hw2.R

Brian Kang

Tue Apr 16 00:41:28 2019

# Hw2  
# 2  
# predict whether a given suburb has a crime rate above or below the median  
#rm(list=ls())  
  
library(MASS)  
data("Boston")  
head(Boston)

## crim zn indus chas nox rm age dis rad tax ptratio black  
## 1 0.00632 18 2.31 0 0.538 6.575 65.2 4.0900 1 296 15.3 396.90  
## 2 0.02731 0 7.07 0 0.469 6.421 78.9 4.9671 2 242 17.8 396.90  
## 3 0.02729 0 7.07 0 0.469 7.185 61.1 4.9671 2 242 17.8 392.83  
## 4 0.03237 0 2.18 0 0.458 6.998 45.8 6.0622 3 222 18.7 394.63  
## 5 0.06905 0 2.18 0 0.458 7.147 54.2 6.0622 3 222 18.7 396.90  
## 6 0.02985 0 2.18 0 0.458 6.430 58.7 6.0622 3 222 18.7 394.12  
## lstat medv  
## 1 4.98 24.0  
## 2 9.14 21.6  
## 3 4.03 34.7  
## 4 2.94 33.4  
## 5 5.33 36.2  
## 6 5.21 28.7

attach(Boston) # from lecture  
summary(Boston)

## crim zn indus chas   
## Min. : 0.00632 Min. : 0.00 Min. : 0.46 Min. :0.00000   
## 1st Qu.: 0.08204 1st Qu.: 0.00 1st Qu.: 5.19 1st Qu.:0.00000   
## Median : 0.25651 Median : 0.00 Median : 9.69 Median :0.00000   
## Mean : 3.61352 Mean : 11.36 Mean :11.14 Mean :0.06917   
## 3rd Qu.: 3.67708 3rd Qu.: 12.50 3rd Qu.:18.10 3rd Qu.:0.00000   
## Max. :88.97620 Max. :100.00 Max. :27.74 Max. :1.00000   
## nox rm age dis   
## Min. :0.3850 Min. :3.561 Min. : 2.90 Min. : 1.130   
## 1st Qu.:0.4490 1st Qu.:5.886 1st Qu.: 45.02 1st Qu.: 2.100   
## Median :0.5380 Median :6.208 Median : 77.50 Median : 3.207   
## Mean :0.5547 Mean :6.285 Mean : 68.57 Mean : 3.795   
## 3rd Qu.:0.6240 3rd Qu.:6.623 3rd Qu.: 94.08 3rd Qu.: 5.188   
## Max. :0.8710 Max. :8.780 Max. :100.00 Max. :12.127   
## rad tax ptratio black   
## Min. : 1.000 Min. :187.0 Min. :12.60 Min. : 0.32   
## 1st Qu.: 4.000 1st Qu.:279.0 1st Qu.:17.40 1st Qu.:375.38   
## Median : 5.000 Median :330.0 Median :19.05 Median :391.44   
## Mean : 9.549 Mean :408.2 Mean :18.46 Mean :356.67   
## 3rd Qu.:24.000 3rd Qu.:666.0 3rd Qu.:20.20 3rd Qu.:396.23   
## Max. :24.000 Max. :711.0 Max. :22.00 Max. :396.90   
## lstat medv   
## Min. : 1.73 Min. : 5.00   
## 1st Qu.: 6.95 1st Qu.:17.02   
## Median :11.36 Median :21.20   
## Mean :12.65 Mean :22.53   
## 3rd Qu.:16.95 3rd Qu.:25.00   
## Max. :37.97 Max. :50.00

# actuals: if greater than median, T; otherwise F  
binCrim <- rep(F, length(crim))  
binCrim[crim > median(crim)] <- T  
Boston <- cbind(Boston, binCrim)  
  
# split data to training and test  
bostonTrain <- Boston[1:(0.8\*nrow(Boston)),] # 80% for training  
bostonTest <- Boston[(0.8\*nrow(Boston)+1):nrow(Boston),] # 20% for test  
train <- rep(T, nrow(Boston)) # logical vector  
train[(0.8\*nrow(Boston)+1):nrow(Boston)+1] <- F # apparently end case is exclusive?  
binCrimTest <- binCrim[!train] # save crim values for "actuals" (test)  
  
# run LOGIT using training data  
logit1 <- glm(binCrim ~ . - binCrim - crim, data = Boston, family = "binomial", subset = train)  
summary(logit1)

##   
## Call:  
## glm(formula = binCrim ~ . - binCrim - crim, family = "binomial",   
## data = Boston, subset = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.3643 -0.1998 -0.0136 0.0687 3.4545   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -42.799126 7.890654 -5.424 5.83e-08 \*\*\*  
## zn -0.066836 0.036978 -1.807 0.070693 .   
## indus -0.088478 0.051755 -1.710 0.087348 .   
## chas 1.023592 0.753024 1.359 0.174049   
## nox 59.170895 9.555418 6.192 5.93e-10 \*\*\*  
## rm -0.676176 0.816094 -0.829 0.407358   
## age 0.008651 0.012974 0.667 0.504913   
## dis 0.654216 0.232581 2.813 0.004910 \*\*   
## rad 0.621347 0.183645 3.383 0.000716 \*\*\*  
## tax -0.001433 0.003760 -0.381 0.703173   
## ptratio 0.485265 0.141219 3.436 0.000590 \*\*\*  
## black -0.009549 0.006112 -1.562 0.118192   
## lstat 0.068709 0.054149 1.269 0.204480   
## medv 0.202732 0.080269 2.526 0.011548 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 548.94 on 404 degrees of freedom  
## Residual deviance: 182.50 on 391 degrees of freedom  
## AIC: 210.5  
##   
## Number of Fisher Scoring iterations: 9

# predict probability  
logit1.prob <- predict(logit1, bostonTest, type = "response")  
  
# predicting if crime rate below or above median  
logit1.pred <- rep(F, nrow(bostonTest))  
logit1.pred[logit1.prob > 0.5] <- T  
  
# confusion matrix  
table(logit1.pred, binCrimTest)

## binCrimTest  
## logit1.pred FALSE TRUE  
## FALSE 4 0  
## TRUE 11 86

mean(logit1.pred==binCrimTest)

## [1] 0.8910891

# We can see that our predictions were around 89% correct.  
# From the confusion matrix we can see that we correctly predicted all 86  
# suburb cities that the crime rates were above the median, which is good.  
# But, looking at the 15 cities had rates below the median, we incorrectly  
# predicted 11 and only 4 were correct, in this sense our predictions are  
# very bad.  
  
# Same logit process but different ratios  
bostonTrain2 <- Boston[1:(0.65\*nrow(Boston)),] # 65% for training  
bostonTest2 <- Boston[(0.65\*nrow(Boston)+1):nrow(Boston),] # 35% for test  
train2 <- rep(T, nrow(Boston)) # logical vector  
train2[(0.65\*nrow(Boston)+1):nrow(Boston)+1] <- F # apparently end case is exclusive?  
binCrimTest2 <- binCrim[!train2] # save crim values for "actuals" (test)  
logit2 <- glm(binCrim ~ . - binCrim - crim, data = Boston, family = "binomial", subset = train)  
summary(logit2)

##   
## Call:  
## glm(formula = binCrim ~ . - binCrim - crim, family = "binomial",   
## data = Boston, subset = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.3643 -0.1998 -0.0136 0.0687 3.4545   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -42.799126 7.890654 -5.424 5.83e-08 \*\*\*  
## zn -0.066836 0.036978 -1.807 0.070693 .   
## indus -0.088478 0.051755 -1.710 0.087348 .   
## chas 1.023592 0.753024 1.359 0.174049   
## nox 59.170895 9.555418 6.192 5.93e-10 \*\*\*  
## rm -0.676176 0.816094 -0.829 0.407358   
## age 0.008651 0.012974 0.667 0.504913   
## dis 0.654216 0.232581 2.813 0.004910 \*\*   
## rad 0.621347 0.183645 3.383 0.000716 \*\*\*  
## tax -0.001433 0.003760 -0.381 0.703173   
## ptratio 0.485265 0.141219 3.436 0.000590 \*\*\*  
## black -0.009549 0.006112 -1.562 0.118192   
## lstat 0.068709 0.054149 1.269 0.204480   
## medv 0.202732 0.080269 2.526 0.011548 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 548.94 on 404 degrees of freedom  
## Residual deviance: 182.50 on 391 degrees of freedom  
## AIC: 210.5  
##   
## Number of Fisher Scoring iterations: 9

logit2.prob <- predict(logit2, bostonTest2, type = "response")  
logit2.pred <- rep(F, nrow(bostonTest2))  
logit2.pred[logit2.prob > 0.5] <- T  
table(logit2.pred, binCrimTest2)

## binCrimTest2  
## logit2.pred FALSE TRUE  
## FALSE 23 1  
## TRUE 19 134

mean(logit2.pred==binCrimTest2)

## [1] 0.8870056

# This time, we correctly predicted about 88.7% of every prediction.  
# From the confusion matrix, we can see that out of the 135 cities with  
# crime rates above median, we correctly predicted 134, which is great.  
# The number size itself grew because now the test data size is larger.  
# Now, looking at the cities that have crime rates lower than the median,  
# out of the 42, we incorrectly predicted 19 and 23 correctly. This does  
# not mean that our prediction got better, it was simply the variation in  
# our data that gave this output. The prediction for "safer" cities are  
# still very bad.  
  
# Now using KNN  
library(class)  
traink <- Boston[train,]  
testk <- Boston[!train,]  
binCrimTestk <- binCrim[train]  
set.seed(999)  
knn1 <- knn(traink, testk, binCrimTestk, k=1)  
table(knn1, binCrimTest)

## binCrimTest  
## knn1 FALSE TRUE  
## FALSE 10 3  
## TRUE 5 83

mean(knn1==binCrimTest)

## [1] 0.9207921

# Using KNN, k=1, we correctly predicted about 92% of the times.  
# Out of the 86 cities with crime rates above median, 83 predictions were  
# correct. Out of the 15 with lower rates, 5 were incorrect. Ratio wise,  
# the incorrectness for "safer" cities are better less using KNN.  
  
# Same KNN but different K value  
set.seed(999)  
knn2 <- knn(traink, testk, binCrimTestk, k=1)  
table(knn2, binCrimTest)

## binCrimTest  
## knn2 FALSE TRUE  
## FALSE 10 3  
## TRUE 5 83

mean(knn2==binCrimTest)

## [1] 0.9207921

# With k=10, we correctly predicted around 92% of the times.  
# The correct and incorrect values for the both kind of cities are the same.