随机森林相关知识介绍

关晓蔷

2014.01

1、决策树

2、分类器组合——集成学习

3、随机森林

1、决策树

2、分类器组合——集成学习

3、随机森林

决策树概述

决策树(Decision Tree)

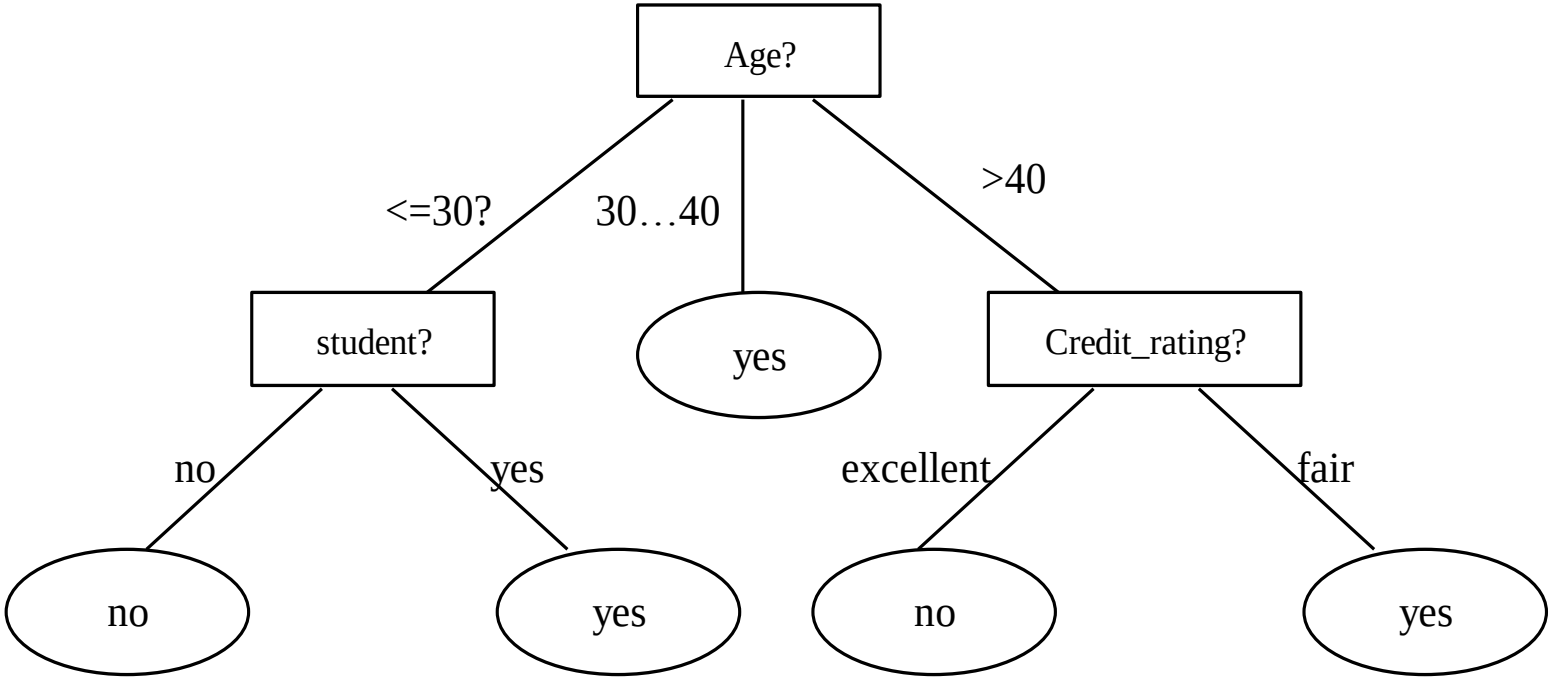
一种描述概念空间的有效的归纳推理办法。基于决策树学习方法可以进行不相关的多概念学习，具有简单的快捷的优势，已经在各个领域取得广泛应用。

决策树方法的起源是亨特（Hunt，1966）的概念学习系统CLS方法，然后发展到由Quinlan研制的ID3方法，然后到著名的C4.5算法，C4.5算法的一个优点是它能够处理连续属性。还有CART算法也是比较有名的决策树方法。

- **决策树是一种树型结构，其中每个内部结点表示在一个属性上的测试，每个分支代表一个测试输出，每个叶结点代表某个类或者类的分布。**

- **决策树提供了一种展示类似在什么条件下会得到什么值这类规则的方法。**下例是为了解决这个问题而建立的一棵决策树，从中可以看到决策树的基本组成部分：决策结点、分支和叶结点。

〖例〗给出了一个商业上使用的决策树的例子。它表示了一个关心电子产品的用户是否会购买PC的知识，用它可以预测某条记录（某个人）的购买意向。



buys_computer的决策树

- 决策树学习采用的是自顶向下的递归方法。
- 决策树的每一层结点依照某一属性值向下分为子结点，待分类的实例在每一结点处与该结点相关的属性值进行比较，根据不同的比较结果向相应的子结点扩展，这一过程在到达决策树的叶结点时结束，此时得到结论。
- 从根结点到叶结点的每一条路经都对应着一条合理的规则，规则间各个部分（各个层的条件）的关系是合取关系。整个决策树就对应着一组析取的规则。

决策树的优点

- 使用者不需要了解很多背景知识，只要训练事例能用属性→结论的方式表达出来，就能用该算法学习；
- 进行分类器设计时，决策树分类方法所需时间相对较少；
- 决策树的分类模型是树状结构，简单直观，可将到达每个叶结点的路径转换为IF→THEN形式的规则，易于理解；
- 决策树模型效率高，对训练集数据量较大的情况较为适合；
- 决策树方法具有较高的分类精确度。

使用决策树进行分类

使用决策树进行分类分为两步：

- 第1步：利用训练集建立一棵决策树，建立决策树模型。这个过程实际上是一个从数据中获取知识，进行机器学习的过程。
- 第2步：利用生成完毕的决策树对输入数据进行分类。对输入的记录，从根结点依次测试记录的属性值，直到到达某个叶结点，从而找到该记录所在的类。

问题的关键是建立一棵决策树。这个过程通常分为两个阶段：

- 建树（Tree Building）：这是一个递归的过程。
- 剪枝（Tree Pruning）：剪枝的目的是降低由于训练集存在噪声而产生的起伏。

关于过渡拟合

上述的决策树算法增长树的每一个分支的深度，直到恰好能对训练样例比较完美地分类。实际应用中，当数据中有噪声或训练样例的数量太少以至于不能产生目标函数的有代表性的采样时，该策略可能会遇到困难。

在以上情况发生时，这个简单的算法产生的树会过渡拟合训练样例（过渡拟合：**Over Fitting**）。

分类模型的误差

一般可以将分类模型的误差分为：

1、训练误差（Training Error）；

2、泛化误差（Generalization Error）

分类模型的误差

训练误差是在训练记录上误分类样本比例；

泛化误差是模型在未知记录上的期望误差；

一个好的模型不仅要能够很好地拟合训练数据，而且对未知样本也要能够准确地分类。

一个好的分类模型必须具有低的训练误差和泛化误差。因为一个具有低训练误差的模型，其泛化误差可能比具有较高训练误差的模型高。（训练误差低，泛化误差高，称为过渡拟合）

解决过度拟合的手段是剪枝

预剪枝（前剪枝）

通过提前停止树的构造来对决策树进行剪枝
一旦停止该结点下树的继续构造，该结点就成了叶结点。
该叶结点持有其数据集中样本最多的类或者其概率分布

后剪枝

首先构造完整的决策树，允许决策树过度拟合训练数据，然后对那些置信度不够的结点的子树用叶结点来替代
该叶结点持有其子树的数据集中样本最多的类或者其概率分布

属性选择度量

- 属性选择度量是一种选择分裂准则，将给定类标号的训练元组最好的进行划分的方法
  - 理想情况，每个划分都是“纯”的，即落在给定划分内的元组都属于相同的类
  - 属性选择度量又称为分裂准则
- 常用的属性选择度量
  - 信息增益
  - 信息增益率
  - Gini指标

ID3算法

由Quinlan在1986年提出的ID3算法是分类规则挖掘算法中最有影响的算法。

ID3算法以信息增益作为属性的选择标准，以使得在对每一个非叶结点进行测试时，能获得关于被测试记录最大的类别信息

ID3算法

ID3 总是选择具有最高信息增益的属性作为当前结点的测试属性。

具体方法是:检测所有的属性，选择信息增益最大的属性产生决策树结点，由该属性的不同取值建立分支，再对各分支的子集递归调用该方法建立决策树结点的分支，直到所有子集仅包含同一类别的数据为止，最后得到一棵决策树，它可以用来对新的样本进行分类。

信息增益

S是一个训练样本的集合，该样本中每个集合的类编号已知。假设**S**中有***m***个类，总共**s**个训练样本，每个类C_i有s_i个样本(***i*** **=1,2,3...*m*)**，那么任意一个样本属于类C_i的概率是s_i / s，那么用来分类一个给定样本的*期望信息*是：

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} \frac{s_i}{s} \log_2 \frac{s_i}{s}$$

信息增益

一个有v个值的属性$A\{a_1,a_2,\ldots,a_v\}$可以将S分成v个子集$\{S_1,S_2,\ldots,S_v\}$，其中S_j包含S中属性A上的值为a_j的样本。假设S_j包含类C_i的s_{ij}个样本。根据A的这种划分的期望信息称为A的*熵*

$$E(A)=\sum_{j=1}^{v}\frac{s_{1j}+\ldots+s_{mj}}{s}I(s_{1j},\ldots,s_{mj})$$

A上该划分的获得的信息增益定义为：

$$Gain(A) = I(s_1, s_2, ..., s_m) - E(A)$$

具有高信息增益的属性，是给定集合中具有高区分度的属性。所以可以通过计算**S**中样本的每个属性的信息增益，来得到一个属性的相关性的排序。

No.	年龄	收入水平	有固定收入	VIP	类别：提供贷款
1	<30	高	否	否	否
2	<30	高	否	是	否
3	[30,50]	高	否	否	是
4	>50	中	否	否	是
5	>50	低	是	否	是
6	>50	低	是	是	否
7	[30,50]	低	是	是	是
8	<30	中	否	否	否
9	<30	低	是	否	是
10	>50	中	是	否	是
11	<30	中	是	是	是
12	[30,50]	中	否	是	是
13	[30,50]	高	是	否	是
14	>50	中	否	是	否

$$E(\text{年龄}) = \frac{5}{14}\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) + \frac{4}{14}\left(-\frac{4}{4}\log_2\frac{4}{4}\right) + \frac{5}{14}\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right)$$

$$= 0.3468 + 0 + 0.3468 = 0.6936$$

No.	年龄	收入水平	有固定收入	VIP	类别：提供贷款
1	<30	高	否	否	否
2	<30	高	否	是	否
3	[30,50]	高	否	否	是
4	>50	中	否	否	是
5	>50	低	是	否	是
6	>50	低	是	是	否
7	[30,50]	低	是	是	是
8	<30	中	否	否	否
9	<30	低	是	否	是
10	>50	中	是	否	是
11	<30	中	是	是	是
12	[30,50]	中	否	是	是
13	[30,50]	高	是	否	是
14	>50	中	否	是	否

$$I(s_1, s_2) = I(9,5) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.9406$$

$$Gain(\text{年龄}) = I(s_1, s_2) - E(\text{年龄}) = 0.9406 - 0.6936 = 0.247$$

ID3算法的评价

优点：

利用信息增益的概念，算法的基础理论清晰，算法较简单.

不足：

(1)信息增益选择属性时偏向于选择取值多的属性；

(2)不能处理连续属性；

(3)对噪声（特征取值取错或类别给错）较为敏感 ；

(4)不易对变化的数据集进行学习。

(5)无法处理缺失数据

C4.5算法

C4.5是对ID3算法的改进，它主要克服了ID3在应用中存在的不足，主要体现在以下几个方面：

（1）用信息增益率来选择属性，克服了用信息增益选择属性时偏向于选择取值多的属性的不足；

（2）在树构造完成之后，进行剪枝；

（3）能够完成对连续属性的离散化处理；

（4）能够对不完整数据进行处理；

ID3是采用“信息增益”来选择分裂属性的。虽然这是一种有效的方法，但其具有明显的倾向性，即它倾向于选择具有大量不同取值的属性，从而产生许多小而纯的子集。

尤其是关系数据库中作为主键的属性，每一个样本都有一个不同的取值。如果以这样的属性作为分裂属性，那么将产生非常多的分支，而且每一个分支产生的子集的熵均为0（因为子集中只有一个样本！）。显然，这样的决策树是没有实际意义的。因此，Quinlan提出使用增益比例来代替信息增益。

设S代表训练数据集，由s个样本组成。A是S的某个属性，有m个不同的取值，根据这些取值可以把S划分为m个子集，S_i表示第i个子集（i=1,2,…,m），$|S_i|$表示子集S_i中的样本数量。那么：

$$Split\_Info(S,A) = -\sum_{i=1}^{m}(\frac{|S_i|}{s}\log_2\frac{|S_i|}{s})$$

称为“数据集S关于属性A的熵”。

$Split\_Info(S,A)$ 用来衡量属性**A**分裂数据集的广度和均匀性。样本在属性**A**上的取值分布越均匀，**Split_Info(S,A)**的值就越大。

增益比例的定义为：

$$GainRatio(S,A)=\frac{Gain(A)}{Split\_Info(S,A)}$$

增益比例消除了选择那些值较多且均匀分布的属性作为分裂属性的倾向性。

连续属性的处理

设属性Y有m个不同的取值，按大小顺序升序排列为$v_1<v_2<,\ldots,<v_m$。

从$\{v_1,v_2,\ldots,v_m\}$中选择一个v_i作为阈值，则可以根据“$Y \leq v_i$”和“$Y> v_i$”将数据集划分为两个部分，形成两个分支。显然，$\{v_1,v_2,\ldots,v_m\}$就是可能的阈值的集合，共m个元素。

把这些阈值一一取出来，并根据“$Y \leq v_i$”和“$Y> v_i$”把训练数据集划分为两个子集，并计算每一种划分方案下的信息增益率，选择最大信息增益率所对应的那个阈值，作为最优的阈值。

No.	年龄	收入水平	有固定收入	VIP	类别：提供贷款
1	25	高	否	否	否
2	28	高	否	是	否
3	40	高	否	否	是
4	56	中	否	否	是
5	60	低	是	否	是
6	65	低	是	是	否
7	40	低	是	是	是
8	25	中	否	否	否
9	28	低	是	否	是
10	55	中	是	否	是
11	20	?	是	是	是
12	46	中	否	是	是
13	58	?	是	否	是
14	70	中	否	是	否

如果要计算“年龄”属性的信息增益，则首先将不同的属性值排序

{20,25,28,40,46,55,56,58,60,65,70}

那么可能的阈值集合为

{20,25,28,40,46,55,56,58,60,65,70}，从中一一取出，并形成分裂谓词，例如取出“**20**”，形成谓词“**≤20**”和“**>20**”，用它们划分训练数据集，然后计算信息增益率。

CART

分类和回归树（**Classification and Regression Trees**）

CART算法中的每一次分裂把数据分为两个子集，每个子集中的样本比被划分之前具有更好的一致性。它是一个递归的过程，也就是说，这些子集还会被继续划分，这个过程不断重复，直到满足终止准则，然后通过修剪和评估，得到一棵最优的决策树。

三个步骤

- 生成最大树
  - 生成一棵充分生长的最大树
- 树的修剪
  - 根据修剪算法对最大树进行修剪，生成由许多子树组成的子树序列
- 子树评估
  - 从子树序列中选择一棵最优的子树作为最后的结果。

基尼指数（Gini Index）

在分类问题中，假设有K个类，样本点属于第k类的概率为p_k,则对于给定的样本集合D，其基尼指数定义为：

$$Gini(D)=1-\sum_{k=1}^{K}p_k^2$$

基尼指数（Gini Index）

- 如果特征***A***将样本集合***D***分成两个子集D_1 和D_2 。则在特征***A***的条件下，集合***D***的基尼指数定义为：

$$Gini(D,A)=\frac{|D_1|}{|D|}Gini(D_1)+\frac{|D_2|}{|D|}Gini(D_2)$$

- 基尼指数 $Gini(D)$ 表示集合**D**的不确定性，基尼指数 $Gini(D,A)$ 表示经过属性**A**划分后集合**D**的不确定性，基尼指数越大，样本集合的不确定性也就越大。

停止准则

以下任何一个规则被满足，结点将不再分裂

- 这个结点是“纯”的，即这个结点的所有样本都属于同一类别；
- 测试属性集为空；
- 当前结点所在的深度已经达到“最大树深度”（如果定义有）；
- 这个结点的样本数量小于“父分支中的最小记录数”（如果定义有）；

树的修剪

- 叶子结点过多，则树的复杂度高。
- 叶子结点过少，则误分类损失大。
- 子树的损失函数

$$C_\alpha(T) = C(T) + \alpha|T|$$

其中，T为任意子树，$C(T)$为对训练数据的预测误差，$|T|$为子树T的叶结点个数，$\alpha \geq 0$ 为参数，$C_\alpha(T)$为参数是α时的子树T的整体损失。参数α权衡训练数据的拟合程度与模型的复杂度。

- 对固定的α，一定存在使损失函数 $C_\alpha(T)$ 最小的子树，将其表示为T_α。 T_α在损失函数 $C_\alpha(T)$ 最小的意义下是最优的。容易验证这样的最优子树是唯一的。当α大的时候，最优子树偏小，当α小的时候，最优子树偏大。在极端的情况下，当$\alpha=0$时，整体树是最优的，当$\alpha \to \infty$时，根结点组成的单结点树是最优的。

树的修剪过程

Breiman等人证明：可以用递归的方法对树进行剪枝。将α从小增大。

- 令$\alpha = 0$，从T_1开始，这里的$T1$就是最大树$Tmax$；
- 逐渐增大α，直到某个结点使得 $C_\alpha(T_t) = C_\alpha(t)$ 成立，将它的分支删除，得到T_2；
- 重复上一步骤，直到被修剪到只有一个根节点，从而得到一个树的序列$T_1, T_2, \ldots, T_k$。

子树评估

要找到一棵分类准确性最好、同时结点数量尽量少的树 。

具体地，利用独立的验证数据集，测试子树序列$T1,T2,\ldots,T_k$中各棵子树的平方误差或基尼指数。平方误差或基尼指数最小的决策树被认为是最优的决策树。

1、决策树

2、分类器组合——集成学习

3、随机森林

集成学习

- 在生活中常常可以碰到一些判断，他们比随机的猜测要准确些，但本身又非常的不准确，可以称这种判断为弱分类器。
- 在一些文献中，弱分类器被定义为一个有低偏差高方差的预测函数，由于方差大，弱分类器通常是不精确的，也就是说与真实分类只有弱相关。
- 与此对应的是强分类器，强分类器与真实分布强相关。一个常常被提出的问题是能否用一系列弱分类器制造一个强分类器？对于这个问题的回答是肯定的，但解决的方法却不是唯一的。

以多个弱分类器组合成的分类器通常被称为集成分类器。

集成学习显著地提高一个学习系统的泛化能力。

所谓泛化能力（**generalization**），是指机器学习算法对新鲜样本的适应能力。

泛化能力越强，处理新数据的能力越好

泛化能力是机器学习关注的基本问题之一

- 提高泛化能力是永远的追求

集成学习的构建：

1、基学习机的生成

2、基学习机的合并

提高泛化能力的两个关键

1、基学习器具有较高的精度

2、各基学习器具有较高的差异度。

常见的集成分类器包括Boosting和Bagging。

Bagging算法

1、从大小为n的原始数据集D中独立随机地有放回的抽取n'个数据（$n' \leq n$），形成一个自助数据集；

2、重复上述过程，产生出多个独立的自助数据集；

3、利用每个自助数据集训练出一个“分量分类器”；

4、最终的分类结果由这些“分量分类器”各自的判别结果投票决定。

基本思想：对训练集有放回地抽取训练样例，从而为每一个基本分类器都构造出一个跟训练集相当大小但各不相同的训练集，从而训练出不同的基本分类器；该算法是基于对训练集进行处理的集成方法中最简单、最直观的一种。

Boosting算法

基本思想：

- 每个样本都赋予一个权重
- *T*次迭代，每次迭代后，对分类错误的样本加大权重，使得下一次的迭代更加关注这些样本。
- 学习多个分类器，并将这些分类器进行线性组合，提高分类的性能。

核心思想

样本的权重

- 没有先验知识的情况下，初始的分布应为等概分布，也就是训练集如果有*N*个样本，每个样本的分布概率为*1/N*
- 每次循环以后提高错误样本的分布概率，分错样本在训练集中所占权重增大， 使得下一次循环的弱学习机能够集中力量对这些错误样本进行判断。

弱学习机的组合

- 采取加权多数表决的方法。
- 加大分类误差率小的弱分类器的权值，使其在表决中起较大的作用，减小分类误差率大的弱分类器的权值，使其在表决中起较小的作用。

Bagging和boosting的区别

- bagging的训练集的选择是随机的，各轮训练集之间相互独立，而 boosting的训练集的选择不是独立的，各轮训练集的选择与前面各轮的学习结果有关；
- bagging的各个预测函数没有权重，而boosting是有权重的；
- bagging的各个预测函数可以并行生成，而boosting的各个预测函数只能顺序生成。。

1、决策树

2、分类器组合——集成学习

3、随机森林

随机森林

随机森林（Random Forests，RF）是由美国科学院院士Leo Breiman于2001 年提出的，它是一个以决策树为基础分类器的集成分类器。

Machine Learning, 45, 5–32, 2001

Random Forests

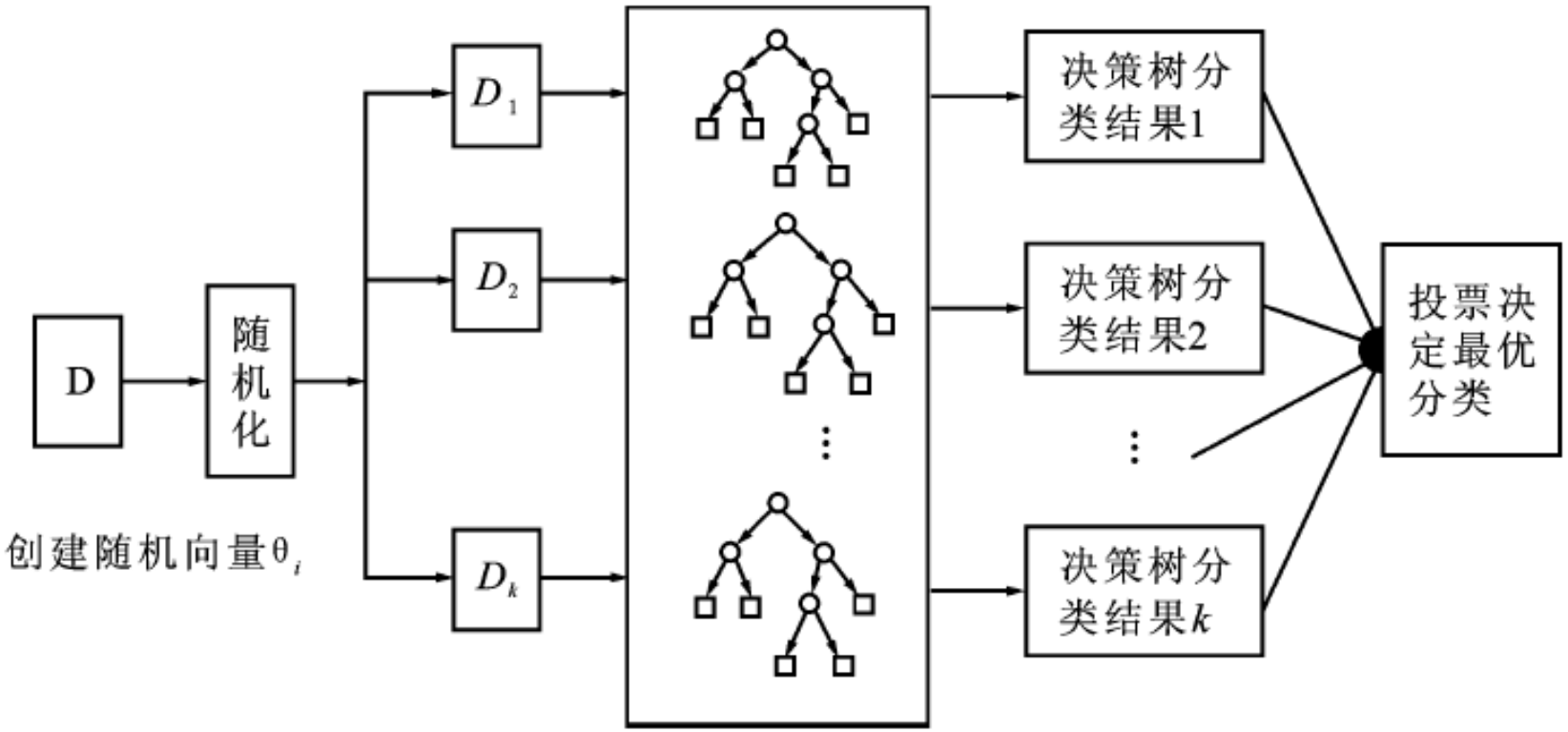
LEO BREIMAN
Statistics Department, University of California, Berkeley, CA 94720

在随机森林的训练中存在树和结点两种随机性：

- 树一级的随机是通过训练值采样得到的；
- 结点一级的随机则是通过随机选取参数值达到的。

随机森林定义

随机森林是一个由一系列树状分类器$\{h(x,\ \theta_k),\ k=1,\cdots\}$组成的分类器，这里$\{\theta_k\}$是独立同分布的随机向量，且每棵树为输入变量X归属于哪个最受欢迎的类投平等的一票。


D
随机化
创建随机向量θi
D1
D2
Dk
决策树分类结果1
决策树分类结果2
决策树分类结果k
投票决定最优分类

随机森林结构示意图

随机森林模型的构建

随机森林模型的构建可以分为以下四步：

1、**训练数据抽样**。即构造每一个分类树首先需要从原始训练数据集中以可放回的方式随机抽取出一部分样本作为训练数据子集；

2、**属性子空间抽样**。即从样本属性空间中以不放回的方式随机的选取一系列新的属性子空间；

3、**建立决策树模型**；

4、**建立随机森林模型**。即将所建立的所有决策树集成为随机森林模型。

当使用随机森林模型进行分类时，它是通过随机森林中各个树对分类结果进行投票而得到最终决策。随机森林中所有的分类树都自然生长，不进行剪枝。

随机森林的优点

- 随机森林分类表现优异；
- 能够有效避免过度拟合；
- 无需做特征选取或数据整理；
- 能够估计哪个特征在分类中更重要；
- 由简单的决策树构成，使得算法容易理解；
- 容易并行处理，这对于处理实际的海量数据非常合适；

Given an ensemble of classifiers $h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_K(\mathbf{x})$, and with the training set drawn at random from the distribution of the random vector Y, X, define the margin function as

$$mg(\mathbf{X}, Y) = av_k I(h_k(\mathbf{X}) = Y) - \max_{j \neq Y} av_k I(h_k(\mathbf{X}) = j).$$

where $I(\cdot)$ is the indicator function. The margin measures the extent to which the average number of votes at X, Y for the right class exceeds the average vote for any other class. The larger [illegible] ation error is given by

PE^*

边际函数用来测度平均正确分类数超过平均错误分类数的程度。边际函数值越大，分类预测就越可靠。

where the subscripts X, Y indicate that the probability is over the X, Y space.

随机森林的收敛性

In random forests, $h_k(\mathbf{X}) = h(\mathbf{X}, \Theta_k)$. For a large number of trees, it follows from the Strong Law of Large Numbers and the tree structure that:

Theorem 1.2. *As the number of trees increases, for almost surely all sequences* $\Theta_{1,\ldots}PE^*$ *converges to*

$$P_{\mathbf{X},Y}\left(P_{\Theta}(h(\mathbf{X},\Theta)=Y)-\max_{j\neq Y}P_{\Theta}(h(\mathbf{X},\Theta)=j)<0\right). \tag{1}$$

这个定理说明了为什么随着树的数目的增加，随机森林不过拟合，而是其泛化误差值收敛于一个极限值。

Definition 2.1. The margin function for a random forest is

$$mr(\mathbf{X}, Y) = P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_{\Theta}(h(\mathbf{X}, \Theta) = j) \tag{2}$$

and the strength of the set of classifiers $\{h(\mathbf{x}, \Theta)\}$ is

$$s = E_{\mathbf{X},Y} mr(\mathbf{X}, Y). \tag{3}$$

Assuming $s \geq 0$, Chebychev's inequality gives

$$PE^* \leq \text{var}(mr)/s^2 \tag{4}$$

A more revealing expression for the variance of mr is derived in the following: Let

$$\hat{j}(\mathbf{X}, Y) = \arg\max_{j \neq Y} P_{\Theta}(h(\mathbf{X}, \Theta) = j)$$

so

$$\begin{aligned} mr(\mathbf{X}, Y) &= P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - P_{\Theta}(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)) \\ &= E_{\Theta}[I(h(\mathbf{X}, \Theta) = Y) - I(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)]. \end{aligned}$$

Definition 2.2. The raw margin function is

$$rmg(\Theta, \mathbf{X}, Y) = I\left(h(\mathbf{X}, \Theta) = Y\right) - I\left(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)\right).$$

Thus, $mr(\mathbf{X}, Y)$ is the expectation of $rmg(\Theta, \mathbf{X}, Y)$ with respect to Θ. For any function f the identity

$$[E_{\Theta} f(\Theta)]^2 = E_{\Theta,\Theta'} f(\Theta) f(\Theta')$$

holds where Θ, Θ' are independent with the same distribution, implying that

$$mr(\mathbf{X}, Y)^2 = E_{\Theta,\Theta'} rmg(\Theta, \mathbf{X}, Y) rmg(\Theta', \mathbf{X}, Y) \tag{5}$$

Using (5) gives

$$\begin{aligned} \text{var}(mr) &= E_{\Theta,\Theta'}(\text{cov}_{\mathbf{X},Y} rmg(\Theta, \mathbf{X}, Y) rmg(\Theta', \mathbf{X}, Y)) \\ &= E_{\Theta,\Theta'}(\rho(\Theta, \Theta') sd(\Theta) sd(\Theta')) \end{aligned} \tag{6}$$

where $\rho(\Theta, \Theta')$ is the correlation between $rmg(\Theta, \mathbf{X}, Y)$ and $rmg(\Theta', \mathbf{X}, Y)$ holding Θ, Θ' fixed and $sd(\Theta)$ is the standard deviation of $rmg(\Theta, \mathbf{X}, Y)$ holding Θ fixed. Then,

$$\begin{aligned} \text{var}(mr) &= \bar{\rho}(E_\Theta sd(\Theta))^2 \\ &\leq \bar{\rho} E_\Theta \text{var}(\Theta) \end{aligned} \tag{7}$$

where $\bar{\rho}$ is the mean value of the correlation; that is,

$$\bar{\rho} = E_{\Theta,\Theta'}(\rho(\Theta, \Theta')sd(\Theta)sd(\Theta'))/E_{\Theta,\Theta'}(sd(\Theta)sd(\Theta'))$$

Write

$$\begin{aligned} E_{\Theta}\text{var}(\Theta) &\leq E_{\Theta}(E_{\mathbf{X},Y}rmg(\Theta, \mathbf{X}, Y))^2 - s^2 \\ &\leq 1 - s^2. \end{aligned} \tag{8}$$

Putting (4), (7), and (8) together yields:

随机森林泛化误差上界

Theorem 2.3. *An upper bound for the generalization error is given by*

$$PE^* \leq \bar{\rho}(1 - s^2)/s^2.$$

Although the bound is likely to be loose, it fulfills the same suggestive function for random forests as VC-type bounds do for other types of classifiers. It shows that the two ingredients involved in the generalization error for random forests are the strength of the individual classifiers in the forest, and the correlation between them in terms of the raw margin functions. The c/s2 ratio is the correlation divided by the square of the strength. In understanding the functioning of random forests, this ratio will be a helpful guide—the smaller it is, the better.

$\bar{\rho}$ 为随机森林中分类树之间的平均相关度，s为在训练数据集上利用OOB估计得到的随机森林分类器的强度。该不等式表明随机森林的泛化误差受到各分类器强度以及个分类器之间的相关性影响。

随机森林泛化误差的两个成分是随机森林的强度和分类模型间原始边际函数的相关系数。

随着树的相关性增加或组合分类模型的强度降低，泛化误差的上界趋向于增加，随机化有助于减少决策树间的相关性，从而改善组合分类模型的泛化误差。

基于 OOB 估计的度量方法

随机森林算法在使用 Bagging 方法随机抽取每个训练数据子集的时候，原始训练数据集中大约有 37% 的样本数据不会在其中出现，这些数据就成为各个训练数据子集的 Out-of-bag (OOB) 数据。这部分数据可以用来评价随机森林的分类性能，可以用来估计随机森林中各分类树的强度、分类树之间的相关度和随机森林的泛化误差[t

假设 D 为训练数据集，Y 为类标集，D_k 为从训练数据集 X 通过有重复抽样得到的第 k 个训练数据子集，$h_k(D_k)$ 为由 D_k 训练得到的第 k 个决策树分类器。分类器 $h_k(D_k)$ 对应的 Out-of-bag 数据集为 OOB_k。$Q(d_i, j)$ 为对输入的随机向量 d_i 在 OOB_k 中投票的分类类别为 j 的比例，记作 $P(h(d_i) = j)$:

$$Q(d_i, j) = \frac{\sum_{k=1}^{K} I(h_k(d_i) = j; d_i \in OOB_k)}{\sum_{k=1}^{K} I(d_i \in OOB_k)} \tag{2-4}$$

随机森林的边缘函数定义为：

$$mr(D, Y) = P(h_k(D) = Y) - max_{j \neq Y}^{c} P(h_k(D) = j) \tag{2-5}$$

同上定义的 $mg(D, Y)$ 意义，这个随机森林边缘函数的定义是整体所有决策树分类器的平均正确投票数超出投票到其他任何类上的测度。

- 随机森林的**强度（Strength）**

随机森林的强度是评价森林中决策树分类器总体分类能力的量，即用来衡量各分类树在随机森林中的联合性能。随机森林的分类树集 $h(D)$ 的强度为随机森林边缘函数的期望，可通过下面公式计算得到：

$$s = E(mr(D,Y)) = \frac{1}{n}\sum_{i=1}^{n}\left(Q(d_i,y_i) - \max_{j\neq y_i} Q(d_i,j)\right) \tag{2-6}$$

其中，n 为训练集 D 的数据个数，y_i 为数据 d_i 的真实类别，对于随机森林模型中每个决策树分类器来说，每个决策树分类器的强度越大，则随机森林模型整体的分类性能越好。

- 随机森林的**相关度（Correlation）**

随机森林的相关度是评价森林中各个决策树分类器之间总体相关度的量，即用于衡量随机森林中各决策树分类器之间的多样性。随机森林的各分类树之间的平均相关度为边缘函数的方差除以森林的标准差的平方，可通过下面公式计算得到:

$$\overline{\rho} = \frac{\frac{1}{n}\sum_{i=1}^{n}\left(Q(d_i, y_i) - \max Q_{j\neq y}(d_i, j)\right)^2 - s^2}{\left(\frac{1}{K}\sum_{k=1}^{K}\sqrt{p_k + \overline{p}_k + (p_k - \overline{p}_k)^2}\right)} \tag{2-7}$$

其中

$$p_k = \frac{\sum_{i=1}^{n} I(h_k(d_i) = y_i; d_i \in OOB_k)}{\sum_{i=1}^{n} I(d_i \in OOB_k)} \tag{2-8}$$

$$\overline{p}_k = \frac{\sum_{i=1}^{n} I(h_k(d_i) = \widehat{j}(d_i, Y); d_i \in OOB_k)}{\sum_{i=1}^{n} I(d_i \in OOB_k)} \tag{2-9}$$

其中

$$\widehat{j}(d_i, Y) = \arg\max_{j\neq y_i} Q(d, j) \tag{2-10}$$

$\widehat{j}(d_i, Y)$ 为除了真实类别之外的其他类别中得到最多投票的类别。

随机特征选取

随机特征选取，是指随机森林为了提高预测精度，引入随机性减小相关系数而保持强度不变，每颗决策树都使用一个从某固定概率分布产生的随机向量，可以使用多种方法将随机向量合并到树的生长过程。目前主要方法有：

随机选择输入变量（forest-RI）

随机组合输入变量（forest-RC）

随机选择输入变量（forest-RI）

- 在每一个节点随机选取F个输入变量进行分割，这样决策树的节点分割时根据这F个选定的特征，而不是考察所有的特征来决定。
- 在随机森林构建过程中选择的输入变量个数F是固定的。

随机选择输入变量（forest-RI）

- 随机森林的强度和相关性都依赖于F的大小，如果F足够小，树的相关性趋向于减弱；另一方面分类模型的强度随着F的增加而提高。
- 实验证明测试集误差对F取值的敏感性不是很大。F=1和F取更大值之间的平均绝对误差率小于1%

$$F = \log_2 M + 1$$

随机选择输入变量（forest-RI）

- 随机输入选择的计算速度远远快于**Adaboost**和**Bagging**。可以证明使用**forest-RI**的计算时间和所有变量构建的随机森林的计算时间比值为

$$F \times \log_2(N) / M$$

F是使用的变量个数，**N**是样本数，**M**是总的输入变量个数

随机组合输入变量（forest-RC）

用许多输入变量的线性组合来定义更多的随机特征来分割树，比如由L个变量线性组合作为一个输入特征。在一个给定的结点，生成F个线性组合，并从中选取最优的分割。

Pattern Recognition 46 (2013) 769–787

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Stratified sampling for feature subspace selection in random forests for high dimensional data

Yunming Ye [a,e,*], Qingyao Wu [a,e], Joshua Zhexue Huang [b,d], Michael K. Ng [c], Xutao Li [a,e]

[a] Department of Computer Science, Shenzhen Graduate School, Harbin Institute of Technology, China
[b] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[c] Department of Mathematics, Hong Kong Baptist University, China
[d] Shenzhen Key Laboratory of High Performance Data Mining, China
[e] Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen, China

随机森林的性能依赖于每一棵树以及各棵树之间的差异性。**Breiman**在文献中证明，随机森林的泛化误差由 $\bar{\rho}/s^2$ 决定。

对于高维数据来说，很大一部分的特征分类不能提供有效的信息。常见的随机森林选择的随机子空间中很少包含有用的信息，在这些子空间上建立的决策树可能会削弱树的强度，进而增加随机森林的误差。

本文提出一种分层抽样的方法来选择随机子空间。

将特征按照对分类提供信息量的大小分成两个子集，然后从两个子集中随机的抽取特征，这个方法可以保证在高维数据下每个随机子空间都包含足够多的有用信息。

分层特征子空间选择

一颗决策树的分类性能取决于选择的特征对类别特征的相关性，为了提高树的强度，需要选择那些跟类别特征具有强相关的特征作为特征子空间，但是对高维数据来说，往往只有很少的一部分特征跟类别特征具有强相关性。

Suppose we have only H features that are informative (i.e., highly correlated to the class feature) for classification purposes. The remaining $N-H$ features are then not informative. If a decision tree is grown in a subspace of p features, often with $p \ll N$, then the total number of possible subspaces is

$$\binom{N}{p} = \frac{N!}{(N-p)!p!}$$

The probability of selecting a subspace of $p > 1$ features without informative features is given by

$$\frac{\binom{N-H}{p}}{\binom{N}{p}} = \frac{\left(1-\frac{H}{N}\right)\cdots\left(1-\frac{H}{N}-\frac{p}{N}-\frac{1}{N}\right)}{\left(1-\frac{1}{N}\right)\cdots\left(1-\frac{p}{N}-\frac{1}{N}\right)} \approx \left(1-\frac{H}{N}\right)^{p}$$

在高维数据中，$N \gg H$，因此子空间中不包含有用信息的可能近似为1，也就是说，使用简单的随机子空间抽样方法，抽取的子空间不包含有用信息的概率很大。因此生成的决策树对分类没有帮助，进而使得树的平均强度下降。分层抽样的方法可以使产生的随机子空间一定包含有用的特征。

分层抽样方法

Let $\boldsymbol{A}$ be a set of N features $\{A_1, A_2, \ldots, A_N\}$ describing a space $\mathcal{S}$. Let Y be a feature identifying the classes of objects. We consider a non-negative function φ as a measure of the informativeness of an input feature A_i with respect to the class feature Y.

We use the Fisher discriminant projection in the optimal direction of the features for calculating informativeness φ_i for the feature A_i. The projection computes $s = \mathbf{w}^T\mathbf{x}$ to relate to a class y in terms of input features $\mathbf{x} = (x_1, \ldots, x_N)$, where $\mathbf{w} = (w_1, \ldots, w_N)$ refers to the weights of the projection. When the feature is important (not important), the value of the weight is large (small). Therefore we use the absolute value of weight w_i as the informativeness φ_i of the feature A_i [30].

The resulting value of φ_i is normalized as follows:

$$\theta_i = \frac{\varphi_i}{\sum_{k=1}^{N} \varphi_k}$$

where θ_i is then in between 0 and 1 and measures the relative informativeness of feature A_i with respect to the set of features $\boldsymbol{A}$. We call a feature A_i strong (weak) if θ_i is large (small). We can then stratify the set $\boldsymbol{A}$ into two groups as $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$ by the following procedure:

(i) Sort the set of features based on $\{\theta_i\}$ in descending order.
(ii) Specify a threshold α and divide $\boldsymbol{A}$ into two disjoint groups so that $\boldsymbol{A} = \boldsymbol{A}_s \cup \boldsymbol{A}_w$, $\boldsymbol{A}_s \cap \boldsymbol{A}_w = \emptyset$, and $A_s = \{A_i \in A \,|\, \theta_i < \alpha\}$ and $A_w = \{A_i \in A \,|\, \theta_i \geq \alpha\}$.

用分层抽样形成的p个特征的子空间，可以从2个子集中按子集大小比例抽取。用 N_s 表示强特征集合的大小，那么从强特征集中可以选择 $p_s = p \times N_s/N$ 个特征，从弱特征集中选 $p_w = p - p_s$ 个特征。

这种方法保证了在任一结点的子空间中都包含有强的和弱的特征。

对包含大量无用信息的高维数据来说，使用分层抽样方法比传统的随机森林可以提供更多的准确性。

子空间差异性分析

Let N_s and N_w be the numbers of features in $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$, respectively, $N_s+N_w=N$, where N is the number of features in $\boldsymbol{A}$. The possible number of the selections of p_s features from $\boldsymbol{A}_s$ is given by

$$C_s=\binom{N_s}{p_s}=\frac{N_s!}{(N_s-p_s)!p_s!}$$

The possible number of the selections of p_w features from $\boldsymbol{A}_w$ is given by

$$C_w=\binom{N-N_s}{p-p_s}=\frac{(N-N_s)!}{(N-N_s-p+p_s)!(p-p_s)!}$$

The subspace diversity C can be calculated by the total number of possible subspaces:

$$C = C_s \times C_w = \frac{N_s!}{(N_s - p_s)!p_s!} \times \frac{(N - N_s)!}{(N - N_s - p + p_s)!(p - p_s)!}$$

$$\approx \frac{(N_s)^{p_s}\left(1 - \frac{p_s}{N_s}\right)^{(p_s - 1)}}{p_s!} \times \frac{(N - N_s)^{(p - p_s)}\left(1 - \frac{p - p_s}{N - N_s}\right)^{(p - p_s - 1)}}{(p - p_s)!}$$

$$= \frac{(N_s)^{p_s}(N - N_s)^{(p - p_s)}}{p_s!(p - p_s)!}\left(1 - \frac{p_s}{N_s}\right)^{(p_s - 1)}\left(1 - \frac{p - p_s}{N - N_s}\right)^{(p - p_s - 1)}$$

If $N_s \ll N$, the diversity of subspaces can be represented as

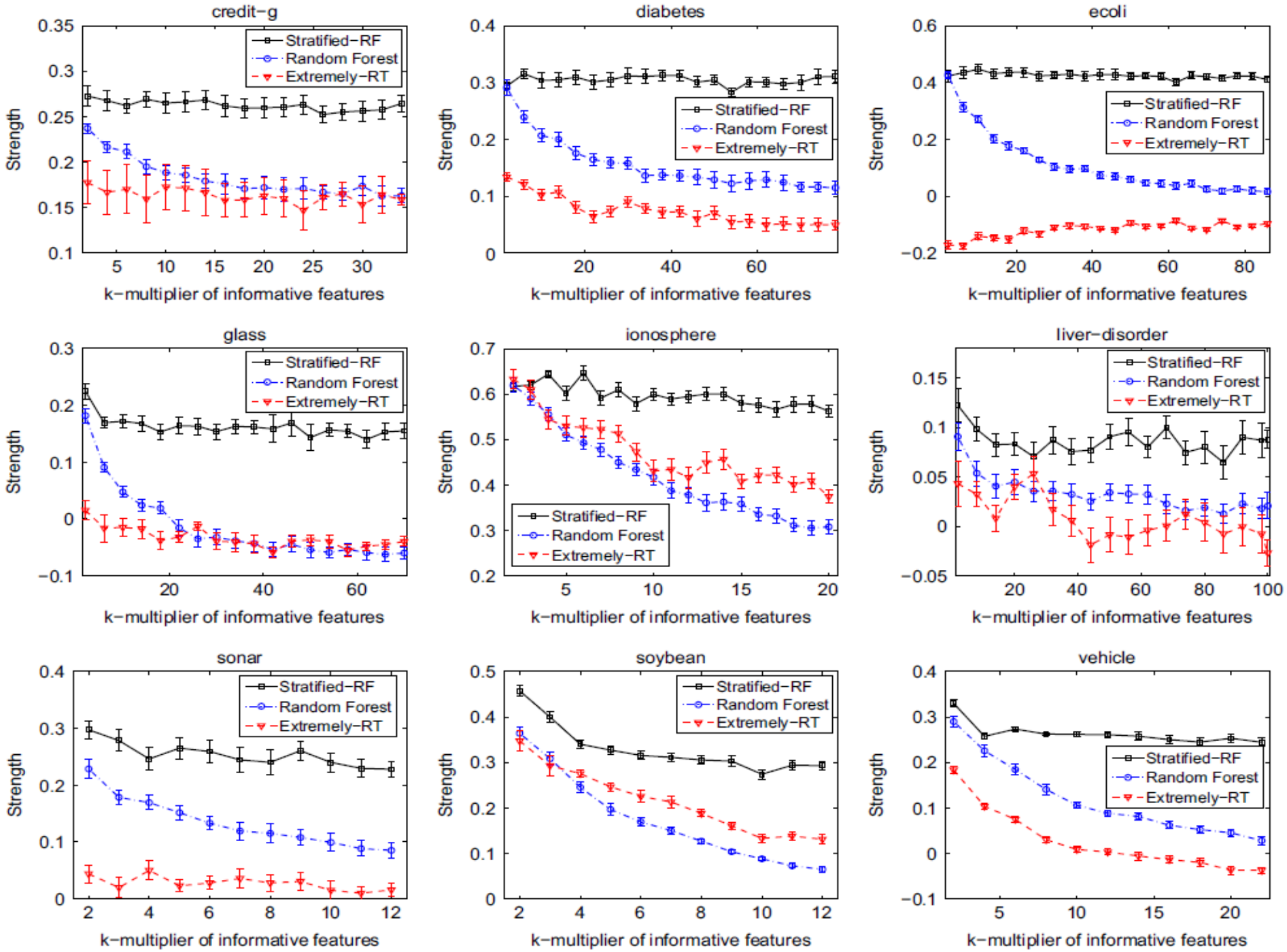
$$C \approx \frac{(N_s)^{p_s}(N)^{(p - p_s)}}{p_s!(p - p_s)!}\left(1 - \frac{p_s}{N_s}\right)^{(p_s - 1)}$$

This formula shows that the diversity of subspaces increases as p increases as p_s increases. The stratified sampling method is sufficient in subspace diversity. For example, suppose the total number of features $N=100$, the number of strong informative features $N_s=50$, and we sample a subspace of 10 features containing five strong features. There are over 4 billions possible subspaces. If we set the subspace size $p=\text{int}(\log_2(N)+1)=7$ as suggested in [2], where $\text{int}(x)$ is the first integer larger than x, we will also have over 300 millions of different subspaces.

The SRF algorithm

1. For each feature A_i, compute its informativeness φ_i with an non-negative informative function φ, and normalize the resulting value to θ_i according to (1).
2. Specify a stratification threshold α to divide $\boldsymbol{A}$ into two groups $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$.
3. Use bagging [22] to generate K subsets $\{X_1, X_2, \ldots, X_K\}$.
4. Grow a decision tree $h_i(X_i)$ for each data set X_i. At each node, randomly sample a feature subspace with $p(>1)$ features from $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$ proportionally. Use a Boolean test function τ on p features to divide the data into left and right children nodes. Continue this process until a stopping criteria are met: all data are pure or have identical value for each attribute, or the number of instances is less than n_{min}.
5. Combine the K unpruned trees $h_1(X_1)$, $h_2(X_2)$, …, $h_K(X_K)$ into a random forest ensemble and use voting to make the final classification decision.

Each tree is now built recursively in a top-down manner. We start building each tree from the training set. At each node, a feature subspace is randomly selected. To split the data we use a Boolean test on $\mathbf{w}^T\mathbf{x} \leq \tau$ or $\mathbf{w}^T\mathbf{x} > \tau$ where τ is the average value of means of the projected samples with respect to different class subsets. That is, $\tau = (1/C)\sum_{i=1}^{C} \tilde{m}_i$ where C is the number classes and $\tilde{m}_i$ is the projected mean of samples labeled by class c_i.


credit-g
diabetes
ecoli
glass
ionosphere
liver-disorder
sonar
soybean
vehicle
Stratified-RF
Random Forest
Extremely-RT
Strength
k-multiplier of informative features

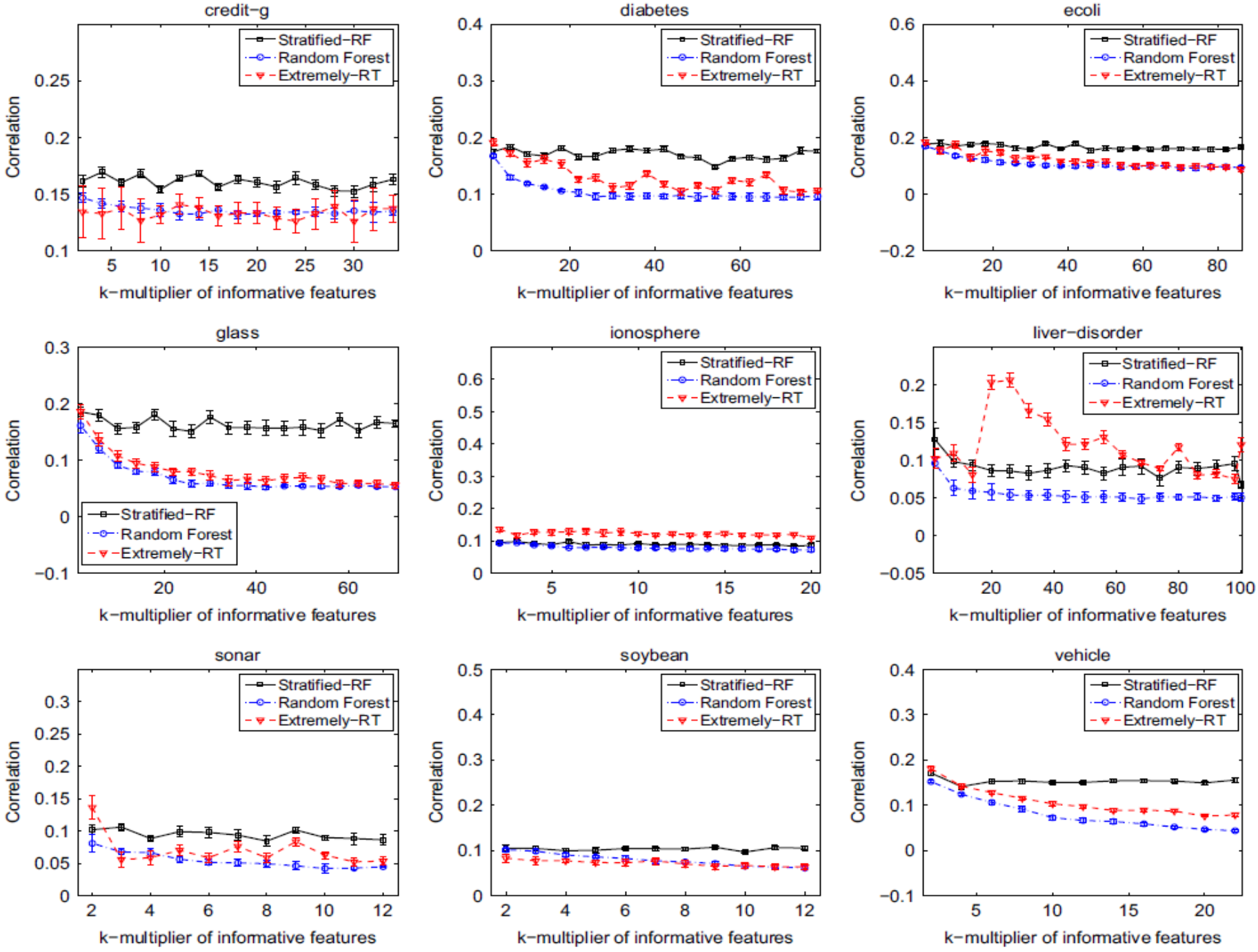


Fig. 2. Correlation changes against the number of non-informative features on the nine synthetic data sets.

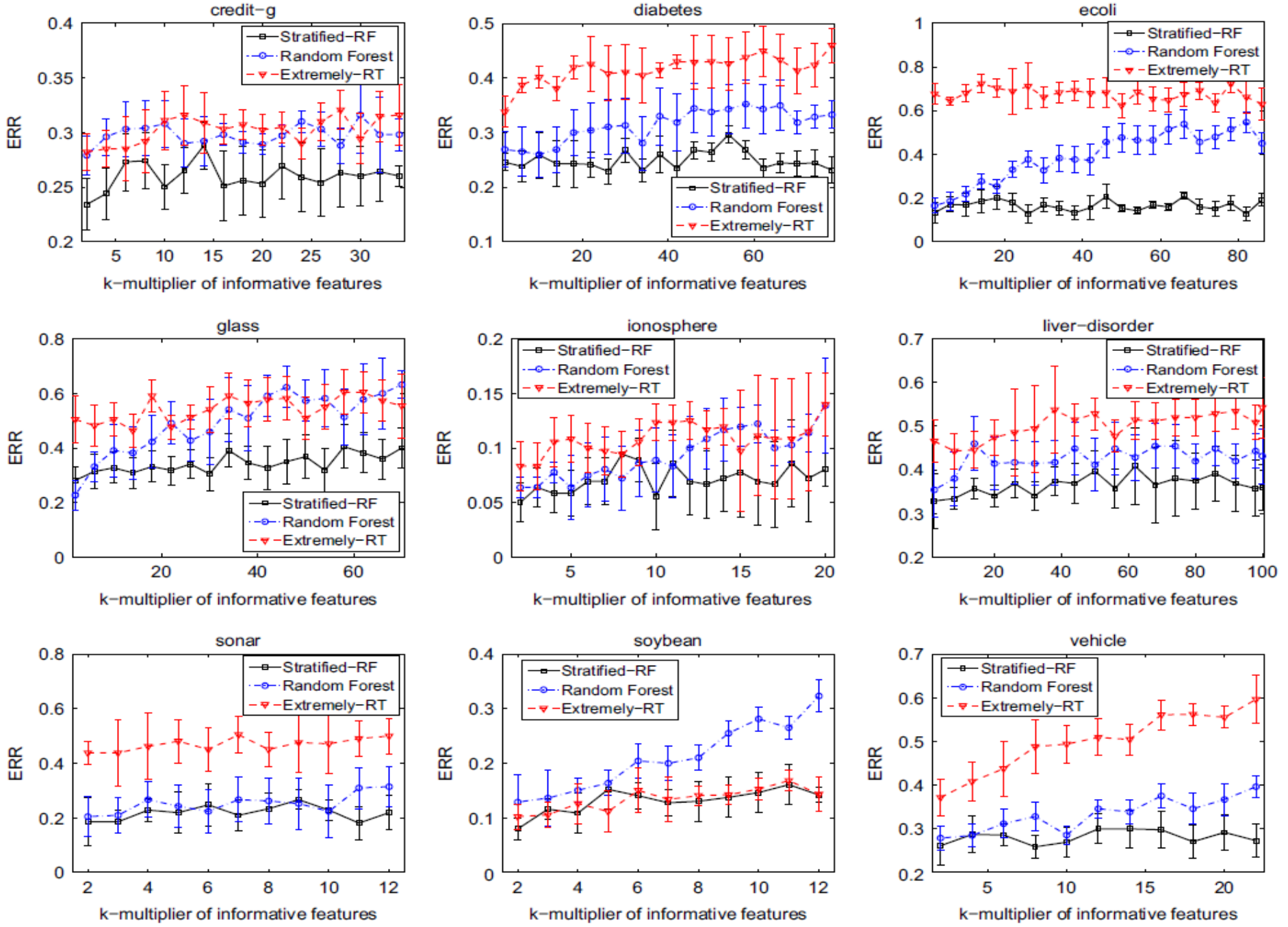


Fig. 3. Testing error (mean and std-dev) changes against the number of non-informative features on the nine synthetic data sets.