两种常用的R语言分类包
——SVM和xgboost

- 组长：彭巍
- 组员：欧阳添荣 王蒙 陈煜斌

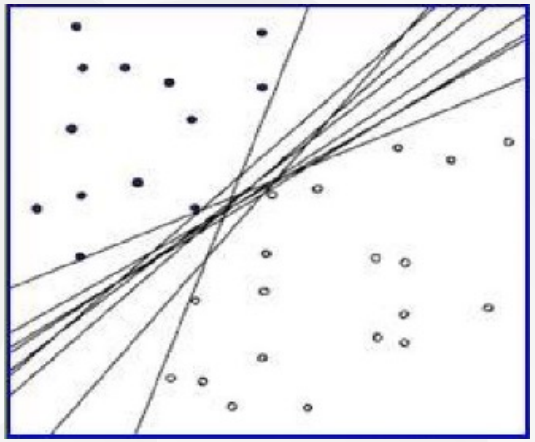
SVM的原理

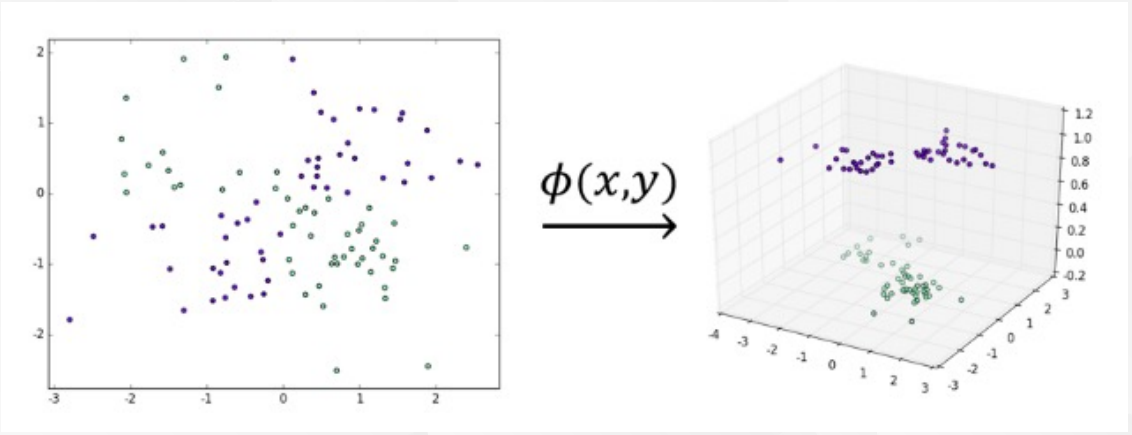
目录

SVM简介

SVM简介

支持向量机（support vector machines, SVM）是一种二分类模型，它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别于感知机；SVM还包括核技巧，这使它成为实质上的非线性分类器。SVM的的学习策略就是间隔最大化，可形式化为一个求解凸二次规划的问题，也等价于正则化的合页损失函数的最小化问题。SVM的的学习算法就是求解凸二次规划的最优化算法。





SVM算法原理

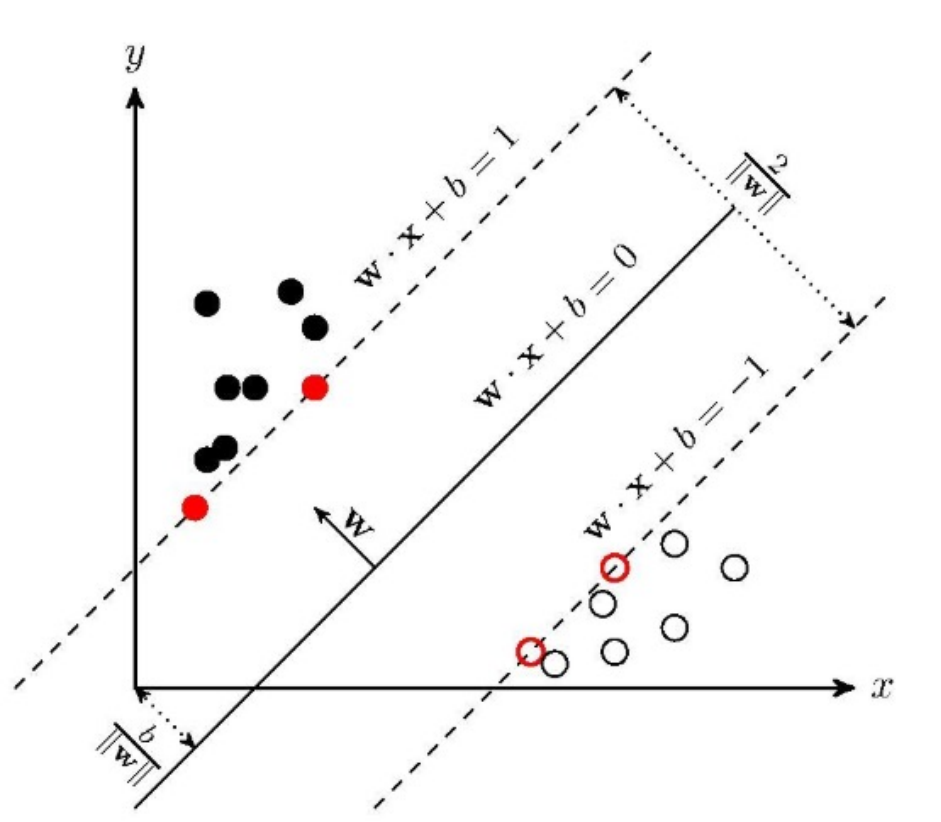
SVM原理

原理的思想是求解能够正确划分训练数据集并且几何间隔最大的分离超平面。

在推导之前，先给出一些定义。假设给定一个特征空间上的训练数据集。

$T = \{(x1, y1), (x2, y2), \cdots\cdots, (xN, yN$

xi 为i个特征向量; y i 为类标记，当它等于+1时为正例；为-1时为负例。再假设训练数据集是线性可分的。



用数学表达式表示就是

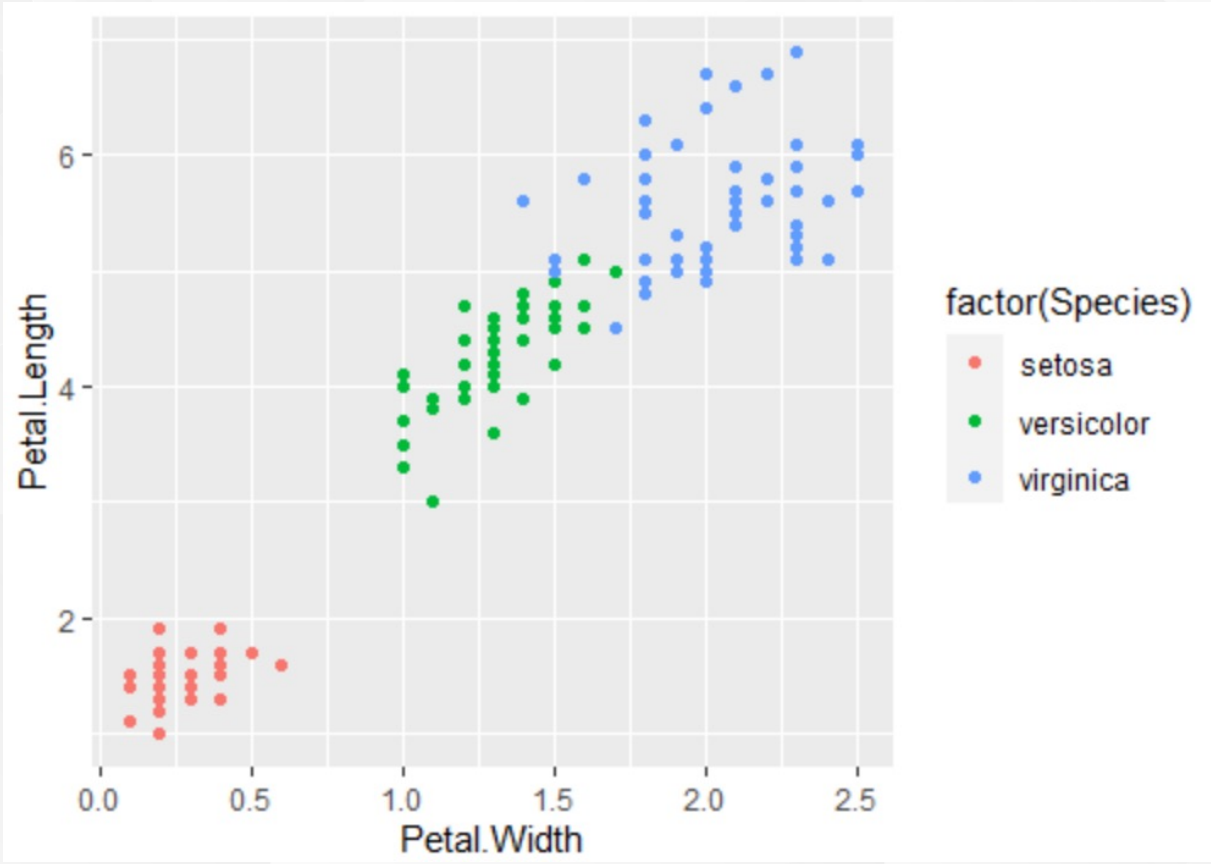
$$\max margin(w,b) = \max \min_{1\le i\le n} distance(w,b,x_i)$$
$$= max \min_{1\le i\le n,} \frac{1}{||w||}(w^T x_i + b)$$
$$st.(w^T x_i + b) > 0, y_i = +1, for\ \forall i = 1 \dots n$$
$$st.(w^T x_i + b) < 0, y_i = -1, for\ \forall i = 1 \dots n$$

SVM应用实例

数据集

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width		Species
1	5.1	3.5	1.4	0.2	1	setosa
2	4.9	3.0	1.4	0.2	2	setosa
3	4.7	3.2	1.3	0.2	3	setosa
4	4.6	3.1	1.5	0.2	4	setosa
5	5.0	3.6	1.4	0.2	5	setosa
6	5.4	3.9	1.7	0.4	6	setosa
7	4.6	3.4	1.4	0.3	7	setosa
8	5.0	3.4	1.5	0.2	8	setosa
9	4.4	2.9	1.4	0.2	9	setosa
10	4.9	3.1	1.5	0.1	10	setosa
11	5.4	3.7	1.5	0.2	11	setosa
12	4.8	3.4	1.6	0.2	12	setosa
13	4.8	3.0	1.4	0.1	13	setosa
14	4.3	3.0	1.1	0.1	14	setosa
15	5.8	4.0	1.2	0.2	15	setosa
16	5.7	4.4	1.5	0.4	16	setosa
17	5.4	3.9	1.3	0.4	17	setosa
18	5.1	3.5	1.4	0.3	18	setosa
19	5.7	3.8	1.7	0.3	19	setosa
20	5.1	3.8	1.5	0.3	20	setosa

由ggplot2得到的散点图



SVM函数（e1071）

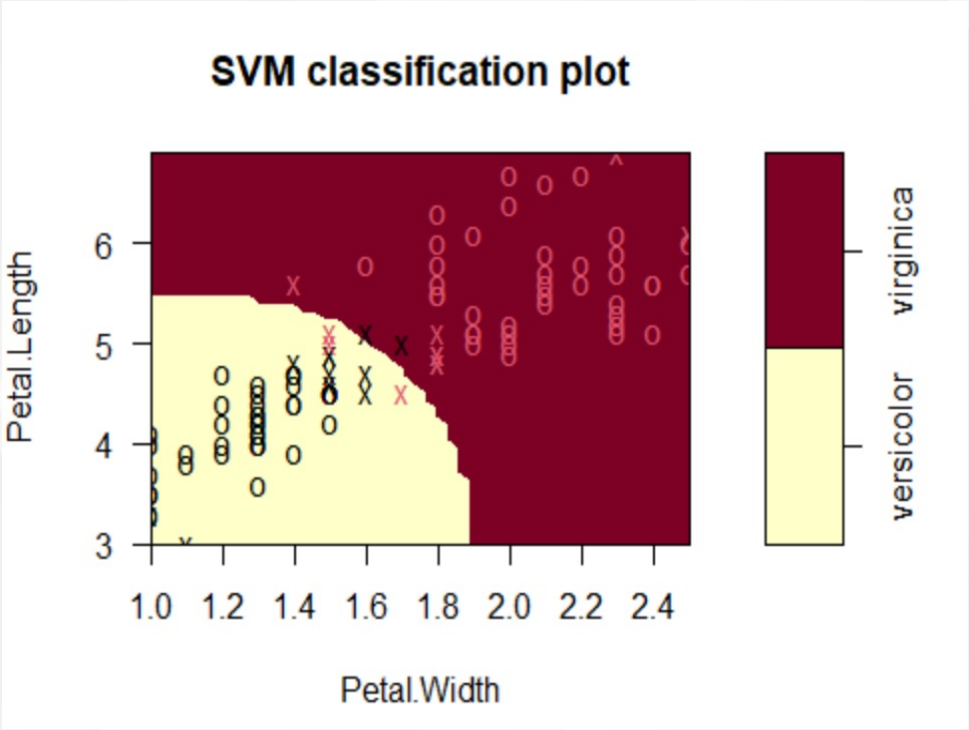
svm函数的重要参数：

参数	意义
formula	y~$x_1 + 1_2 + ... + x_n$，确定自变量和因变量
data	使用的数据集
type	分类：C-classification(default)/nu-classification；文本分类：one-classification；回归：eps-regression(default)/nu-regression
kernel	linear/polynomial/radial/sigmoid

具体实现过程：

```
install.packages("e1071")
library("e1071")
library("ggplot2")
ggplot(iris,aes(x=Petal.Width,y=Petal.Length))+geom
_point(aes(color=factor(Species)))

subdata<-iris[iris$Species !="setosa",]
subdata
subdata$Species <- factor(subdata$Species)
subdata
model1 <- svm(Species ~ Petal.Length + Petal.Width,
data = subdata)
plot(model1, subdata, Petal.Length ~ Petal.Width)
summary(model1)
```



Xgboost原理

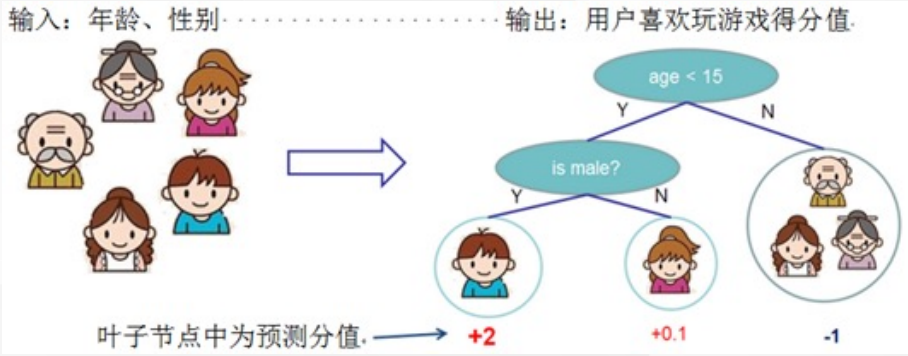
引言

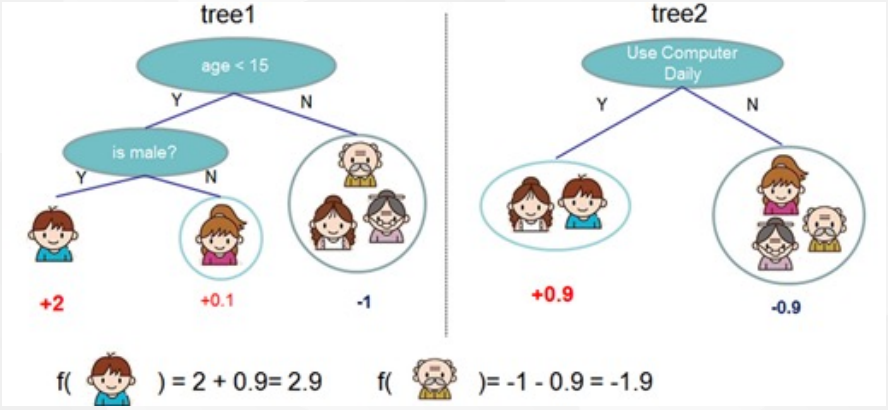
北京大学 PEKING UNIVERSITY

XGBoost全名叫（eXtreme Gradient Boosting）极端梯度提升，是现阶段许多竞赛最常用的工具之一，其效果显著。它是大规模并行boosted tree的工具，它是目前最快最好的开源boosted tree工具包。下面我们将XGBoost的介绍分为3步：①集成思想 ②目标函数 ③算法，对他们进行简单介绍。

集成思想

在学习XGBoost之前，我们得需要先明白集成思想。集成学习方法是指将多个学习模型组合，以获得更好的效果，使组合后的模型具有更强的泛化能力。另外XGBoost是以分类回归树(CART树)进行组合。故在此之前，我们先看下CART树。如下，通过输入用户年龄、性别进行判断用户是否喜欢玩游戏的得分值。由此得到一颗CART树模型。





我们知道对于单个的决策树模型容易出现过拟合，并且不能在实际中有效应用。所以出现了集成学习方法。如下图，通过两棵树组合进行玩游戏得分值预测。其中tree1中对小男生的预测分值为2，tree2对小男生的预测分值为0.9。则该小男生的最后得分值为2.9。

02 目标函数

$$Obj(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

其中为$l(y_i, \hat{y}_i)$样本的训练误差，$\Omega(f_k)$表示第k棵树的正则项，表示树的复杂度的函数，值越小复杂度越低，泛化能力越强。

03 核心算法思想

XGBoost的核心算法思想，基本就是：
不断地添加树，不断地进行特征分裂来生长一棵树，每次添加一个树，其实是学习一个新函数f(x)，去拟合上次预测的残差。
当我们训练完成得到k棵树，我们要预测一个样本的分数，其实就是根据这个样本的特征，在每棵树中会落到对应的一个叶子节点，每个叶子节点就对应一个分数
最后只需要将每棵树对应的分数加起来就是该样本的预测值。

- Start from constant prediction, add a new function each time

$$
\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\cdots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned}
$$

New function

Model at training round t

Keep functions added in previous round

我们如何选择每一轮加入什么 f 呢？答案是非常直接的，选取一个 f 来使得我们的目标函数尽量最大地降低。

北京大学
PEKING UNIVERSITY

xgboost分类案例——红酒质量分类

北京大学
PEKING UNIVERSITY

01

PART ONE

案例简介

01/ 案例简介

北京大学 PEKING UNIVERSITY

数据内容

红酒质量分类数据集

包括非挥发性酸性、挥发性酸性、柠檬酸、剩余糖分、氯化物、游离二氧化硫、二氧化硫总量、浓度、pH、硫酸盐、酒精、质量等十二个属性。共1599条记录

分类目标

选取70%的数据为训练集，30%的数据为测试集

根据数据集中的前十一个属性，将测试集中所有的记录分为“低质量”和“高质量”两类。

北京大学
PEKING UNIVERSITY

02

PART TWO

数据读取与处理

02 数据读取与处理

北京大学 PEKING UNIVERSITY

```
install.packages("openxlsx")
library(openxlsx)
wine = read.xlsx( “winequality-red.xlsx” ) #读取数据

train_sub = sample(nrow(wine),7/10*nrow(wine))
train_data = wine[train_sub,]
test_data = wine[-train_sub,] #将数据集分为训练集和测试集，比例为7:3
```

北京大学
PEKING UNIVERSITY

03

PART THREE

Xgboost实现

03 xgboost实现

北京大学 PEKING UNIVERSITY

训练集的数据预处理

```
library(Matrix)
# 将自变量转化为矩阵
traindata1 <- data.matrix(train_data[,c(1:11)])
# 利用Matrix函数，将sparse参数设置为TRUE，转化为稀疏矩阵
traindata2 <- Matrix(traindata1,sparse=T)#自变量
 traindata3 <- train_data[,13]#因变量
# 将自变量和因变量拼接为list
traindata4 <- list(data=traindata2,label=traindata3)
 # 构造模型需要的xgb.DMatrix对象，处理对象为稀疏矩阵
dtrain <- xgb.DMatrix(data = traindata4$data, label =
traindata4$label)
```

03 xgboost实现

北京大学 PEKING UNIVERSITY

测试集的数据预处理

```
# 将自变量转化为矩阵
testset1 <- data.matrix(test_data[,c(1:11)])
# 利用Matrix函数，将sparse参数设置为TRUE，转化为稀疏矩阵
testset2 <- Matrix(testset1,sparse=T)
# 将因变量转化为numerictestset3 <- test_data[,13]
# 将自变量和因变量拼接为list
testset4 <- list(data=testset2,label=testset3)
# 构造模型需要的xgb.DMatrix对象，处理对象为稀疏矩阵
dtest <- xgb.DMatrix(data = testset4$data, label = testset4$label)
```

03 xgboost实现

北京大学 PEKING UNIVERSITY

得到测试集的分类结果

```
xgb <- xgboost(data = dtrain,max_depth=6, eta=0.5,
objective='binary:logistic', nround=25)

pre_xgb = round(predict(xgb,newdata = dtest))
```

pre_xgb 是一个长度等于测试集中的记录个数的0—1向量，第i个分量为0代表第i个记录为“低质量”，第i个分量为1代表第i个记录为“高质量”

03 xgboost实现

北京大学 PEKING UNIVERSITY

```
> pre_xgb
  [1] 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 1 1 0 0 0 1 0 0
 [59] 0 0 1 1 1 0 0 0 0 1 1 1 1 0 0 0 1 0 0 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1
[117] 0 1 1 0 1 0 1 0 0 0 1 1 0 1 0 0 1 0 1 0 1 1 1 0 1 1 0 1 1 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 1 1 1 0 1 0 0 1 1 1 0 0
[175] 0 0 1 1 0 0 1 0 1 1 0 1 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1
[233] 1 0 0 1 1 0 1 0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 0 0 1 0 1 0 0 1 0 1 0 1 1 1 1 1 1 0 1 1 0 0 1 0 0 0 1
[291] 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1
[349] 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 1 0 1 0 0 1 0 0 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 0 0 1
[407] 0 1 1 0 0 0 1 1 0 0 1 1 0 0 0 1 1 0 1 0 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0 1 0 0 1 0 1 0 0 1 0 0 1 1 1 1 0 1 0 1 1 1 1
[465] 0 0 0 0 0 0 1 1 0 1 1 0 0 1 1 1
```

在测试集上的分类结果

请老师批评指正