**ЗАДАНИЕ ДР1 - Решение задачи классификации методом построения деревьев решений**

Выполнил студент 2 курса

Санамян Артак Размикович

группы 09-715(1)

**Текст задания:**

1. Примените метод деревьев решений для задачи классификации (для использованных ранее или новых данных).

2. Если число признаков m = 2, визуализуйте данные (постройте облако точек).

3. Исследуйте дерево решений; если позволяет размерность, постройте его график.

4. Проанализируйте точность полученных решений для тестовых данных (с известным значением переменной отклика), сравните результаты с ранее полученными.

5. Задайте нескольких новых данных, покажите соответствующие точки на графике (при m = 2) (выделите их другим цветом).

6. Определите класс для  новых данных (т.е. для данных, где известны только значения факторов, но не переменной отклика).

7. Проанализируйте полученные результаты, оформите отчёт.

Источник числовых данных, использованный для выполнения задания: <http://archive.ics.uci.edu/ml/datasets/Flags>

**Описание выполненной работы:**

Данные представляют собой информацию о странах на 1990 год. Необходимо определить религию страны:

0 – Католики;

1 – Остальные христиане;

2 – Мусульмане;

3 – Буддисты;

4 – Индуисты;

5 – Язычники;

6 – Марксисты;

7 – Другие.

Факторами являются следующие признаки:

1. Название страны (не числовой);
2. Материк (1-6);
3. Зона: географический квадрант, основанный на Гринвиче и экваторе(1-4)
4. Площадь: в тысячах квадратных километров;
5. Население: в миллионах;
6. Язык: 1 = английский, 2 = испанский, 3 = французский, 4 = немецкий, 5 = славянский, 6 = другой индоевропейский, 7 = китайский, 8 = арабский, 9 = японский / турецкий / финский / мадьяр, 10 = Другие;
7. Религия: 0 = католик, 1 = другой христианин, 2 = мусульманин, 3 = буддист, 4 = индус, 5 = этнический, 6 = марксист, 7 = другие;
8. Полосы: количество вертикальных полос на флаге;
9. Полосы: количество горизонтальных полос на флаге;
10. Цвета: количество различных цветов на флаге;
11. Красный: 0 – нет, 1 – цвет есть на флаге;
12. Зеленый: 0 – нет, 1 – цвет есть на флаге;
13. Синий: 0 – нет, 1 – цвет есть на флаге;
14. Золотой: 0 – нет, 1 – цвет есть на флаге (также желтый);
15. Белый: 0 – нет, 1 – цвет есть на флаге;
16. Черный: 0 – нет, 1 – цвет есть на флаге;
17. Оранжевый: 0 – нет, 1 – цвет есть на флаге (также коричневый);
18. Основной оттенок (не числовой);
19. Количество кругов;
20. Количество крестов;
21. Количество Андреевских крестов
22. Количество частей флага;
23. Количество символов солнца или звезд;
24. Полумесяц: 1, если присутствует символ полумесяца, иначе 0
25. Треугольник: 1, если присутствуют какие-либо треугольники, 0 в противном случае
26. 1, если присутствует неодушевленное изображение (например, лодка), в противном случае 0
27. 1, если присутствует одушевленное изображение (например, орел, дерево, человеческая рука), 0 в противном случае
28. Текст: 1 если какие-либо буквы или письма на флаге (например, девиз или слоган), 0 в противном случае
29. Цвет в верхнем левом углу;
30. Цвет в левом нижнем углу.

Подсчитаем частоты встречаемости классов в исходной выборке:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 20.6 | 30.9 | 18.6 | 4.1 | 2.1 | 13.9 | 7.7 | 2.1 |

Подсчитаем частоту присутствия каждого класса в обучающей выборке:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 18.2 | 34.5 | 15.5 | 4.5 | 3.6 | 16.4 | 5.5 | 1.8 |

Репрезентативность выборки обеспечена путем ее перемешивания.

Рассмотрим кросс-валидационную таблицу:

Cell Contents

|-------------------------|

| N |

| N / Row Total |

| N / Col Total |

| N / Table Total |

|-------------------------|

Total Observations in Table: 84

| flag\_pred

test\_data\_labels | 0 | 1 | 2 | 5 | Row Total |

-----------------|-----------|-----------|-----------|-----------|-----------|

0 | 17 | 2 | 1 | 0 | 20 |

| 0.850 | 0.100 | 0.050 | 0.000 | 0.238 |

| 0.850 | 0.062 | 0.042 | 0.000 | |

| 0.202 | 0.024 | 0.012 | 0.000 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

1 | 1 | 17 | 3 | 1 | 22 |

| 0.045 | 0.773 | 0.136 | 0.045 | 0.262 |

| 0.050 | 0.531 | 0.125 | 0.125 | |

| 0.012 | 0.202 | 0.036 | 0.012 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

2 | 0 | 2 | 15 | 2 | 19 |

| 0.000 | 0.105 | 0.789 | 0.105 | 0.226 |

| 0.000 | 0.062 | 0.625 | 0.250 | |

| 0.000 | 0.024 | 0.179 | 0.024 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

3 | 0 | 2 | 1 | 0 | 3 |

| 0.000 | 0.667 | 0.333 | 0.000 | 0.036 |

| 0.000 | 0.062 | 0.042 | 0.000 | |

| 0.000 | 0.024 | 0.012 | 0.000 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

5 | 0 | 3 | 1 | 5 | 9 |

| 0.000 | 0.333 | 0.111 | 0.556 | 0.107 |

| 0.000 | 0.094 | 0.042 | 0.625 | |

| 0.000 | 0.036 | 0.012 | 0.060 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

6 | 2 | 5 | 2 | 0 | 9 |

| 0.222 | 0.556 | 0.222 | 0.000 | 0.107 |

| 0.100 | 0.156 | 0.083 | 0.000 | |

| 0.024 | 0.060 | 0.024 | 0.000 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

7 | 0 | 1 | 1 | 0 | 2 |

| 0.000 | 0.500 | 0.500 | 0.000 | 0.024 |

| 0.000 | 0.031 | 0.042 | 0.000 | |

| 0.000 | 0.012 | 0.012 | 0.000 | |

-----------------|-----------|-----------|-----------|-----------|-----------|

Column Total | 20 | 32 | 24 | 8 | 84 |

| 0.238 | 0.381 | 0.286 | 0.095 | |

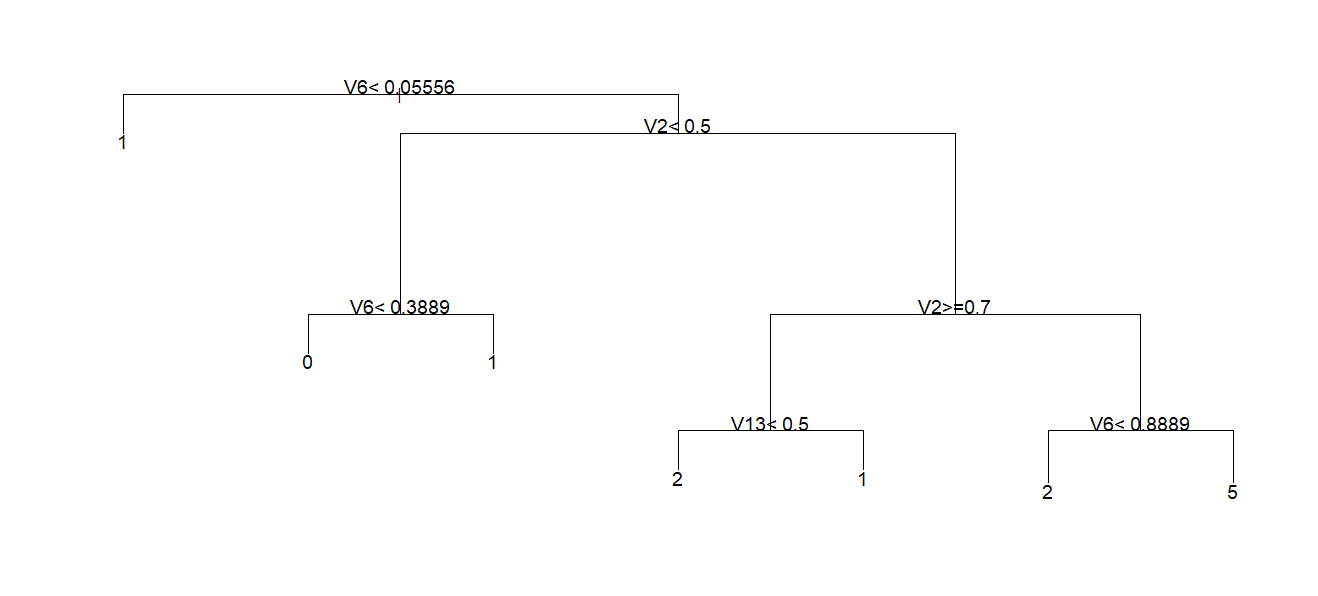
-----------------|-----------|-----------|-----------|-----------|-----------|

Как видно из кросс - валидационной таблицы, метод деревьев решений допустил 29 ошибки из 84:

Процент всех ошибок = 34, 52.

Таким образом, точность метода составляет 65, 48%.

Построим дерево решений:



Зададим несколько новых данных, для которых известно значение класса, и определим значение переменной отклика (номер класса) для новых данных:

> new\_predict

1 2 3 4 5 6 7 8 9 10 11 12

1 2 1 2 2 1 2 1 2 1 2 2

Levels: 0 1 2 3 4 5 6 7

Правильные классы: 1 0 1 2 2 1 0 1 5 1 3 0

5 из 12 данных определены неверно.

Сравним полученные результаты с результатами, полученными для той же выборки методом KNN и наивным Байесовским классификатором. Самым точным методом оказался метод Деревьев решений - 65, 48%.

**Код программы:**

setwd("C://Users/pc/Documents/Магистратура/2 семестр/R/task5 ДР")

getwd()

#очищаем рабочее пространство

rm (list=ls())

# Прочитаем файл с данными в переменную flagdata

flagdata <- read.csv("flagData.txt", head = FALSE, sep=",") #, sep="\t"

# str(flagdata) # структура данных

flagdata

# Формула нормализации данных

normalize <- function(x) {return ((x - min(x)) / (max(x) - min(x)))}

# Нормализуем данные

# исключим не численные факторы и переменную отклика

flag <- as.data.frame(lapply(flagdata[c(2:6, 8:17, 19:28)], normalize))

dims = dim(flag) # n \* m

n = dims[1] # объём выборки # 194

m = dims[2] # количество факторов # 25

# добавим перменную отклика в конец для удобства

flag[m+1] = flagdata[7]

flag

# Добавим названия столбцов

# colnames(flag)<- c('V2','V3','V4','V5','V6','V8','V9','V10','V11','V12','V13','V14','V15','V16','V17','V19','V20','V21','V22','V23','V24','V25','V26','V27','V28','Type')

# Прочитали 194 записей, в каждой - 26 значений, где

# последняя (26-я) переменная - номер класса (переменная отклика)

# Часть выборки, например, 110 записей, будем использовать как

# обучающую

# Сохраним объём всей выборки и объём обучающей выборки в переменных

n\_data = n

n\_train = 110

# Обучающая выборка должна быть репрезентативной -

# т.е. каждый класс должен быть представлен в ней в той же пропорции,

# что и во всей выборке.

# Подсчитаем частоты встречаемости классов в исходной выборке

# (выразим эти частоты в %, округлив результат до десятых)

round(prop.table(table(flag[m+1]))\*100, digits = 1) # априорные вероятности гипотез (классов) (p(Hk))

# Чтобы обеспечить репрезентативность выборки, перемешаем её

set.seed(12345)

flagdata\_mixed=flag[order(runif(n\_data)),]

# Выберем обучающую выборку

train\_data = flagdata\_mixed[1:n\_train,]

# Сохраним номера классов для строк обучающей выборки

train\_data\_labels = train\_data[,m+1]

# формирование меток соответствующих уровней значения переменной отклика

train\_data\_labels<-factor(train\_data\_labels)

# Подсчитаем частоту присутствия каждого класса в обучающей выборке

# и сравним с соответствующими частотами в исходной выборке

round(prop.table(table(train\_data[m+1]))\*100, digits = 1)

# Оставшуюся часть "перемешанной выборки" будем использовать как тестовую выборку

test\_data = flagdata\_mixed[(n\_train+1):n\_data, ]

test\_data\_labels = test\_data[,m+1]

# Подключим пакет rpart

library("rpart")

# построим дерево

my\_tree <- rpart(V7 ~ V2+V3+V4+V5+V6+V8+V9+V10+V11+V12+V13+V14+V15+V16+V17+V19+V20+V21+V22+V23+V24+V25+V26+V27+V28, method="class",data=train\_data)

# Построим график дерева

plot(my\_tree)

text(my\_tree, cex = 1.2, all = FALSE, use.n = FALSE)

# Здесь:

# my\_tree - дерево решений

# cex (character extension) - размер символов

# summary(my\_tree)

# Можем распечатать совокупность правил дерева,

# но они довольно громоздки

# Протестируем построенное дерево

# Теперь, когда дерево уже построено,

# номера классов (m+1-й столбец) нужно удалить

test\_data <- test\_data[-(m+1)]

# "Спрогнозируем" значение переменной отклика

# для тестовых данных

flag\_pred <- predict(my\_tree,

test\_data,

type="class")

# Для оценки качества прогноза

# подключим библиотеку gmodels

library("gmodels")

# Построим кросс-валидационную таблицу:

CrossTable(x = test\_data\_labels, y = flag\_pred, prop.chisq=FALSE)

# Зададим новые данные

predict <- read.csv("predict.txt", sep=",", head=FALSE)

predict = predict[c(2:6, 8:17, 19:28)]

# predict = as.data.frame(lapply(predict[c(2:6, 8:17, 19:28)], normalize))

# Добавим названия столбцов

# colnames(predict)<- c('V2','V3','V4','V5','V6','V8','V9','V10','V11','V12','V13','V14','V15','V16','V17','V19','V20','V21','V22','V23','V24','V25','V26','V27','V28','Type')

# Определим значение переменной отклика (номер класса) для новых данных

new\_predict <- predict(my\_tree, predict, type="class")

new\_predict