Лабораторная работа №1

Наивный Байесовский классификатор

Выполнил

студент гр. 33504/2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лелюхин Д. О.

Руководитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Селин И. А.

**Формулировка Задания**

1. Исследуйте, как объем обучающей выборки и количество тестовых данных, влияет на точность классификации или на вероятность ошибочной классификации в примере крестики-нолики и примере о спаме e-mail сообщений.

2. Сгенерируйте 100 точек с двумя признаками X1 и X2 в соответствии с нормальным распределением так, что первые 50 точек (class -1) имеют параметры: мат. ожидание X1 равно 10, мат. ожидание X2 равно 14, среднеквадратические отклонения для обеих переменных равны 4. Вторые 50 точек (class +1) имеют параметры: мат. ожидание X1 равно 20, мат. ожидание X2 равно 18, среднеквадратические отклонения для обеих переменных равны 3. Построить соответствующие диаграммы, иллюстрирующие данные. Построить байесовский классификатор и оценить качество классификации.

### Выполнение 1-ого задания:

#### Код программы на языке R:

#-------------------1------------------------

library(e1071)

# импортируем данные в R

# установить параметр stringsAsFactors = TRUE,

# так как все данные - категориальные

A\_raw <- read.table("Tic\_tac\_toe.txt", sep = ",", stringsAsFactors = TRUE)

# число строк в базе

n <- dim(A\_raw)[1]

# Создан фрейм, который можно просмотреть, используя str(A\_raw).

# Имеется 9 столбцов признаков V1-V9 и V10 (класс) и

# все имеют один и тот же тип Factor.

# 2 ###########################################################

# Создание обучающей и тестирующей выборки

# Скажем, имеем n примеров в исходной выборке,

# используем 80% для обучения и оставшиеся - для тестирования.

# Устанавливаем базу генерации случайных чисел и рандомизируем выборку

set.seed(12345)

A\_rand <- A\_raw[ order(runif(n)), ]

# разделим данные на обучающие и тестирующие

for (i in seq(0, 1, by = 0.05))

{

nt <- as.integer(n\*i)

A\_train <- A\_rand[1:nt, ]

A\_test <- A\_rand[(nt+1):n, ]

# Можно убедиться, какой имеется процент каждого

# класса V2 в обучающей и тестирующей выборке

prop.table(table(A\_train$V10))

prop.table(table(A\_test$V10))

# 3 ############################################################

# Используем Наивный Байесовский классификатор из пакета e1071

# A\_classifier <- naiveBayes(A\_train[,-10], A\_train$V10)

# Другой вариант классификатора

A\_classifier <- naiveBayes(V10 ~ ., data = A\_train)

# 4 ############################################################

# Теперь оценим полученную модель:

A\_predicted <- predict(A\_classifier, A\_test)

# Используем table для сравнения прогнозируемых значений с тем, что есть

t <- table(A\_predicted, A\_test$V10)

print(t)

}

#### Результаты работы программы:

A\_predicted negative positive

negative 0 0

positive 332 626

A\_predicted negative positive

negative 117 117

positive 198 479

A\_predicted negative positive

negative 133 104

positive 163 463

A\_predicted negative positive

negative 137 117

positive 138 423

A\_predicted negative positive

negative 119 104

positive 139 405

A\_predicted negative positive

negative 99 64

positive 149 407

A\_predicted negative positive

negative 96 73

positive 132 370

A\_predicted negative positive

negative 88 68

positive 118 349

A\_predicted negative positive

negative 80 63

positive 115 317

A\_predicted negative positive

negative 71 59

positive 105 292

A\_predicted negative positive

negative 66 51

positive 91 271

A\_predicted negative positive

negative 62 51

positive 79 240

A\_predicted negative positive

negative 50 43

positive 74 217

A\_predicted negative positive

negative 44 39

positive 62 191

A\_predicted negative positive

negative 34 38

positive 53 163

A\_predicted negative positive

negative 28 28

positive 44 140

A\_predicted negative positive

negative 22 20

positive 40 110

A\_predicted negative positive

negative 17 13

positive 30 84

A\_predicted negative positive

negative 13 7

positive 18 58

A\_predicted negative positive

negative 8 7

positive 7 26

A\_predicted negative positive

negative 0 0

positive 0 1

- - - - - -

#### Код программы на языке R:

library(kernlab)

library(e1071)

data(spam)

for(size in seq(20, 4581, by = 500))

{

idx <- sample(1:dim(spam)[1], size)

spamtrain <- spam[-idx, ]

spamtest <- spam[idx, ]

model <- naiveBayes(type ~ ., data = spamtrain)

t <- table(predict(model, spamtest), spamtest$type)

print(t)

}

#### Результаты работы программы:

nonspam spam

nonspam 9 0

spam 9 2

nonspam spam

nonspam 191 11

spam 134 184

nonspam spam

nonspam 336 21

spam 286 377

nonspam spam

nonspam 531 31

spam 373 585

nonspam spam

nonspam 660 33

spam 558 769

nonspam spam

nonspam 825 55

spam 687 953

nonspam spam

nonspam 1038 64

spam 799 1119

nonspam spam

nonspam 1070 55

spam 1070 1325

nonspam spam

nonspam 1155 69

spam 1281 1515

nonspam spam

nonspam 769 177

spam 1976 1598

## Вывод

В результате, мы получаем что, при уменьшении объема обучающей выборки уменьшается количество угаданных результатов.

## Выполнение 2-ого задания:

#### Код программы на языке R:

library(e1071)

x1\_1 <- rnorm(50, mean = 10, sd = 4)

x1\_2 <- rnorm(50, mean = 20, sd = 3)

x2\_1 <- rnorm(50, mean = 14, sd = 4)

x2\_2 <- rnorm(50, mean = 18, sd = 4)

plot(x1\_1, x2\_1, pch=21, xlim=c(0, 30),ylim=c(0, 30))

points(x1\_2,x2\_2, pch=22)

x1 <- c(x1\_1, x1\_2)

x2 <- c(x2\_1, x2\_2)

class<-c(rep('-1',50),rep('1',50))

t<-data.frame(x1, x2, class, stringsAsFactors = TRUE)

for (i in seq(20,80,by=10))

{

idx<-sample(1:dim(t)[1],20)

train<-t[-idx, ]

test<-t[idx, ]

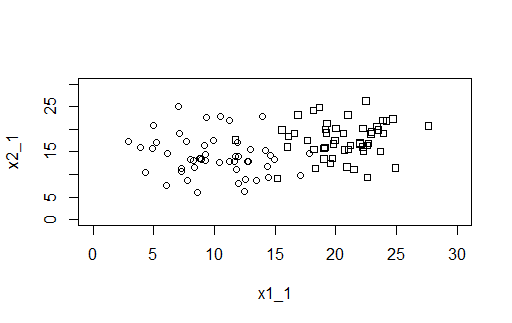
model<-naiveBayes(train[,-3],train$class)

t\_predicted<-predict(model, test)

print(table(t\_predicted,test$class))

}

#### Результаты работы программы:



t\_predicted -1 1

-1 9 1

1 0 10

t\_predicted -1 1

-1 14 0

1 0 6

t\_predicted -1 1

-1 8 2

1 0 10

t\_predicted -1 1

-1 11 0

1 0 9

t\_predicted -1 1

-1 8 0

1 1 11

t\_predicted -1 1

-1 10 0

1 0 10

t\_predicted -1 1

-1 8 1

1 0 11