Упражнение № 9

Людуховский В.В.

12 05 2021

Необходимо построить модель на основе SVM для указанной в варианте зависимой переменной.

Данные взять из упражнения №3.

Для модели:

1 Отложить 25% наблюдений в тестовую выборку (ядро генератора случайных чисел указано в варианте к упражнению №3).

2 На обучающей выборке (оставшихся 75% наблюдений) сравнить несколько видов ядер SVM по точности модели (AUC) методом сеточного поиска.

3 Для оптимальной формы ядерной функции на обучающей выборке подобрать оптимальное значение настроечных параметров по минимальной ошибке с перекрёстной проверкой (функция tune).

4 Подогнать лучшую модель на всей обучающей выборке. Построить ROC-кривую и рассчитать матрицу неточностей, чувствительность и специфичность.

5 Сделать прогноз по лучшей модели на тестовую выборку, оценить его качество точность по матрице неточностей, чувствительность и специфичность, построить ROC-кривую.

6 Сравнить результаты, которые дал SVM, с результатами, полученными в упражнении 3. Какой из методов оказался лучше?

Вариант - 18 Данные: *Glass{mlbench}* Зависимая переменная: *Type 7* (1 – наличие признака, все остальные – отсутствие) все остальные

Исходные данные: набор Glass

Зададим ядро генератора случайных чисел и объём обучающей выборки.

##   
## 0 1   
## 0.864486 0.135514

Доля наименьшего класса, в данном случае 0.355, это ошибка нулевого классификатора: если бы мы прогнозировали Type = 7 для всех наблюдений, ровно в такой доле случаев мы бы ошиблись. Точность моделей целесообразно будет сравнивать с этой величиной.

# Сеточный поиск

## [1] "Starting linear kernel"  
## [1] "Starting polynomial kernel"

## cost = 1 cost = 1.5 cost = 2 cost = 2.5 cost = 3 cost = 3.5  
## kernel = linear 0.926 0.926 0.944 0.944 0.944 0.944  
## kernel = polynomial 0.981 0.981 0.981 0.981 0.981 0.981  
## cost = 4 cost = 4.5 cost = 5 cost = 5.5 cost = 6 cost = 6.5  
## kernel = linear 0.926 0.926 0.907 0.907 0.889 0.907  
## kernel = polynomial 0.981 0.981 0.981 0.981 0.981 0.981  
## cost = 7 cost = 7.5 cost = 8 cost = 8.5 cost = 9 cost = 9.5  
## kernel = linear 0.907 0.907 0.907 0.907 0.907 0.907  
## kernel = polynomial 0.981 0.981 0.981 0.981 0.981 0.981  
## cost = 10 cost = 10.5 cost = 11 cost = 11.5 cost = 12  
## kernel = linear 0.907 0.907 0.907 0.907 0.907  
## kernel = polynomial 0.981 0.981 0.981 0.981 0.981  
## cost = 12.5 cost = 13 cost = 13.5 cost = 14 cost = 14.5  
## kernel = linear 0.907 0.907 0.907 0.907 0.907  
## kernel = polynomial 1.000 1.000 1.000 1.000 1.000  
## cost = 15 cost = 15.5 cost = 16 cost = 16.5 cost = 17  
## kernel = linear 0.907 0.907 0.907 0.907 0.907  
## kernel = polynomial 1.000 1.000 1.000 1.000 1.000  
## cost = 17.5 cost = 18 cost = 18.5 cost = 19 cost = 19.5  
## kernel = linear 0.907 0.907 0.907 0.907 0.907  
## kernel = polynomial 1.000 1.000 1.000 1.000 1.000  
## cost = 20  
## kernel = linear 0.907  
## kernel = polynomial 1.000

Из полученных результатов видно, что оптимальной формой ядерной функции будет линейная модель.

# Оптимальное значение настроечного параметра

## [1] 13 31 57 61 118 129 41 78 106 137

##   
## Call:  
## svm(formula = y ~ ., data = dat, kernel = "linear", cost = 10, cale = FALSE)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 10   
##   
## Number of Support Vectors: 10  
##   
## ( 6 4 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost  
## 1  
##   
## - best performance: 0.0375   
##   
## - Detailed performance results:  
## cost error dispersion  
## 1 1e-03 0.11250 0.07095578  
## 2 1e-01 0.04375 0.04218428  
## 3 1e+00 0.03750 0.04370037  
## 4 5e+00 0.05625 0.05472469  
## 5 1e+01 0.04375 0.05145454  
## 6 1e+02 0.06250 0.05103104

# Лучшая модель на всей обучающей выборке

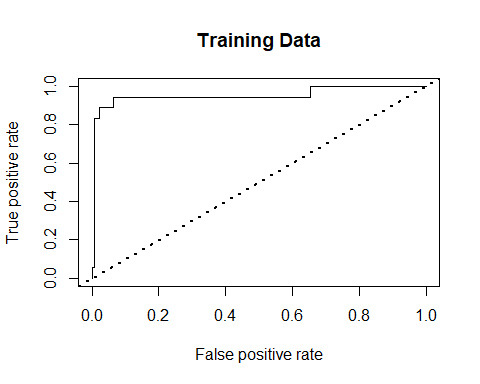
##   
## Call:  
## best.tune(method = svm, train.x = y ~ ., data = dat, ranges = list(cost = c(0.001,   
## 0.1, 1, 5, 10, 100)), kernel = "linear")  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 1   
##   
## Number of Support Vectors: 16  
##   
## ( 8 8 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

## Fact  
## Predicts 0 1  
## 0 141 2  
## 1 1 16

## [1] 0.941

## [1] 0.986

## Fact  
## Predicts 0 1  
## 0 141 3  
## 1 1 15



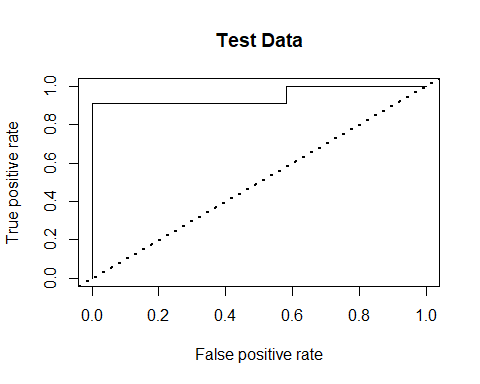
# Лучшая модель на тестовой выборке

## Fact  
## Predicts 0 1  
## 0 43 2  
## 1 0 9

## [1] 0.963

## [1] 1

## [1] 0.956



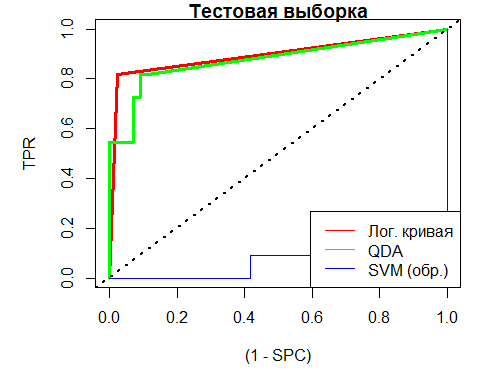
Как видно из графиков ROC-кривых, и для обучающей, и для тестовой выборок значение AUC менее 0.5, а значит классификатор действует с точностью до наоборот: если положительные классификации назвать отрицательными и наоборот, классификатор будет работать лучше. Учтём это при сравнении моделей на тестовой выборке.

# Сравнение моделей (логистическая регрессия, LDA, SVM) на тестовой выборке

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##   
## Call:  
## glm(formula = y ~ ., family = "binomial", data = dat)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.386e-04 -2.100e-08 -2.100e-08 -2.100e-08 1.223e-04   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -156697.8 43940239.5 -0.004 0.997  
## x.RI 52946.8 11492005.1 0.005 0.996  
## x.Na 762.0 289468.2 0.003 0.998  
## x.Mg 672.9 276451.6 0.002 0.998  
## x.Al 799.2 300218.4 0.003 0.998  
## x.Si 780.5 298538.2 0.003 0.998  
## x.K 681.9 306654.4 0.002 0.998  
## x.Ca 645.9 265314.2 0.002 0.998  
## x.Ba 771.5 273851.0 0.003 0.998  
## x.Fe -860.3 458916.0 -0.002 0.999  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1.1255e+02 on 159 degrees of freedom  
## Residual deviance: 6.3281e-08 on 150 degrees of freedom  
## AIC: 20  
##   
## Number of Fisher Scoring iterations: 25



Сравнивая ROC-кривые, полученные на тестовой выборке, видно, что логистическая регрессия обладает большей предсказательной способностью, чем QDA-модель и SVM(обр.).