R Notebook

Predicting parole violators

In many criminal justice systems around the world, inmates deemed not to be a threat to society are released from prison under the parole system prior to completing their sentence. They are still considered to be serving their sentence while on parole, and they can be returned to prison if they violate the terms of their parole.

Parole boards are charged with identifying which inmates are good candidates for release on parole. They seek to release inmates who will not commit additional crimes after release. In this problem, we will build and validate a model that predicts if an inmate will violate the terms of his or her parole. Such a model could be useful to a parole board when deciding to approve or deny an application for parole.

For this prediction task, we will use data from the United States 2004 National Corrections Reporting Program, a nationwide census of parole releases that occurred during 2004. We limited our focus to parolees who served no more than 6 months in prison and whose maximum sentence for all charges did not exceed 18 months. The dataset contains all such parolees who either successfully completed their term of parole during 2004 or those who violated the terms of their parole during that year. The dataset contains the following variables:

male: 1 if the parolee is male, 0 if female **race**: 1 if the parolee is white, 2 otherwise **age**: the parolee's age (in years) when he or she was released from prison **state**: a code for the parolee's state. 2 is Kentucky, 3 is Louisiana, 4 is Virginia, and 1 is any other state. The three states were selected due to having a high representation in the dataset. **time.served**: the number of months the parolee served in prison (limited by the inclusion criteria to not exceed 6 months). **max.sentence**: the maximum sentence length for all charges, in months (limited by the inclusion criteria to not exceed 18 months). **multiple.offenses**: 1 if the parolee was incarcerated for multiple offenses, 0 otherwise. **crime**: a code for the parolee's main crime leading to incarceration. 2 is larceny, 3 is drug-related crime, 4 is driving-related crime, and 1 is any other crime. **violator**: 1 if the parolee violated the parole, and 0 if the parolee completed the parole without violation.

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.4.2
```

```
## -- Attaching packages -------------------------------------------------------------------------------
```

```
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  2.0.1     v dplyr   0.7.8
## v tidyr   0.8.0     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
## Warning: package 'tibble' was built under R version 3.4.4
```

```
## Warning: package 'tidyr' was built under R version 3.4.3
```

```
## Warning: package 'purrr' was built under R version 3.4.4
```

```
## Warning: package 'dplyr' was built under R version 3.4.4
```

```
## Warning: package 'forcats' was built under R version 3.4.3
```

```
## -- Conflicts ---------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

1.1) Loading the Dataset

```r
parole = read.csv('parole.csv')

str(parole)
```

```
## 'data.frame':    675 obs. of  9 variables:
##  $ male             : int  1 0 1 1 1 1 1 0 0 1 ...
##  $ race             : int  1 1 2 1 2 2 1 1 1 2 ...
##  $ age              : num  33.2 39.7 29.5 22.4 21.6 46.7 31 24.6 32.6 29.1 ...
##  $ state            : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ time.served      : num  5.5 5.4 5.6 5.7 5.4 6 6 4.8 4.5 4.7 ...
##  $ max.sentence     : int  18 12 12 18 12 18 18 12 13 12 ...
##  $ multiple.offenses: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ crime            : int  4 3 3 1 1 4 3 1 3 2 ...
##  $ violator         : int  0 0 0 0 0 0 0 0 0 0 ...
```

1.2) How many of the parolees in the dataset violated the terms of their parole?

```r
table(parole$violator)
```

```
##
##   0   1
## 597  78
```

```r
# 78
```

2.1) Preparing the Dataset

You should be familiar with unordered factors (if not, review the Week 2 homework problem "Reading Test Scores"). Which variables in this dataset are unordered factors with at least three levels? Select all that apply.

```r
table(parole$violator)
```

```
##
##   0   1
## 597  78
```

```r
# state and crime
```

Explanation While the variables male, race, state, crime, and violator are all unordered factors, only state and crime have at least 3 levels in this dataset.

2.2) Preparing the Dataset

In the last subproblem, we identified variables that are unordered factors with at least 3 levels, so we need to convert them to factors for our prediction problem (we introduced this idea in the "Reading Test Scores" problem last week). Using the as.factor() function, convert these variables to factors. Keep in mind that we are not changing the values, just the way R understands them (the values are still numbers).

How does the output of summary() change for a factor variable as compared to a numerical variable?

```
parole$state = as.factor(parole$state)

parole$crime = as.factor(parole$crime)

summary(parole)
```

```
##       male             race            age          state    time.served   
##  Min.   :0.0000   Min.   :1.000   Min.   :18.40   1:143   Min.   :0.000  
##  1st Qu.:1.0000   1st Qu.:1.000   1st Qu.:25.35   2:120   1st Qu.:3.250  
##  Median :1.0000   Median :1.000   Median :33.70   3: 82   Median :4.400  
##  Mean   :0.8074   Mean   :1.424   Mean   :34.51   4:330   Mean   :4.198  
##  3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:42.55           3rd Qu.:5.200  
##  Max.   :1.0000   Max.   :2.000   Max.   :67.00           Max.   :6.000  
##   max.sentence   multiple.offenses crime      violator     
##  Min.   : 1.00   Min.   :0.0000    1:315   Min.   :0.0000  
##  1st Qu.:12.00   1st Qu.:0.0000    2:106   1st Qu.:0.0000  
##  Median :12.00   Median :1.0000    3:153   Median :0.0000  
##  Mean   :13.06   Mean   :0.5363    4:101   Mean   :0.1156  
##  3rd Qu.:15.00   3rd Qu.:1.0000            3rd Qu.:0.0000  
##  Max.   :18.00   Max.   :1.0000            Max.   :1.0000
```

Ans: the output becomes similar to that of the table() function applied to that varibale

3.1) Spliting into a Training and Testing Set

```
set.seed(144)
library(caTools)
split = sample.split(parole$violator, SplitRatio = 0.7)
train = subset(parole, split == TRUE)
test = subset(parole, split == FALSE)
```

70% to the training set, 30% to the testing set

```
nrow(train)
```

```
## [1] 473
```

```
nrow(test)
```

```
## [1] 202
```

3.2) If you instead ONLY re-ran lines [3]-[5], what would you expect?

Explanation

If you set a random seed, split, set the seed again to the same value, and then split again, you will get the same split. However, if you set the seed and then split twice, you will get different splits. If you set the seed to different values, you will get different splits. You can also verify this by running the specified code in R. If you have training sets train1 and train2, the function sum(train1 != train2) will count the number of values in those two data frames that are different.

4.1) Building a Logistic Regression Model

If you tested other training/testing set splits in the previous section, please re-run the original 5 lines of code to obtain the original split.

Using glm (and remembering the parameter family="binomial"), train a logistic regression model on the training set. Your dependent variable is "violator", and you should use all of the other variables as independent variables.

What variables are significant in this model? Significant variables should have a least one star, or should have a probability less than 0.05 (the column Pr(>|z|) in the summary output). Select all that apply.

```
set.seed(144)
library(caTools)
split = sample.split(parole$violator, SplitRatio = 0.7)
train = subset(parole, split == TRUE)
test = subset(parole, split == FALSE)
```

```
mod_1 = glm(violator ~ ., data = train, family = 'binomial')

summary(mod_1)
```

```
## 
## Call:
## glm(formula = violator ~ ., family = "binomial", data = train)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -1.7041  -0.4236  -0.2719  -0.1690   2.8375  
## 
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)    
## (Intercept)       -4.2411574  1.2938852  -3.278  0.00105 ** 
## male               0.3869904  0.4379613   0.884  0.37690    
## race               0.8867192  0.3950660   2.244  0.02480 *  
## age               -0.0001756  0.0160852  -0.011  0.99129    
## state2             0.4433007  0.4816619   0.920  0.35739    
## state3             0.8349797  0.5562704   1.501  0.13335    
## state4            -3.3967878  0.6115860  -5.554 2.79e-08 ***
## time.served       -0.1238867  0.1204230  -1.029  0.30359    
## max.sentence       0.0802954  0.0553747   1.450  0.14705    
## multiple.offenses  1.6119919  0.3853050   4.184 2.87e-05 ***
## crime2             0.6837143  0.5003550   1.366  0.17180    
## crime3            -0.2781054  0.4328356  -0.643  0.52054    
## crime4            -0.0117627  0.5713035  -0.021  0.98357    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 340.04  on 472  degrees of freedom
## Residual deviance: 251.48  on 460  degrees of freedom
## AIC: 277.48
## 
## Number of Fisher Scoring iterations: 6
```

4.2) Building a Logistic Regression Model

What can we say based on the coefficient of the multiple.offenses variable?

The following two properties might be useful to you when answering this question:

1) If we have a coefficient c for a variable, then that means the log odds (or Logit) are increased by c for a unit increase in the variable.
2) If we have a coefficient c for a variable, then that means the odds are multiplied by e^c for a unit increase in the variable.

Answer: Our model predicts that a parolee who committed multiple offenses has 5.01 times higher odds of being a violator than a parolee who did not commit multiple offenses but is otherwise identical Explanation

For parolees A and B who are identical other than A having committed multiple offenses, the predicted log odds of A is 1.61 more than the predicted log odds of B. Then we have:

ln(odds of A) = ln(odds of B) + 1.61

exp(ln(odds of A)) = exp(ln(odds of B) + 1.61)

exp(ln(odds of A)) = exp(ln(odds of B)) * exp(1.61)

odds of A = exp(1.61) * odds of B

odds of A= 5.01 * odds of B

In the second step we raised e to the power of both sides. In the third step we used the exponentiation rule that e^(a+b) = e^a * e^b. In the fourth step we used the rule that e^(ln(x)) = x.

4.3) Building a Logistic Regression Model

Consider a parolee who is male, of white race, aged 50 years at prison release, from the state of Maryland, served 3 months, had a maximum sentence of 12 months, did not commit multiple offenses, and committed a larceny. Answer the following questions based on the model's predictions for this individual. (HINT: You should use the coefficients of your model, the Logistic Response Function, and the Odds equation to solve this problem.)

According to the model, what are the odds this individual is a violator? **Ans: 0.1825687** According to the model, what is the probability this individual is a violator? **Ans: 0.1543831 Explanation**

From the logistic regression equation, we have log(odds) = -4.2411574 + 0.3869904*male + 0.8867192*race - 0.0001756*age + 0.4433007*state2 + 0.8349797*state3 - 3.3967878*state4 - 0.1238867*time.served + 0.0802954*max.sentence + 1.6119919*multiple.offenses + 0.6837143*crime2 - 0.2781054*crime3 - 0.0117627*crime4.

This parolee has male=1, race=1, age=50, state2=0, state3=0, state4=0, time.served=3, max.sentence=12, multiple.offenses=0, crime2=1, crime3=0, crime4=0. We conclude that log(odds) = -1.700629.

Therefore, the odds ratio is exp(-1.700629) = 0.183, and the predicted probability of violation is 1/(1+exp(1.700629)) = 0.154.

5.1) Evaluating the Model on the Testing Set

Use the predict() function to obtain the model's predicted probabilities for parolees in the testing set, remembering to pass type="response".

What is the maximum predicted probability of a violation?

```r
predict_test = predict(mod_1, type = 'response', newdata = test)

summary(predict_test)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
## 0.002334 0.023777 0.057905 0.146576 0.147452 0.907279
```

Ans: 0.907279 Explanation

The following commands make the predictions and display a summary of the values:

predictions = predict(mod, newdata=test, type="response") summary(predictions)

5.2) Evaluating the Model on the Testing Set

In the following questions, evaluate the model's predictions on the test set using a threshold of 0.5. What is the model's sensitivity? What is the model's specificity? What is the model's accuracy?

```
predict_test = predict(mod_1, type = 'response',newdata = test)

table(test$violator, predict_test > 0.5)
```

```
##
##     FALSE TRUE
##   0   167   12
##   1    11   12
```

```
sensitivity = 12 / (12 + 11)

specificity = 167 / (167 + 12)

accuracy = (167 + 12) / (167 + 12 + 11 + 12)
```

```
sensitivity
```

```
## [1] 0.5217391
```

```
specificity
```

```
## [1] 0.9329609
```

```
accuracy
```

```
## [1] 0.8861386
```

Explanation

To obtain the confusion matrix, use the following command:

table(test$violator, as.numeric(predictions >= 0.5))

There are 202 observations in the test set. The accuracy (percentage of values on the diagonal) is (167+12)/202 = 0.886. The sensitivity (proportion of the actual violators we got correct) is 12/(11+12) = 0.522 The specificity (proportion of the actual non-violators we got correct) is 167/(167+12) = 0.933.

5.3) Evaluating the Model on the Testing Set

What is the accuracy of a simple model that predicts that every parolee is a non-violator?

```
accuracy_simple = (167 + 12) / 202

accuracy_simple
```

```
## [1] 0.8861386
```

Explanation If you table the outcome variable using the following command:

table(test$violator)

you can see that there are 179 negative examples, which are the ones that the baseline model would get correct. Thus the baseline model would have an accuracy of 179/202 = 0.886.

5.4) Evaluating the Model on the Testing Set

Consider a parole board using the model to predict whether parolees will be violators or not. The job of a parole board is to make sure that a prisoner is ready to be released into free society, and therefore parole boards tend to be particularily concerned about releasing prisoners who will violate their parole. Which of the following most likely describes their preferences and best course of action?

will violate = positive predict to not violate but will violate = false negative => more cost to false negative => higher specificity

```
table(test$violator, predict_test > 0.2)
```

```
##
##     FALSE TRUE
##   0   154   25
##   1     6   17
```

```
spec_low_cutoff =154 / (154 + 25)

spec_low_cutoff
```

```
## [1] 0.8603352
```

```
table(test$violator, predict_test > 0.7)
```

```
##
##     FALSE TRUE
##   0   176     3
##   1    20     3
```

```
spec_high_cutoff = 176 / (176 + 3)

spec_high_cutoff
```

```
## [1] 0.9832402
```

Answer: The board assigns more cost to a false negative than a false positive, and should therefore use a logistic regression cutoff less than 0.5.

Explanation

If the board used the model for parole decisions, a negative prediction would lead to a prisoner being granted parole, while a positive prediction would lead to a prisoner being denied parole. The parole board would experience more regret for releasing a prisoner who then violates parole (a negative prediction that is actually positive, or false negative) than it would experience for denying parole to a prisoner who would not have violated parole (a positive prediction that is actually negative, or false positive).

Decreasing the cutoff leads to more positive predictions, which increases false positives and decreases false negatives. Meanwhile, increasing the cutoff leads to more negative predictions, which increases false negatives and decreases false positives. The parole board assigns high cost to false negatives, and therefore should decrease the cutoff

5.5) Evaluating the Model on the Testing Set

Which of the following is the most accurate assessment of the value of the logistic regression model with a cutoff 0.5 to a parole board, based on the model's accuracy as compared to the simple baseline model?

The model is likely of value to the board, and using a different logistic regression cutoff is likely to improve the model's value.

Explanation

The model at cutoff 0.5 has 12 false positives and 11 false negatives, while the baseline model has 0 false positives and 23 false negatives. Because a parole board is likely to assign more cost to a false negative, the model at cutoff 0.5 is likely of value to the board.

From the previous question, the parole board would likely benefit from decreasing the logistic regression cutoffs, which decreases the false negative rate while increasing the false positive rate.

5.6) Evaluating the Modelon the Testing Set

Using the ROCR package, what is the AUC value for the model?

```
# predict_test = predict(mod_1, type = 'response', newdata = quality_test)

ROCR_pred_test = prediction(predict_test, test$violator)

auc = as.numeric(performance(ROCR_pred_test, "auc")@y.values)

auc
```

```
## [1] 0.8945834
```

Explanation

This can be obtained with the following code:

library(ROCR)

pred = prediction(predictions, test$violator)

as.numeric(performance(pred, “auc”)@y.values)

5.7) Evaluating the Model on the Testing Set

Describe the meaning of AUC in this context.

The probability the model can correctly differentiate between a randomly selected parole violator and a randomly selected parole non-violator.

Explanation

The AUC deals with differentiating between a randomly selected positive and negative example. It is independent of the regression cutoff selected.

6.1) Identifying Bias in Observational Data

Our goal has been to predict the outcome of a parole decision, and we used a publicly available dataset of parole releases for predictions. In this final problem, we'll evaluate a potential source of bias associated with our analysis. It is always important to evaluate a dataset for possible sources of bias.

The dataset contains all individuals released from parole in 2004, either due to completing their parole term or violating the terms of their parole. However, it does not contain parolees who neither violated their parole nor completed their term in 2004, causing non-violators to be underrepresented. This is called “selection bias” or “selecting on the dependent variable,” because only a subset of all relevant parolees were included in our analysis, based on our dependent variable in this analysis (parole violation). How could we improve our dataset to best address selection bias?

We should use a dataset tracking a group of parolees from the start of their parole until either they violated parole or they completed their term

Explanation

While expanding the dataset to include the missing parolees and labeling each as violator=0 would improve the representation of non-violators, it does not capture the true outcome, since the parolee might become a violator after 2004. Though labeling these new examples with violator=NA correctly identifies that we don't know their true outcome, we cannot train or test a prediction model with a missing dependent variable.

As a result, a prospective dataset that tracks a cohort of parolees and observes the true outcome of each is more desirable. Unfortunately, such datasets are often more challenging to obtain (for instance, if a parolee had a 10-year term, it might require tracking that individual for 10 years before building the model). Such a prospective analysis would not be possible using the 2004 National Corrections Reporting Program dataset.