Упражнение №3

Галиченко Евгений

14 03 2020

## Постановка задачи

На наборе данных из своего варианта построить указанные модели для прогноза бинарной зависимой переменной. Доля обучающей выборки - 75%.

Построить три графика:

* 1. Матричный график взаимного разброса переменных модели (ggpairs).
  2. Две ROC-кривые на одних осях: сравнение качества прогноза сравниваемых моделей на обучающей выборке.
  3. Две ROC-кривые на одних осях: сравнение качества прогноза сравниваемых моделей на тестовой выборке.

### Вариант 4

* Ядро для set.seed() - 123.
* Данные: Glass{mlbench} - химический состав разных типов стекла.
* Зависимая переменная: Type 1 (1 - наличие признака, все остальные - отсутствие).
* Объясняющие переменные: Все остальные.
* Методы: Логистическая регрессия, LDA.

*Пакеты*:

library('ISLR')  
library('GGally')  
library('MASS')  
library('mlbench')  
  
data(Glass)  
head(Glass)

## RI Na Mg Al Si K Ca Ba Fe Type  
## 1 1.52101 13.64 4.49 1.10 71.78 0.06 8.75 0 0.00 1  
## 2 1.51761 13.89 3.60 1.36 72.73 0.48 7.83 0 0.00 1  
## 3 1.51618 13.53 3.55 1.54 72.99 0.39 7.78 0 0.00 1  
## 4 1.51766 13.21 3.69 1.29 72.61 0.57 8.22 0 0.00 1  
## 5 1.51742 13.27 3.62 1.24 73.08 0.55 8.07 0 0.00 1  
## 6 1.51596 12.79 3.61 1.62 72.97 0.64 8.07 0 0.26 1

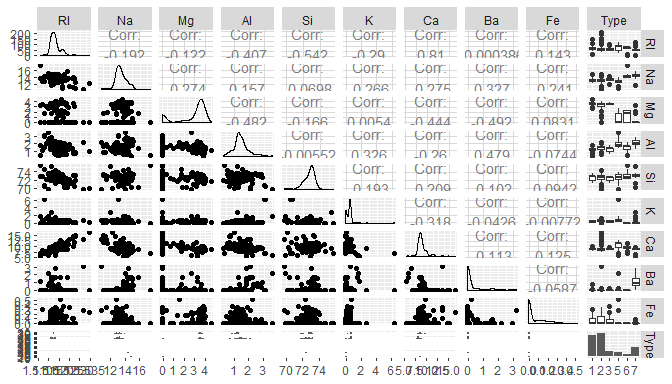
# Графики разброса

Зададим ядро генератора случайных чисел и объем обучающей выборки.

# Зададим ядро генератора случайных чисел и объем обучающей выборки  
my.seed <- 123  
train.percent <- 0.75  
options("ggmatrix.progress.bar" = FALSE)

Исходные данные: набор Glass (Химический состав разных типов стекла)

ggp <- ggpairs(Glass)  
print(ggp, progress = FALSE)



# Создаем вектор Type1  
Type1 <- rep(0, length(Glass$Type))  
# Добавляем Type1 во фрейм Glass  
Glass <- cbind(Glass, Type1)  
# Если Type = 1, то Type1 = 1, остальные 0  
for(i in 1:length(Glass$Type)) {if (Glass$Type[i] == 1) {Glass$Type1[i] = 1}}  
  
# Определение долей  
table(Glass$Type1) / sum(table(Glass$Type1))

##   
## 0 1   
## 0.6728972 0.3271028

Для наименьшего класса, в данном случае 0.327, это ошибка нулевого классификатора: если бы мы прогнозировали Type = 1 для всех наблюдений, ровно в такой доле случаев мы бы ошиблись. Точность моделей целесообразно будет сравнивать с этой величиной.

# Отбираем наблюдения в обучающую выборку  
set.seed(my.seed)  
inTrain <- sample(seq\_along(Glass$Type1),  
 nrow(Glass)\*train.percent)  
df <- Glass[inTrain, ]  
dfp <- Glass[-inTrain, ]  
  
# Фактические значения на обучающей выборке  
Fact <- df$Type1  
# Фактические значения на тестовой выборке  
Factp <- dfp$Type1

# Строим модели, чтобы спрогнозировать Type

# Логистическая регрессия

# Обучающая выборка  
model.logit <- glm(Type1 ~ RI + Na + Mg + Al + Si + K + Ca + Ba + Fe, data = df, family = 'binomial')  
summary(model.logit)

##   
## Call:  
## glm(formula = Type1 ~ RI + Na + Mg + Al + Si + K + Ca + Ba +   
## Fe, family = "binomial", data = df)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.09108 -0.75026 -0.08737 0.80974 1.84482   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -402.8465 365.1582 -1.103 0.26994   
## RI -85.1697 211.8588 -0.402 0.68768   
## Na 4.4270 2.4722 1.791 0.07334 .   
## Mg 6.7264 2.5942 2.593 0.00952 \*\*  
## Al 2.1844 2.7020 0.808 0.41884   
## Si 5.4672 2.4414 2.239 0.02513 \*   
## K 5.7741 2.7770 2.079 0.03759 \*   
## Ca 5.4173 2.6019 2.082 0.03733 \*   
## Ba 6.8283 2.9445 2.319 0.02040 \*   
## Fe -0.3573 2.3658 -0.151 0.87997   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 201.79 on 159 degrees of freedom  
## Residual deviance: 138.12 on 150 degrees of freedom  
## AIC: 158.12  
##   
## Number of Fisher Scoring iterations: 7

# Прогноз: вероятности принадлежности классу Type = 1  
p.logit <- predict(model.logit, df, type = 'response')  
  
Forecast1 <- factor(ifelse(p.logit > 0.5, 2, 1), levels = c(1, 2), labels = c('0', '1'))  
  
# Матрица неточностей  
conf.m <- table(Fact, Forecast1)  
conf.m

## Forecast1  
## Fact 0 1  
## 0 93 15  
## 1 19 33

# Чувствительность  
conf.m[2, 2] / sum(conf.m[2, ])

## [1] 0.6346154

# Специфичность  
conf.m[1, 1] / sum(conf.m[1, ])

## [1] 0.8611111

# Верность  
sum(diag(conf.m)) / sum(conf.m)

## [1] 0.7875

# Ошибка нулевого классификатора  
sum(Glass$Type1 == 1) / length(Glass$Type1)

## [1] 0.3271028

У этой модели сильная чувствительность

#LDA

model.lda <- lda(Type1 ~ RI + Na + Mg + Al + Si + K + Ca + Ba + Fe, data = Glass[inTrain, ])  
  
model.lda

## Call:  
## lda(Type1 ~ RI + Na + Mg + Al + Si + K + Ca + Ba + Fe, data = Glass[inTrain,   
## ])  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.675 0.325   
##   
## Group means:  
## RI Na Mg Al Si K Ca Ba  
## 0 1.518087 13.49426 2.228426 1.563519 72.75278 0.4868519 9.039630 0.23861111  
## 1 1.518724 13.24327 3.538077 1.149231 72.60808 0.4571154 8.825962 0.01538462  
## Fe  
## 0 0.06185185  
## 1 0.05750000  
##   
## Coefficients of linear discriminants:  
## LD1  
## RI -12.0172992  
## Na 4.3747100  
## Mg 5.1330725  
## Al 3.4093619  
## Si 4.6789140  
## K 4.7351448  
## Ca 4.6137088  
## Ba 4.9598638  
## Fe 0.7542241

# Прогноз: вероятности принадлежности классу Type = 1  
  
p.lda <- predict(model.lda, df, type = 'response')  
  
Forecast2 <- factor(ifelse(p.lda$posterior[, '1'] > 0.5, 2, 1), levels = c(1, 2), labels = c('0', '1'))  
  
#Матрица неточностей  
conf.m <- table(Fact, Forecast2)  
conf.m

## Forecast2  
## Fact 0 1  
## 0 94 14  
## 1 24 28

# Чувствительность  
conf.m[2, 2] / sum(conf.m[2, ])

## [1] 0.5384615

# Специфичность  
conf.m[1, 1] / sum(conf.m[1, ])

## [1] 0.8703704

# Верность  
sum(diag(conf.m)) / sum(conf.m)

## [1] 0.7625

# Ошибка нулевого классификатора  
sum(Glass$Type1 == 1) / length(Glass$Type1)

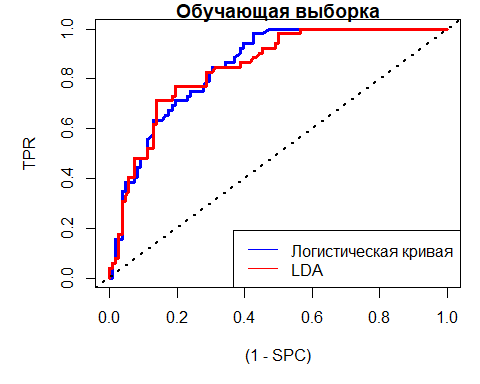
## [1] 0.3271028

У этой модели чувствительность меньше

# Подбор границы отсечения вероятностей классов

# ROC-кривые для обучающей выборки

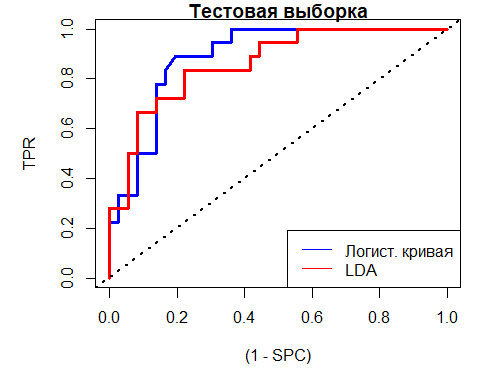
# Считаем 1-SPC и TPR для всех вариантов границы отсечения  
# Для (1 - SPC)  
x1 <- NULL   
# Для TPR  
y1 <- NULL  
  
# Логистическая регрессия  
# Заготовка под матрицу неточностей  
tbl1 <- as.data.frame(matrix(rep(0, 4), 2, 2))  
rownames(tbl1) <- c('fact.0', 'fact.1')  
colnames(tbl1) <- c('predict.0', 'predict.1')  
  
# Цикл по вероятностям отсечения  
for(p in seq(0, 1, length = 501)){  
 # Прогноз  
 Forecast1 <- factor(ifelse(p.logit > p, 2, 1), levels = c(1, 2), labels = c('0', '1'))  
   
 # Фрейм со сравнением факта и прогноза  
 df.compare <- data.frame(Fact = Fact, Forecast = Forecast1)  
   
 #Заполняем матрицу неточностей  
 tbl1[1, 1] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '0', ])  
 tbl1[2, 2] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '1', ])  
 tbl1[1, 2] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '1', ])  
 tbl1[2, 1] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '0', ])  
   
 # Считаем характиристики  
 TPR <- tbl1[2, 2] / sum(tbl1[2, ])  
 y1 <- c(y1, TPR)  
 SPC <- tbl1[1, 1] / sum(tbl1[1, ])  
 x1 <- c(x1, 1 - SPC)  
}  
  
# LDA  
# Для (1 - SPC)  
x2 <- NULL  
# Для TPR  
y2 <- NULL  
# Заготовка под матрицу неточностей  
tbl2 <- as.data.frame(matrix(rep(0, 4), 2, 2))  
rownames(tbl2) <- c('fact.0', 'fact.1')  
colnames(tbl2) <- c('predict.0', 'predict.1')  
# Цикл по вероятностям отсечения  
for (p in seq(0, 1, length = 501)){  
 # Прогноз  
 Forecast2 <- factor(ifelse(p.lda$posterior[, '1'] > p, 2, 1),  
 levels = c(1, 2),  
 labels = c('0', '1'))  
   
 # фрейм со сравнением факта и прогноза  
 df.compare <- data.frame(Fact = Fact, Forecast = Forecast2)  
   
 # Заполняем матрицу неточностей  
 tbl2[1, 1] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '0', ])  
 tbl2[2, 2] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '1', ])  
 tbl2[1, 2] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '1', ])  
 tbl2[2, 1] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '0', ])  
   
 # Считаем характеристики  
 TPR <- tbl2[2, 2] / sum(tbl2[2, ])  
 y2 <- c(y2, TPR)  
 SPC <- tbl2[1, 1] / sum(tbl2[1, ])  
 x2 <- c(x2, 1 - SPC)  
}  
  
# Строим ROC-кривую  
par(mar = c(5, 5, 1, 1))  
  
# Кривая (логистическая регрессия)  
plot(x1, y1, type = 'l', col = 'blue', lwd = 3,  
 xlab = '(1 - SPC)', ylab = 'TPR',  
 xlim = c(0, 1), ylim = c(0, 1), main = 'Обучающая выборка')  
  
# Кривая (LDA)  
lines(x2, y2, type = 'l', col = 'red', lwd = 3)  
  
# Прямая случайного классификатора  
abline(a = 0, b = 1, lty = 3, lwd = 2)  
  
# Легенда  
legend('bottomright', names <- c('Логистическая кривая', 'LDA'), lty = 1, col = c('blue', 'red'))



Сравнивая ROC-кривые, полученные на обучающей выборке, сложно сказать, какая из моделей наиболее предпочтительна. Для того, чтобы ответить на этот вопрос построим ROC-кривые на тестовых данных.

# ROC-кривые для тестовой выборки

# Логистическая модель  
# Прогноз: вероятности принадлежности классу Type = 2  
p.logit <- predict(model.logit, dfp,   
 type = 'response')  
# Считаем 1-SPC и TPR для всех вариантов границы отсечения  
x1 <- NULL # Для (1 - SPC)  
y1 <- NULL # Для TPR  
# Заготовка под матрицу неточностей  
tbl1 <- as.data.frame(matrix(rep(0, 4), 2, 2))  
rownames(tbl1) <- c('fact.0', 'fact.1')  
colnames(tbl1) <- c('predict.0', 'predict.1')  
# Цикл по вероятностям отсечения  
for (p in seq(0, 1, length = 501)){  
 # Прогноз  
 Forecast1 <- factor(ifelse(p.logit > p, 2, 1),  
 levels = c(1, 2),  
 labels = c('0', '1'))  
 # Фрейм со сравнением факта и прогноза  
 df.compare <- data.frame(Fact = Factp, Forecast = Forecast1)  
 # Заполняем матрицу неточностей  
 tbl1[1, 1] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '0', ])  
 tbl1[2, 2] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '1', ])  
 tbl1[1, 2] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '1', ])  
 tbl1[2, 1] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '0', ])  
 # Считаем характеристики  
 TPR <- tbl1[2, 2] / sum(tbl1[2, ])  
 y1 <- c(y1, TPR)  
 SPC <- tbl1[1, 1] / sum(tbl1[1, ])  
 x1 <- c(x1, 1 - SPC)}  
# LDA  
# Прогноз: вероятности принадлежности классу Type = 2  
p.lda <- predict(model.lda, dfp,   
 type = 'response')  
x2 <- NULL # для (1 - SPC)  
y2 <- NULL # для TPR  
# Заготовка под матрицу неточностей  
tbl2 <- as.data.frame(matrix(rep(0, 4), 2, 2))  
rownames(tbl2) <- c('fact.0', 'fact.1')  
colnames(tbl2) <- c('predict.0', 'predict.1')  
# Цикл по вероятностям отсечения  
for (p in seq(0, 1, length = 501)){  
 # Прогноз  
 Forecast2 <- factor(ifelse(p.lda$posterior[, '1'] > p, 2, 1),  
 levels = c(1, 2),  
 labels = c('0', '1'))  
   
 # Фрейм со сравнением факта и прогноза  
 df.compare <- data.frame(Fact = Factp, Forecast = Forecast2)  
   
 # Заполняем матрицу неточностей  
 tbl2[1, 1] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '0', ])  
 tbl2[2, 2] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '1', ])  
 tbl2[1, 2] <- nrow(df.compare[df.compare$Fact == '0' & df.compare$Forecast == '1', ])  
 tbl2[2, 1] <- nrow(df.compare[df.compare$Fact == '1' & df.compare$Forecast == '0', ])  
   
 # Считаем характеристики  
 TPR <- tbl2[2, 2] / sum(tbl2[2, ])  
 y2 <- c(y2, TPR)  
 SPC <- tbl2[1, 1] / sum(tbl2[1, ])  
 x2 <- c(x2, 1 - SPC)  
   
}  
# Строим ROC-кривую  
par(mar = c(5, 5, 1, 1))  
# Кривая (логистическая регрессия)  
plot(x1, y1, type = 'l', col = 'blue', lwd = 3,  
 xlab = '(1 - SPC)', ylab = 'TPR',   
 xlim = c(0, 1), ylim = c(0, 1), main = 'Тестовая выборка')  
# Кривая (LDA)  
lines(x2, y2, type = 'l', col = 'red', lwd = 3)  
# Прямая случайного классификатора  
abline(a = 0, b = 1, lty = 3, lwd = 2)  
# Легенда  
legend('bottomright', names <- c('Логист. кривая', 'LDA'), lty = 1, col = c('blue', 'red'))



Сравнивая ROC-кривые, полученные на тестовой выборке, видно, что LDA-модель обладает большей предсказательной способностью, чем логистическая регрессия.

Дискриминантный анализ не имеет столько допущений, как логистическая регрессия. Поэтому если допущения логистической регрессии не выполняются, то дискриминантный анализ является лучшем средством для анализа.