J}$的取值取决于分类器$t$的选择的$j$是 1 或 0。然后集成学习选择获得最高票数的$j$类。

（1）多数投票

$$\sum_{t=1}^{T} d_{t,J}(x) = max_{j=1,\ldots,C} \sum_{t=1}^{T} d_{t,j} \tag{2.11}$$

在分类器输出是独立的条件下，可以证明多数表决组合将始终导致足够数量的分类器的性能改进。如果针对一个两类问题总共有 T 个分类器，那么至少(T/2+1)分类器选择正确的类别。现在，假设每个分类器都有做出正确决定的概率

p。然后，集成学习做出正确决定的概率具有二项式分布，即选择 k>(T/2+1)中的正确分类器的概率是

$$P_{ens} = \sum_{k=(T/2)+1}^{T} \binom{T}{k} p^k (1-p)^{T-k} \quad (2.12)$$

当$T \to \infty, p > 0.5$时

$$P_{ens} \to 1$$

当$T \to \infty, p < 0.5$时

$$P_{ens} \to 0$$

（2）加权多数投票

$$\sum_{t=1}^{T} w_t d_{t,J}(x) = max_{j=1,\ldots,C} \sum_{t=1}^{T} w_t d_{t,j} \quad (2.13)$$

如果 T 分类器是类条件独立的，且精确度为 $p_1,\cdots,p_T$ 则加权多数投票规则的最优权值为

$$wt \propto \frac{p_t}{1-p_t} \quad (2.14)$$

## 2.4 随机森林算法

### 2.4.1Bagging 思想

Bagging 思想代表引导聚合，是最早，最直观，也是最简单的基于集成的算法之一，具有良好性能，它最早被 Breiman[1]于 1996 年提出，是随机森林算法的前身。Bagging 中分类器的多样性是通过使用训练数据的自举获得的。也就是说，从整个训练数据集中随机抽取不同的训练数据子集。每个训练数据子集用于训练同一类型的不同分类器。然后，通过对决策者进行简单的多数表决将各个分类器合并。对于任何给定的实例，大多数分类器选择的类是整体决策。由于训练数据集可能会基本重叠，因此可以使用其他措施来增加多样性，例如使用训练数.据的子集来训练每个分类器，或使用相对较弱的分类器（例如决策树桩）。
通过构造多个决策树，将这些决策树看作弱分类器，弱分类器之间的权重相同，最后进行多数投票。

### 2.4.2 随机森林思想

随机森林算法是在 bagging 思想上进行了进一步的改进，每棵树的规模，性质都不一样。随机森林算法就是将每棵树作为弱学习器，集成的一个强学习器。与 bagging 思想不同的是，随机森林中决策树对于原本分析的规模是随机化的，也就是决策树对于原数据不是遍历的，每一个决策树通过随机数从样本中随机抽取数据进行训练，根据每个决策树对应的样本自己进行学习。随机森林算法中决

策树训练样本时进行的是有放回抽样，这样可以保证决策树分析数据的公平性。最后的出的决策树不仅形状各异，而且集成具有很强的容错率，由这些决策树组成的随机森林能够泛化更多的随机性。

### 2.4.3 随机森林生成流程

（1）对于每个弱学习器，首先进行随机有放回抽样。

（2）对每个弱学习器进行决策树学习。

（3）将弱学习器得到的结果进行投票。

### 2.4.4 构造随机森林模型关键点

（1）n_estimators：森林中树木的数量。树木的数量越大越好，但是计算所需的时间越长。此外当树木数量超过关键值时，结果将不会进一步优化。

（2）max_features：分割结点时要考虑要素随机子集的大小。越低，方差越小，然而偏差也就回越大。减小特征选择个数，决策树树的相关性和分类性能也会随之降低；增大 m，两者也相应增大。所以关键问题是如何选择最优特征选择个数。对于回归问题，max_features=None（始终考虑所有特征，而不是随机子集，对于分类任务其特征数考虑 max_features="sqrt"大小的随机子集。

（3）控制决策树的大小：指定 max_depth=h，h 则将增长为特定深度的二叉树。这样的树将（最多）具有 2h 个叶节点和（2h – 1）个分裂节点。另外，可以通过参数指定叶子节点的数量来控制树的大小。在这种情况下，将使用最佳优先划分生长树木，将首先扩展具有最大信息增益的节点。

# 第 3 章　随机森林算法处理回归问题

## 3.1 数学原理

对于集成模型$y_i$对于给定的输入的$x_i$具有以下形式:

$$\hat{y_i} = F_M(x_i) = \sum_{m=1}^{M} h_m(x_i) \tag{3.1}$$

其中，$h_m$是代表了弱学习器的估计值。梯度树增强使用固定大小的决策树回归作为弱学习器。常数$M$对应于估计量参数。

与其他增强算法相似，回归使用贪婪的方式构建：

$$F_m(x) = F_{m-1}(x) + h_m(m) \tag{3.2}$$

其中，$h_m(m)$表示新添加的树，$F_{m-1}(x)$表示添加这颗树之前的集成模型，两者相加得到新的集成模型。

鉴于上一集成模型$F_{m-1}(x)$，最新的决策树需要适配最小的损失值:

$$h_m = arg\min_h L_m = arg\min_h \sum_{i=1}^{n} l\left(y_i,\ F_{m-1}(x_i)+h(x_i)\right) \tag{3.3}$$

其中$L_m$为损失参数。有以下几种方式可以计算：

（1）最小二乘法：由于其卓越的计算性能，因此是回归的自然选择。初始模型由目标值的平均值给出。

（2）最小绝对偏差：用于回归的稳健损失函数。初始模型由目标值的中位数给出。

（3）Huber：另一个结合最小二乘和最小绝对偏差的鲁棒损失函数。

（4）分位数：分位数回归的损失函数。使用指定的位数。此损失函数可用于创建预测间隔。

## 3.2 实例分析

### 3.2.1 波士顿房价的预测

我们使用决策树与随机森林两种模型进行预测美国波士顿市的房价。特征变量有人均犯罪率等 14 种。模型建立流程如下：

（1）导入数据，绘制特征向量矩阵，如图 3.1 所示。

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.00632 | 18.0 | 2.31 | 0.0 | 0.538 | 6.575 | 65.2 | 4.0900 | 1.0 | 296.0 | ... |
| 1 | 0.02731 | 0.0 | 7.07 | 0.0 | 0.469 | 6.421 | 78.9 | 4.9671 | 2.0 | 242.0 | ... |
| 2 | 0.02729 | 0.0 | 7.07 | 0.0 | 0.469 | 7.185 | 61.1 | 4.9671 | 2.0 | 242.0 | ... |
| 3 | 0.03237 | 0.0 | 2.18 | 0.0 | 0.458 | 6.998 | 45.8 | 6.0622 | 3.0 | 222.0 | ... |
| 4 | 0.06905 | 0.0 | 2.18 | 0.0 | 0.458 | 7.147 | 54.2 | 6.0622 | 3.0 | 222.0 | ... |

5 rows × 61 columns

图 3.1 波士顿房价特征变量

Fig. 3.1 Characteristic variables of housing price in Boston

经过数据的初步处理，我们得到了 5 行 16 列的特征向量矩阵。

（2）计算基础统计数据，对样本特征变量进一步处理，计算房价总和，平均数，标准差，最小值，等 8 项数据。如图 3.2 所示:

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | PTRATIO | B | LS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.00 |
| mean | 3.613524 | 11.363636 | 11.136779 | 0.069170 | 0.554695 | 6.284634 | 68.574901 | 3.795043 | 9.549407 | 408.237154 | 18.455534 | 356.674032 | 12.65 |
| std | 8.601545 | 23.322453 | 6.860353 | 0.253994 | 0.115878 | 0.702617 | 28.148861 | 2.105710 | 8.707259 | 168.537116 | 2.164946 | 91.294864 | 7.14 |
| min | 0.006320 | 0.000000 | 0.460000 | 0.000000 | 0.385000 | 3.561000 | 2.900000 | 1.129600 | 1.000000 | 187.000000 | 12.600000 | 0.320000 | 1.73 |
| 1% | 0.013610 | 0.000000 | 1.253500 | 0.000000 | 0.398000 | 4.524450 | 6.610000 | 1.206540 | 1.000000 | 188.000000 | 13.000000 | 6.730000 | 2.88 |
| 50% | 0.256510 | 0.000000 | 9.690000 | 0.000000 | 0.538000 | 6.208500 | 77.500000 | 3.207450 | 5.000000 | 330.000000 | 19.050000 | 391.440000 | 11.36 |
| 99% | 41.370330 | 90.000000 | 25.650000 | 1.000000 | 0.871000 | 8.335000 | 100.000000 | 9.222770 | 24.000000 | 666.000000 | 21.200000 | 396.900000 | 33.91 |
| max | 88.976200 | 100.000000 | 27.740000 | 1.000000 | 0.871000 | 8.780000 | 100.000000 | 12.126500 | 24.000000 | 711.000000 | 22.000000 | 396.900000 | 37.97 |

图 3.2 特征变量基础统计数据

Fig. 3.2 Feature variables basic statistics

（3）计算特征向量之间的相关性

计算每个特征变量之间的相关程度如图 3.3 所示。

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | PTRATIO | B | LSTAT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CRIM | 1.000000 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 0.625505 | 0.582764 | NaN | NaN | NaN |
| ZN | NaN | 1.000000 | -0.533828 | NaN | -0.516604 | NaN | -0.569537 | 0.664408 | NaN | NaN | NaN | NaN | NaN |
| INDUS | NaN | -0.533828 | 1.000000 | NaN | 0.763651 | NaN | 0.644779 | -0.708027 | 0.595129 | 0.720760 | NaN | NaN | 0.603800 |
| CHAS | NaN | NaN | NaN | 1.0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| NOX | NaN | -0.516604 | 0.763651 | NaN | 1.000000 | NaN | 0.731470 | -0.769230 | 0.611441 | 0.668023 | NaN | NaN | 0.590879 |
| RM | NaN | NaN | NaN | NaN | NaN | 1.000000 | NaN | NaN | NaN | NaN | NaN | NaN | -0.613808 |
| AGE | NaN | -0.569537 | 0.644779 | NaN | 0.731470 | NaN | 1.000000 | -0.747881 | NaN | 0.506456 | NaN | NaN | 0.602339 |
| DIS | NaN | 0.664408 | -0.708027 | NaN | -0.769230 | NaN | -0.747881 | 1.000000 | NaN | -0.534432 | NaN | NaN | NaN |
| RAD | 0.625505 | NaN | 0.595129 | NaN | 0.611441 | NaN | NaN | NaN | 1.000000 | 0.910228 | NaN | NaN | NaN |
| TAX | 0.582764 | NaN | 0.720760 | NaN | 0.668023 | NaN | 0.506456 | -0.534432 | 0.910228 | 1.000000 | NaN | NaN | 0.543993 |
| PTRATIO | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 1.0 | NaN | NaN |
| B | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 1.0 | NaN |
| LSTAT | NaN | NaN | 0.603800 | NaN | 0.590879 | -0.613808 | 0.602339 | NaN | NaN | 0.543993 | NaN | NaN | 1.000000 |

图 3.3 特征变量之间的相关性

Fig.3.3 The correlation between the characteristic variables

从图中可得知，物业税税率(TAX)于辐射高速公路可达性快速增长指数(RAD)的相关性绝对值大于 0.9。每市镇非零售商业用地的工业比(INDUS)与氮氧化物浓度(NOX)、分配到五个波士顿就业中心的距离(DIS)、物业税税率(TAX)相关程度大于 0.7，氮氧化物浓度(NOX)与楼龄比例 AGE、分配到五个波士顿就业中心的距离(DIS)相关程度大于 0.7。楼龄比例(AGE)与每市镇非零售商业用地的工业比(INDU)，分配到五个波士顿就业中心的距离(DIS)与居住用地比例(ZN)，辐射高速公路可达性快速增长指数(RAD)与人均犯罪率(CRIM)、氮氧化物浓度(NOX)，物业税税率(TAX)与氮氧化物浓度(NOX)，人口减少状态(LSTAT)与每市镇非零售商业用地的工业比(INDU)、住宅平均房屋数(RM)、楼龄比例(AGE)相关性绝对值大于 0.6.

（4）绘制关联矩阵

关联矩阵是包含皮尔森积矩相关系数的正方形矩阵，用来度量特征对之间的线性依赖关系。如图 3.3 所示。

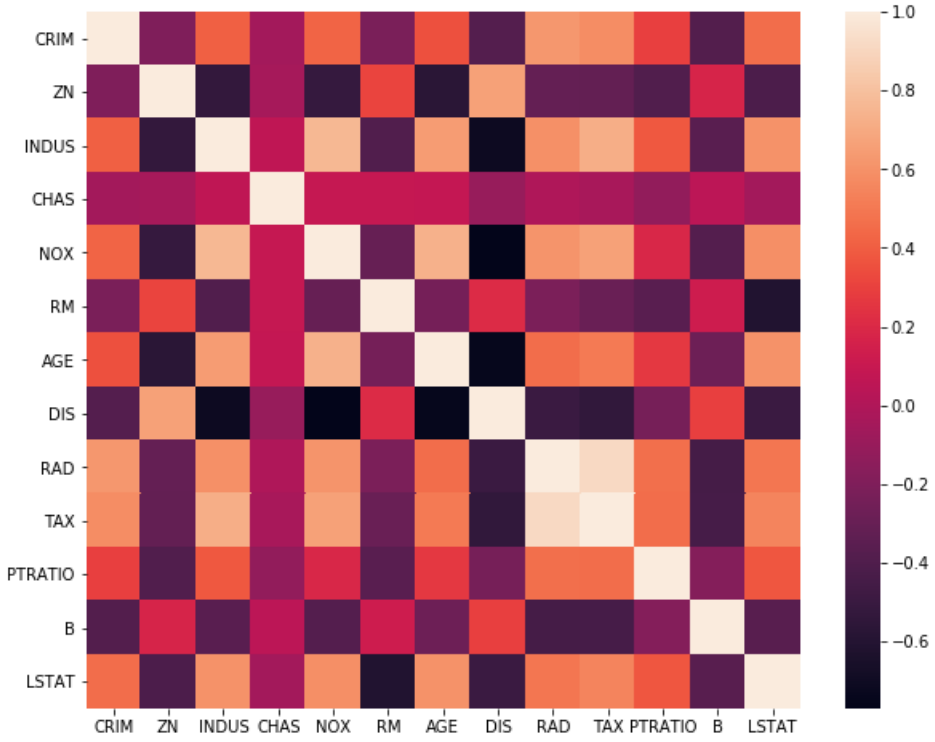


图 3.3 特征向量之间的关联矩阵

Fig.3.3 The incidence matrix between the eigenvectors

从每个正方形色块的颜色深浅，可以直观看出特征之间的相关性大小。

（5）删除异常值，处理之前样本规模是 506，删除异常值之后变成了 490.

（6）进行随机森林回归预测，选取样本集合的百分之 75 作为训练集，百分之 25 作为测试集用于检验回归效果。

（7）对随机森林预测结果进行评价，得分为 0.84。

（8）为了检验预测效果准确度，我们将随机森林与决策树预测效果进行对比。这一次通过正交检验的方法分别评价两种模型的预测效果。决策树的得分为 0.7325564358740533，随机森林的得分为 0.7325564358740533。很明显，随机森林的预测效果要优于决策树。

### 3.2.2 对连续函数的预测

为了检验随机森林算法对连续变量的预测能力，我们使用该算法对连续函数进行预测，流程如下。

（1）构造一个连续函数，形式如下:

$$y = 0.5 * \cos x + 0.4 * \sin x$$

选择训练集样本为$x$取值为 0 到 60，训练 500 个值。测试集样本为$x$取值 60 到 200，预测 500 个$y$值。

训练集可视化如图 3.4 所示。

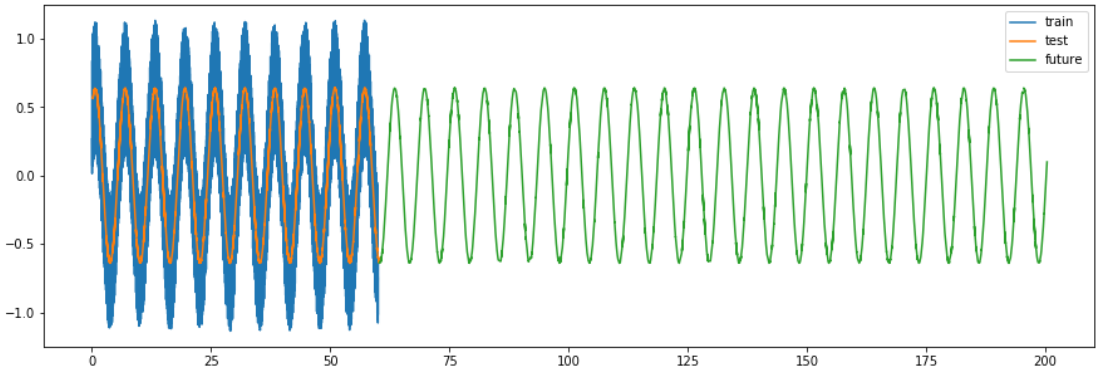


图 3.3 训练集可视化效果

Fig.3.3 Visualization of training set

（2）首先，我们使用决策树回归模型进行预测，结果如图 3.5 所示。

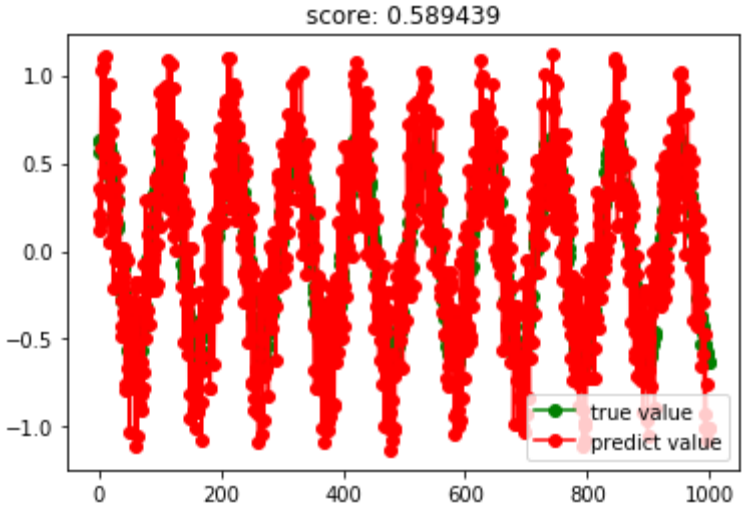

图 3.5 决策树回归预测效果

Fig.3.5 The predictive effect of decision tree regression

可以看出，使用决策树进行连续变量回归分析时，准确率很低，不能打到很好的拟合效果。

（3）使用 bagging 进行预测

使用 bagging 模型预测效果如图 3.6 所示。

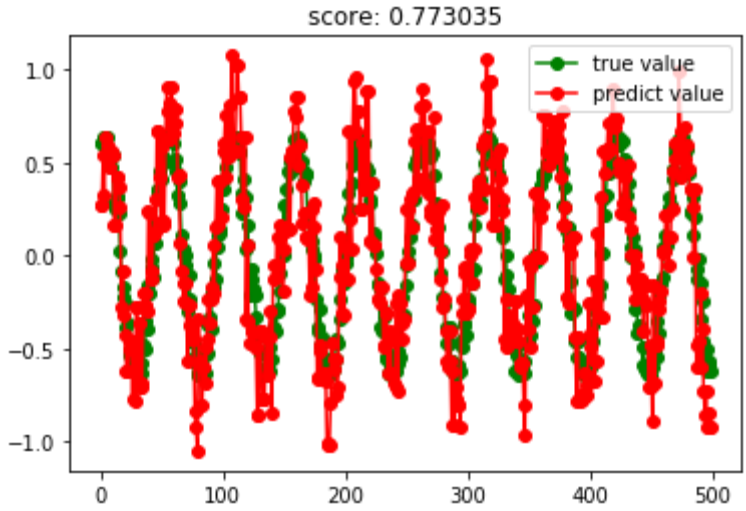


图 3.6 bagging 回归预测效果

Fig.3.6 The predictive effect of bagging

相对于决策树回归模型，使用 bagging 这种集成学习的思想进行回归分析，效果要远优于单一弱回归器的模拟效果。

（4）使用随机森林模型进行预测

我们设置决策树的数量为 50 棵，随机森林模型预测效果如图 3.7 所示。

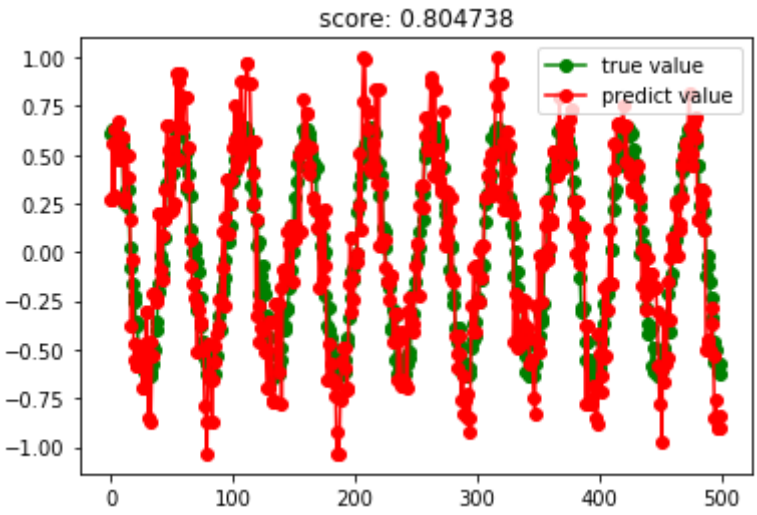


图 3.7 随机森林模型回归预测效果（决策树数目为 50）

Fig.3.7 The predictive effect of RF(number of decision trees is 50)

从图中可以看出，随机森林模型进行连续函数的回归分析时，效果要优于 bagging 思想

（5）将决策树数量调整到 20，随机森林模型预测结果如图 3.8 所示。

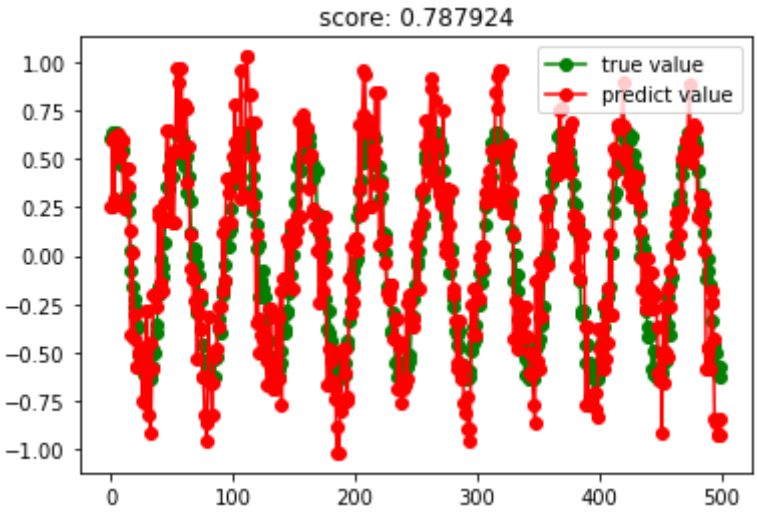


图 3.7 随机森林模型回归预测效果（决策树数目为 20）

Fig.3.7 The predictive effect of RF(number of decision trees is 20)

由图可知，当决策树数量为 20 时，预测效果降低。

（6）将决策树数量调整为 200，预测结果如图 3.9 所示。

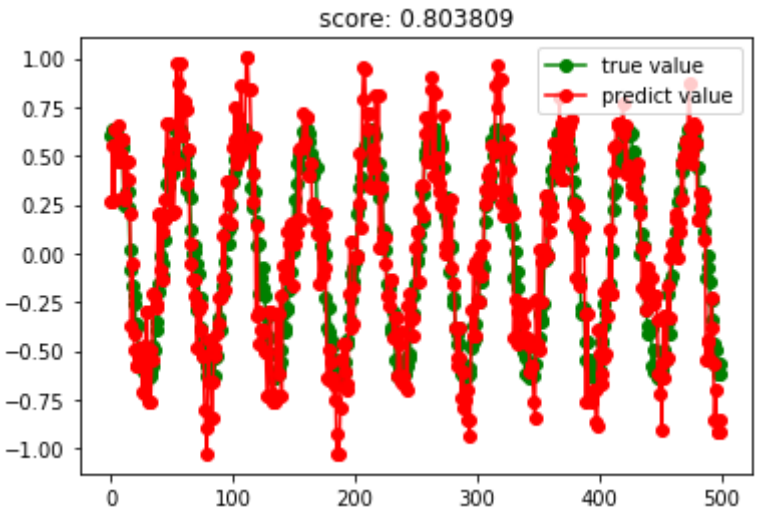


图 3.7 随机森林模型回归预测效果（决策树数目为 200）

Fig.3.7 The predictive effect of RF(number of decision trees is 200)

从图中可知，当决策树数量为 200 时，预测效果与决策树数量为 50 时差不多。我们可以得出结论：当决策树达到一定数量时，预测结果不会进一步优化。

# 第 4 章　随机森林算法处理分类问题

## 4.1 数学原理

用于分类的梯度提升与回归情况非常相似。但是，树木的总和$F_m(x)$与预测不是同类的：因为回归分析必须预测连续的值。

从值的映射 $F_m(x)$一个类别或一个概率取决于损失值。对于损失值（或对数损失）， $x_i$属于正类被建模为

$$p(y_i = 1|x_i) = \sigma\big(F_M(x_i)\big) \tag{4.1}$$

对于多重分类，K 棵树取决于每一个 m 的迭代次数。

请注意，即使对于分类任务，$h_m$ 子估算器仍然是回归器，而不是分类器。这是因为训练了子估计量以预测（负）梯度，该梯度始终是连续量。

分类问题的损失值可以用以下几种方法计算：

（1）二项式偏差：二元分类的负二项式对数似然损失函数（提供概率估计）。初始模型由对数比值比给出。

（2）多项式偏差：具有互斥类的多类分类的负多项式对数似然损失函数 ，它提供了概率估计。初始模型由每个类别的先验概率给出。在每次迭代中，类的数目都必须构建回归树。

## 4.2 实例分析

使用随机森林算法构造的分类器对鸢尾花种类的预测，通过分析花萼的长度与宽度两种特征变量对鸢尾花的种类进行分类。模型构造过程如下：

（1）先使用决策树模型对鸢尾花类别进行分类，选取树木最大深度为 4，特征划分标准为 gini 系数进行，百分之七十五的数据作为训练集，百分之二十五的数据作为测试集。

（2）对决策树模型进行评分，得出结果为 0.93。

（3）依次检验决策树的不同深度时的错误率大小，如图 4.1 所示。

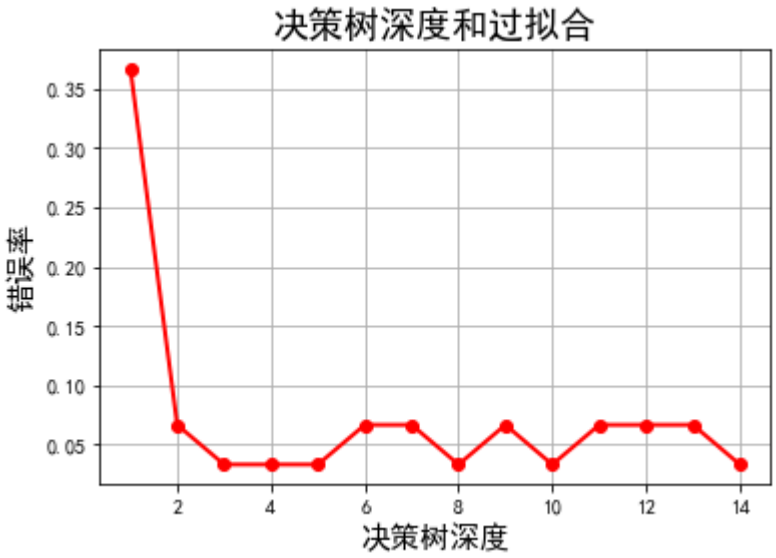


图 4.1 决策树深度与错误率的关系

Fig. 4.1 relation between decision tree depth and error rate

（5）使用随机森林模型进行预测，然后对比决策树数量不同时的准确率大小。设定决策树数量为 10，得分为 0.98。这个效果要高于单颗决策树的预测效果。

（6）计算特征向量影响花瓣类型的程度，如图 4.2 所示。

```
sepal length (cm) 0.10113161370082478
sepal width (cm) 0.024607359175393265
petal length (cm) 0.43934779617055864
petal width (cm) 0.43491323095322343
```

图 4.2 特征向量重要性大小

Fig.4.2 importance of feature vectors

从图中可以看出，花萼的长度和宽度对决策树影响最大。

# 第 5 章 结论

本文介绍了随机森林的历史背景和国内外使用该算法处理各种学科领域的问题以及算法的改进，并且将构成算法的基本原理加以概括。使用了三个案例具体讨论了随机森林再回归分析和判别分析中的具体应用，将其与单一分类器的效果做对比，发现这种集成了强学习器的算法再精确度上要由于单一弱学习器。并且，随机森林更适用于处理连续的变量，当变量为连续时，应先将其转化为离散型数据再进行处理。

随机森林能够处理很多高纬度数据，并且对于单个学习器而言具有很强的容错能力。每个学习器之间是相互独立的，更加提高了泛化能力。而且在训练过程中，能够计算出特征向量之间的相互联系。既能处理回归问题也可以处理分类问题，在两种情况下都有很好的表现。

# 参考文献

[1] Breiman，L.Bagging Predictors[J].Machine Learning,1996,24(2):123-140.

[2] LEO BREIMAN.Random Forests[J].Machine learning,2001,45(1):5-32.

[期刊论文] LEO BREIMAN- 《Machine learning》2001 年 1 期

[3] 彭潮,兰彦冰,邹春,蔡磊.煤粉富氧燃烧着火温度预测的优化随机森林(GA-RF)模型[J].洁净煤技术,2020,26(01):71-76.

[4] 胡梦珺,王佳,张亚云,李春艳,李娜娜.基于随机森林评价的兰州市主城区校园地表灰尘重金属污染[J].环境科学,2020,41(04):1838-1846.

[5] 陈军飞,董然.基于随机森林算法的洪水灾害风险评估研究[J].水利经济,2019,37(03):55-61+87.

[6] 杭琦. 随机森林算法在城市空气质量评价中的应用研究[D].上海第二工业大学,2019.

[7] 包青岭,丁建丽,王敬哲,蔡亮红.基于随机森林算法的土壤有机质含量高光谱检测[J].干旱区地理,2019,42(06):1404-1414.

[8] 潘登,郁培义,吴强.基于气象因子的随机森林算法在湘中丘陵区林火预测中的应用[J].西北林学院学报,2018,33(03):169-177.

[9] 孙凯.随机森林在医学影像分析中的应用研究进展[J].北京生物医学工程,2018,37(04):413-418.

[10] 杨美洁,唐建军.基于随机森林算法的糖尿病预测研究[J].医学信息学杂志,2019,40(09):47-49.

[11] 张英男,吴宇航,李赫,桂预风.基于随机森林算法的阿尔茨海默病预测模型[J].临床医药文献电子杂志,2018,5(45):191-192.

[12] 王平,单文英.改进的随机森林算法在乳腺肿瘤诊断中的应用[J].计算机应用与软件,2016,33(04):252-257+264.

[13] 郭建山,钱军浩.基于随机森林的信用卡违约预测研究[J].现代信息科技,2020,4(03):1-4+9.

[14] 马晓君,董碧滢,王常欣.一种基于 PSO 优化加权随机森林算法的上市公司信用评级模型设计[J].数量经济技术经济研究,2019,36(12):165-182.

[15] 朱青.开放式基金风险评级研究——层次聚类法和随机森林算法的应用[J].金融管理研究,2018(02):137-152.

[16] 张潇,韦增欣.随机森林在股票趋势预测中的应用[J].中国管理信息化,2018,21(03):120-123.

[17] Jerome H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. 2001, 29(5):1189-1232.

[18] GEURTSP,ERNST D,WEHENKEL L.Extremely randomized trees[J].Mach.Learn.2006,63(1):3-42.

[19] Quadrianto Novi,Ghahramani Zoubin. A Very Simple Safe-Bayesian Random Forest.[J]. IEEE transactions on pattern analysis and machine intelligence,2015,37(6).

[20] Quinlan J R.Induction of decision trees[J].Machine Learning,1986,1(1):81-106.

[21] Quinlan J R.C4.5:programs for machine learning[J].1992.

# 附录 A 波士顿房价预测关键代码

```
y = load_boston().target
def dataProcessing(df):
    field_cut = {
    'CRIM' : [0,10,20, 100],
    'ZN' : [-1, 5, 18, 20, 40, 80, 86, 100],
    'INDUS' : [-1, 7, 15, 23, 40],
    'NOX' : [0, 0.51, 0.6, 0.7, 0.8, 1],
    'RM' : [0, 4, 5, 6, 7, 8, 9],
    'AGE' : [0, 60, 80, 100],
    'DIS' : [0, 2, 6, 14],
    'RAD' : [0, 5, 10, 25],
    'TAX' : [0, 200, 400, 500, 800],
    'PTRATIO' : [0, 14, 20, 23],
    'B' : [0, 100, 350, 450],
    'LSTAT' : [0, 5, 10, 20, 40]
    }
    df = df[load_boston().feature_names].copy()
    cut_df = pd.DataFrame()
    for field in field_cut.keys():
        cut_series = pd.cut(df[field], field_cut[field], right=True)
        onehot_df = pd.get_dummies(cut_series, prefix=field)
        cut_df = pd.concat([cut_df, onehot_df], axis=1)
    new_df = pd.concat([df, cut_df], axis=1)
    return new_df
dfboston = pd.DataFrame(load_boston().data, columns=load_boston().feature_names)
new_df = dataProcessing(df)
print(new_df.columns)
new_df.head()
dfboston.info()
dfboston.describe([0.01,0.99])
plt.figure(figsize=(10,8))
sns.heatmap(dfboston.corr())
dfboston.corr()[np.abs(dfboston.corr())>0.5]
dfboston['price']=pd.Series(boston['target'])
dfboston.describe()
dfboston.corr()
X = new_df.values
```

```
y = load_boston().target
print(X.shape)
X = X[y!=50]
y = y[y!=50]
print(X.shape)
boston = load_boston()
print(boston.DESCR)
X = boston.data
y = boston.target
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.25,random_state =
33)
rfr=RandomForestRegressor()
rfr.fit(X_train,y_train)
rfr_y_predict=rfr.predict(X_test)
DecisionTree_model = DecisionTreeRegressor()
kf = KFold(n_splits=5, shuffle=True)
score_ndarray = cross_val_score(DecisionTree_model, X, y, cv=kf)
print(score_ndarray)
print(score_ndarray.mean())
randomForest_model = RandomForestRegressor()
kf = KFold(n_splits=5, shuffle=True)
score_ndarray = cross_val_score(randomForest_model, X, y, cv=kf)
print(score_ndarray)
print(score_ndarray.mean())
```

# 附录 B　线性函数的预测关键代码

```
import numpy as np
import matplotlib.pyplot as plt
def f(x1):
    y = 0.5 *np.cos(x1) + 0.4 * np.sin(x1)
    return y
def load_data():
    x1_train = np.linspace(0,60,500)
    data_train = np.array([[x1,f(x1) + (np.random.random(1)-0.5)] for x1 in
x1_train])
    x1_test = np.linspace(0,60,500)+ 0.5 * np.random.random(500)
    data_test = np.array([[x1,f(x1)] for x1 in x1_test])
    x1_future = np.linspace(60,200,500)+ 0.5 * np.random.random(500)
    data_future = np.array([[x1,f(x1)] for x1 in x1_future])
    return data_train, data_test, data_future
train, test, future = load_data()
x_train, y_train = train[:,:1], train[:,1]
x_test ,y_test = test[:,:1], test[:,1]
x_future, y_future = future[:,:1] ,future[:,1]
plt.figure(figsize=(15,5))
plt.plot(x_train[:,0],y_train,label='train')
plt.plot(x_test[:,0],y_test,label='test')
plt.plot(x_future[:,0],y_future,label='future')
plt.legend()
plt.show()
model_DecisionTreeRegressor = tree.DecisionTreeRegressor()
from sklearn import svm
model_SVR = svm.SVR()
from sklearn import ensemble
model_RandomForestRegressor =
ensemble.RandomForestRegressor(n_estimators=200)#这里使用 200 个决策树
from sklearn.ensemble import BaggingRegressor
model_BaggingRegressor = BaggingRegressor()
try_different_method(model_DecisionTreeRegressor)
try_different_method(model_RandomForestRegressor)
try_different_method(model_BaggingRegressor)
```

# 附录 C　鸢尾花数据集的分类关键代码

```
iris = datasets.load_iris()
x=iris['data']
y=iris['target']
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.8)
tree_clf = DecisionTreeClassifier(max_depth=4, criterion='gini')
tree_clf.fit(x_train, y_train)
y_test_hat = tree_clf.predict(x_test)
print("acc score:", accuracy_score(y_test, y_test_hat))
depth = np.arange(1, 15)
err_list = []
for d in depth:
    clf = DecisionTreeClassifier(criterion='entropy', max_depth=d)
    clf.fit(x_train, y_train)
    y_test_hat = clf.predict(x_test)
    result = (y_test_hat == y_test)
    err = 1 - np.mean(result)
    err_list.append(err)
    print(d, '错误率：%.2f%%' % (100 * err))
mpl.rcParams['font.sans-serif'] = ['SimHei']
plt.figure(facecolor='w')
plt.plot(depth, err_list, 'ro-', lw=2)
plt.xlabel('决策树深度', fontsize=15)
plt.ylabel('错误率', fontsize=15)
plt.title('决策树深度和过拟合', fontsize=18)
plt.grid(True)
plt.show()
###使用决策树进行分类，特征向量划分标准为基尼系数。并且绘制当树的深度不同时的精确度大小###
iris = datasets.load_iris()
X=iris['data']
y=iris['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=42)
rnd_clf = RandomForestClassifier(n_estimators=10)
rnd_clf.fit(X_train, y_train)
y_pred_rf = rnd_clf.predict(X_test)
```

```
print(accuracy_score(y_test, y_pred_rf))
iris = load_iris()
rnd_clf = RandomForestClassifier(n_estimators=500, n_jobs=-1)
rnd_clf.fit(iris["data"], iris['target'])
for name, score in zip(iris['feature_names'], rnd_clf.feature_importances_):
print(name, score)
```

###使用随机森林算法进行分类，并且计算特征变量对目标值的影响程度###

# 致谢

随着在键盘上敲下最后一个字符，我意识到，四年的本科生涯已经结束了。这四年一幕幕的汗水与成就，失落与喜悦一下子冲进了我的脑海。时光荏苒，仿佛上的一堂堂课，考的一次次试还发生在昨天。虽然不舍离开，但是朝前看会有更多的精彩，因为人生就是为了不甘平庸的去创造一次又一次的惊喜，去踏上更高一层的台阶。

这四年，我要感谢所有不辞疲倦站在讲台上传授知识的老师们，是他们的汗水，人类的智慧的结晶才能得到传承，我们才能创造更好的价值。其次，我要感谢我身边的同学，是他们的鼓励与陪伴让我在本科四年中不再是孤身一人，在每次失败中能够重振旗鼓坚持下去。

在几个月的论文写作过程中，我要感谢我的指导老师武国宁老师。从选题开始，武老师了解每个人合适的领域，因材施教，选出最适合我们的题目。在论文写作过程中，武老师孜孜不倦的解答了我的疑惑，尽全力帮助论文的完成。再他的帮助下我才能完成这片论文的写作

四年的石大生活给予我许多珍贵的记忆，教会我很多的品质。博学、睿思、行健、至善这 8 个字将激励我不断前行。

最后，我要郑重感谢我大学时期各位老师，您们对于我的关怀与帮助是我永生难忘的记忆，我会在今后的学习中、工作中不畏困难、不忘初心、脚踏实地，用我的知识更好的反馈于社会，帮助需要帮助的人。再次感谢，祝您们工作顺利，身体安康！