Predicting Baseball Division Winners

Tom Corbin

today

Contents

First, we need to load the packages.

```
library("car")
library("tidyverse")
library("magrittr")
library("here")
library("janitor")
library("lubridate")
library("gridExtra")
library("readxl")
library("glmnet")
library("Lahman")
library("viridis")
library("lindia")
library("lme4")
library("caret")
library("pROC")
```

1. Datasets

We will be using some datasets about baseball from the 'Lahman' package. First, we will create a dataset called 'TeamSalaries' in which there is a row for each team and each year, and the variables are: i. 'Rostercost' = the sum of all player salaries for the given team in the given year. ii. 'meansalary' = the mean salary for that team that year. iii. 'rostersize' = the number of players listed that year for that team.

```
TeamSalaries<- Lahman::Salaries %>%
  group_by(teamID,yearID) %>%
  summarise(Rostercost = sum(salary),meansalary=mean(salary),rostersize=n())
```

```
## `summarise()` has grouped output by 'teamID'. You can override using the `.groups` argument.
```

We will then create a dataset called 'Teamdata' by taking the data from Lahman's 'Teams' dataset for the years 1984 to 2016 inclusive and adding to that data the variables in TeamSalaries.

```
Teamdata<- Teams %>% filter(yearID %in% 1984:2016) %>%
 left_join(TeamSalaries) %>%
 drop_na()
```

```
## Joining, by = c("yearID", "teamID")
```

Now we will create a dataset called DivWinners by removing all of the variables that are team or park identifiers in the dataset, as well as 'lgID', 'Rank','franchID','divID', 'WCWin','LgWin', and 'WSwin'.

```
DivWinners<- Teamdata %>% select(-c(2:6,12:14,41:42,46:48))
```

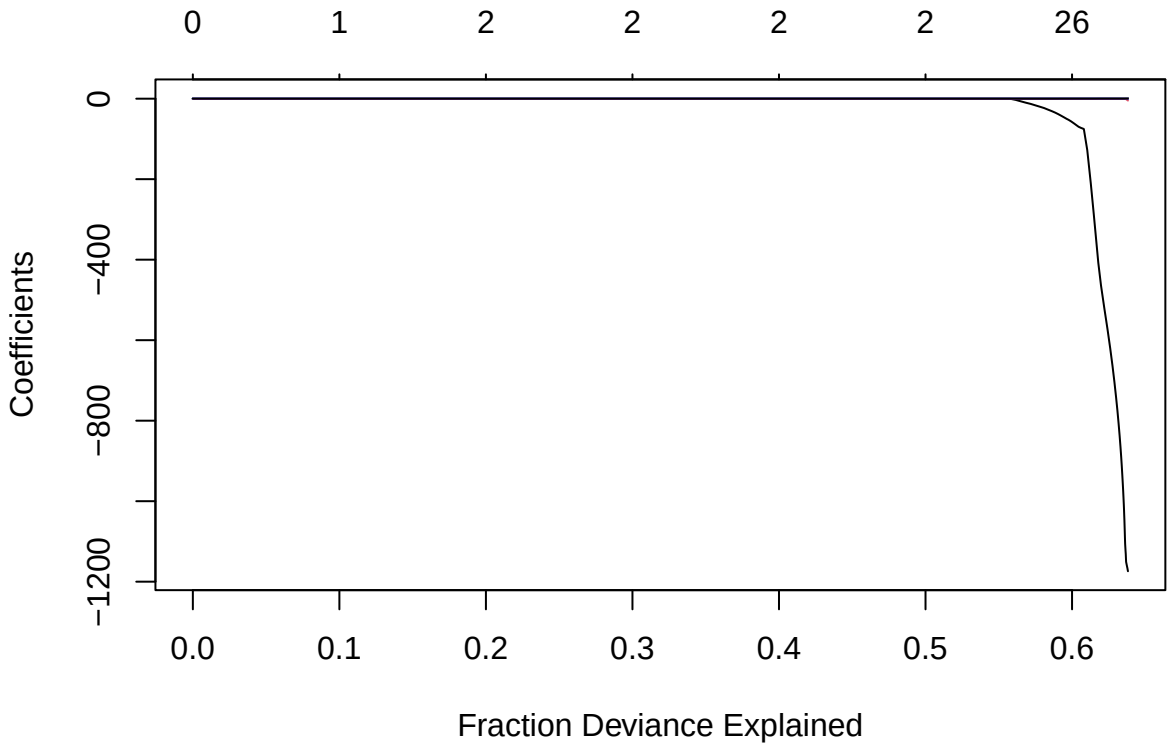
2. Lasso Regression for Logistic Regression

We will now split the resulting into a training and a testing set so that the variable 'DivWin' is balanced between the two datasets.

```
set.seed(123)
training.samples <- DivWinners$DivWin %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data  <- DivWinners[training.samples, ]
test.data <- DivWinners[-training.samples, ]
```

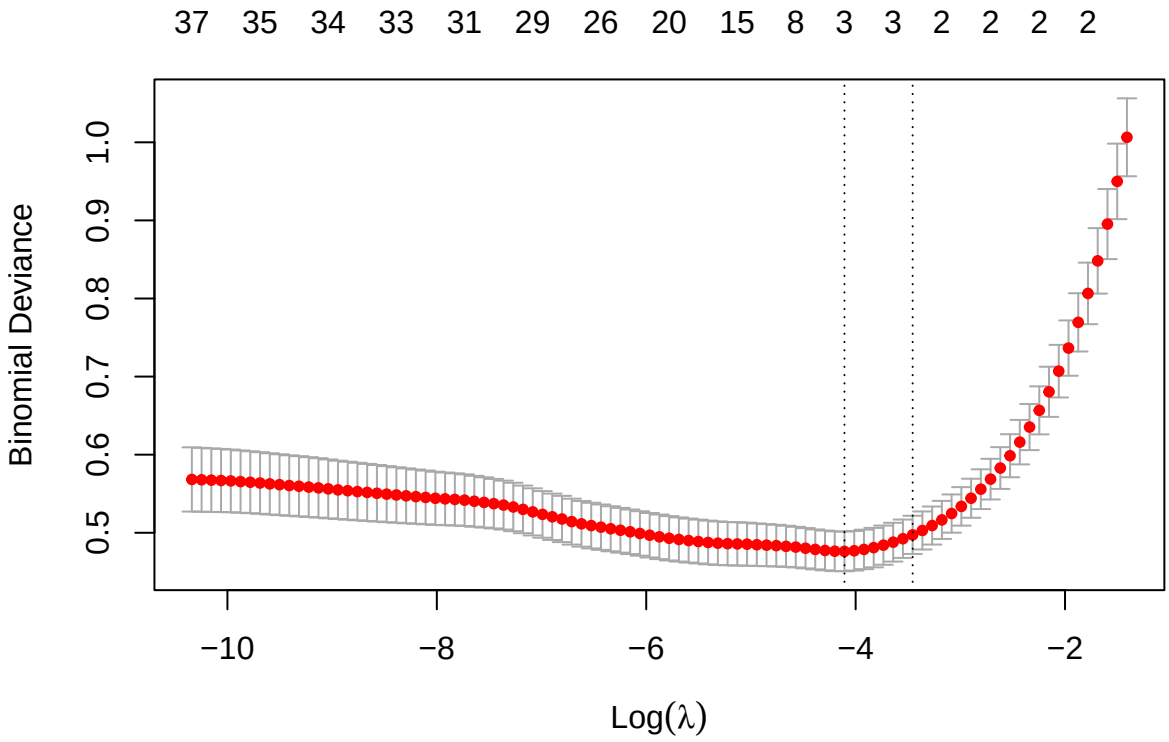
We will use the training data to fit a logistic regression model and plot residual deviance against number of predictors.

```
# removing response variable DivWin
DivWinvector<-as.vector(train.data$DivWin)

# expanding factors into dummy variables
DivWinpredict<-model.matrix(~.-1,train.data[,-c(6)])

# fitting model
DivWinfit<-glmnet(DivWinpredict, DivWinvector, family="binomial")

# plotting residual deviance against number of predictors
plot(DivWinfit,xvar="dev")
```



Next we will use cross-validation to choose a moderately conservative model.

```
set.seed(123)
DivWincv <- cv.glmnet(DivWinpredict, DivWinvector, family="binomial")
plot(DivWincv)
```




```
DivWin1sd <- coef(DivWinfit, s = DivWincv$lambda.1se)
DivWin1sd@Dimnames[[1]][1+DivWin1sd@i]
```

```
## [1] "(Intercept)" "W"           "L"           "attendance"
```

We will include the variables W (Wins), L (Losses), and attendace.

Now we will fit the model on the training data, predict on the testing data, then plot comparative ROC curves.

```
# fitting model
DivWinmodel <- glm(as.factor(DivWin) ~ W + L + attendance, family = "binomial", data = train.data)

# predicting on training and test data
predtrain <- predict(DivWinmodel, newdata = train.data, type = "response")
predtest <- predict(DivWinmodel, newdata = test.data, type = "response")

# plotting comparative ROC curves
roctrain <- roc(response = train.data$DivWin, predictor=predtrain, plot = TRUE,auc=TRUE)
```

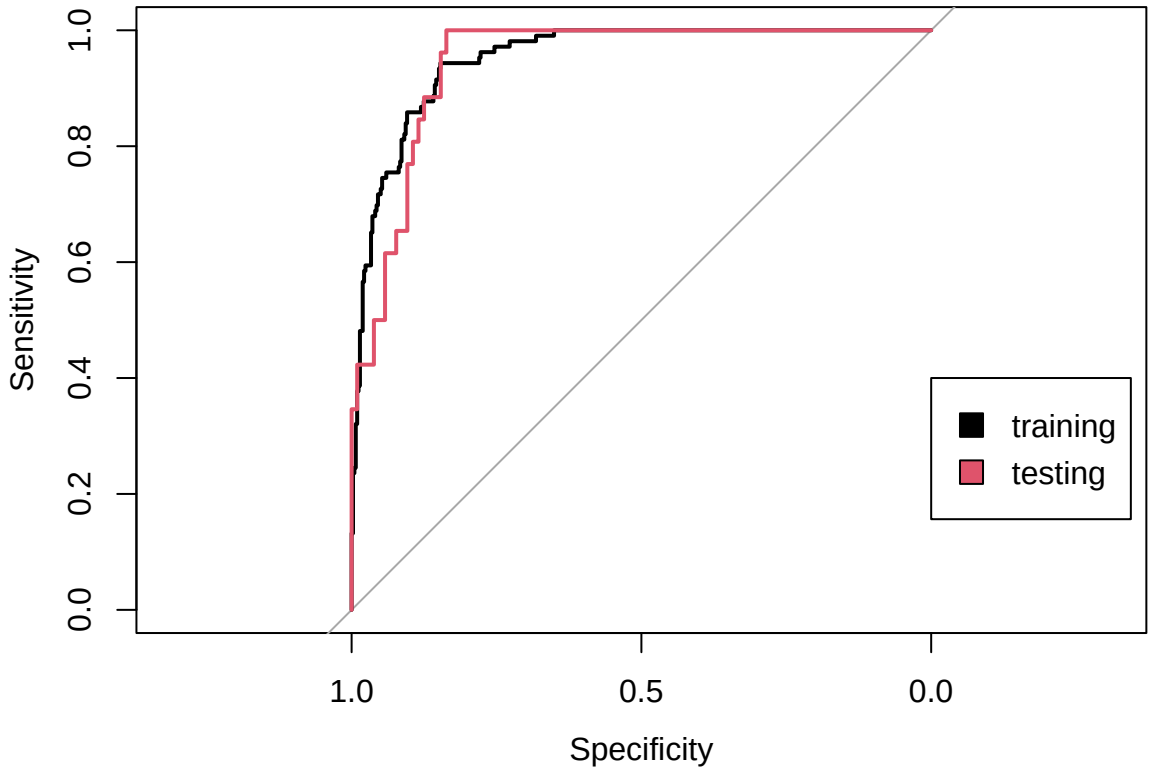
```
## Setting levels: control = N, case = Y
```

```
## Setting direction: controls < cases
```

```
roctest <- roc(response = test.data$DivWin, predictor = predtest, plot = TRUE, auc = TRUE, add = TRUE, c
```

```
## Setting levels: control = N, case = Y
## Setting direction: controls < cases
```

```
legend(0,0.4, legend = c("training","testing"), fill = 1:2)
```



The two curves are very similar, which suggests that the data is not overfitted to the training data. There are two small gaps between the curves, but it is still quite decent.

f. Next we will find Youden's index for the training data and calculate confusion matrices at this cutoff for both training and testing data.

```
# finding Youden's index
youdenDivWin <- coords(roctrain, "b", best.method = "youden", transpose = TRUE)
youdenDivWin
```

```
##   threshold specificity sensitivity
##   0.1836071   0.8468900   0.9433962
```

```
youdenDivWin[2] + youdenDivWin[3]
```

```
## specificity
##    1.790286
```

This tells us that we are better at predicting Y (division winners) than N (division losers), which we can see is the case here:

```
# creating confusion matrix for train.data
train.data$predDivWin <- ifelse(predict(DivWinmodel, train.data, type = "response") >= 0.1836071, "Y", "
table(train.data$predDivWin, train.data$DivWin)
```

```
##
##       N   Y
##   N 354   6
##   Y  64 100
```

Considering that our goal is to identify division winners, we have a very high sensitivity, correctly identifying 100/(100+6). The proportion of teams that are NOT division winners and are correctly labelled as such, is also quite high, with a specificity of 354/(354+64).

```
test.data$predDivWin <- ifelse(predict(DivWinmodel, test.data, type = "response") >= 0.1836071, "Y", "N
table(test.data$predDivWin, test.data$DivWin)
```

```
##
##      N  Y
##   N 87  1
##   Y 17 25
```

For the testing data, the sensitivity has actually increased slightly to 0.9615385 (25/25+1), and the specificity has decreased a little to 0.8365385 (87/87+17). So it seems that our training data is slightly more balanced, but overall, both the training and the testing data look very good.

In order to see how the model predicts on each division, we are going to calculate the sensitivity + specificity on the testing data as a function of divID and plot it as a barchart. To do this, we will first add 'divID' back into the dataset and create new training and testing data. Then, during the prediction stage, we will filter by division in order to find the Youden's index of each division. Then, having calculated the sensitivity + specificity of each division, we will place these figures back in the dataset in a new variable and plot the sensitivity + specificity of each division on a bar chart.

```
# adding 'divID' back into DivWinners dataset
DivWinners2<- Teamdata %>% select(-c(2:4,6,12:14,41:42,46:48))

# turning divID into factor
DivWinners2 <- DivWinners2 %>%
  mutate(divID = factor(divID))

# creating train and test datasets
set.seed(123)
training.samples <- DivWinners2$DivWin %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data2 <- DivWinners2[training.samples, ]
test.data2 <- DivWinners2[-training.samples, ]

# fitting model with additional divID variable
DivWinmodel2 <- glm(as.factor(DivWin) ~ W + L + attendance + divID, family = "binomial", data = train.da
```

Division C

```
# predicting on testing data and filtering by divID
predtestC <- predict(DivWinmodel2, newdata = test.data2[test.data2$divID == "C",], type = "response")

# finding Youden's index of test data for division C
roctestC <- roc(response = test.data2[test.data2$divID == "C",]$DivWin,
                predictor = predtestC,
                auc = TRUE
                )
```

```
## Setting levels: control = N, case = Y
```

```
## Setting direction: controls < cases
```

```
youdenDivWinC <- coords(roctestC, "b", best.method = "youden", transpose = TRUE)
youdenDivWinC
```

```
##   threshold specificity sensitivity
##   0.1909609   0.8780488   1.0000000
```

```
youdenDivWinC[2] + youdenDivWinC[3]
```

```
## specificity
##    1.878049
```

Division E

```
# predicting on testing data and filtering by divID
predtestE <- predict(DivWinmodel2, newdata = test.data2[test.data2$divID == "E",], type = "response")

# finding Youden's index for test data for division E
roctestE <- roc(response = test.data2[test.data2$divID == "E",]$DivWin,
                predictor = predtestE,
                auc = TRUE
                )
```

```
## Setting levels: control = N, case = Y
```

```
## Setting direction: controls < cases
```

```
youdenDivWinE <- coords(roctestE, "b", best.method = "youden", transpose = TRUE)
youdenDivWinE
```

```
##   threshold specificity sensitivity
##   0.3743142   0.8484848   1.0000000
```

```
youdenDivWinE[2] + youdenDivWinE[3]
```

```
## specificity
##    1.848485
```

Division W

```
# predicting on testing data and filtering by divID
predtestW <- predict(DivWinmodel2, newdata = test.data2[test.data2$divID == "W",], type = "response")

# finding Youden's index for test data for division W
roctestW <- roc(response = test.data2[test.data2$divID == "W",]$DivWin,
                predictor = predtestW,
                auc = TRUE
                )
```

```
## Setting levels: control = N, case = Y
```

```
## Setting direction: controls < cases
```

```
youdenDivWinW <- coords(roctestW, "b", best.method = "youden", transpose = TRUE)
youdenDivWinW
```

```
##   threshold specificity sensitivity
##   0.3401528   0.9000000   1.0000000
```

```
youdenDivWinW[2] + youdenDivWinW[3]
```

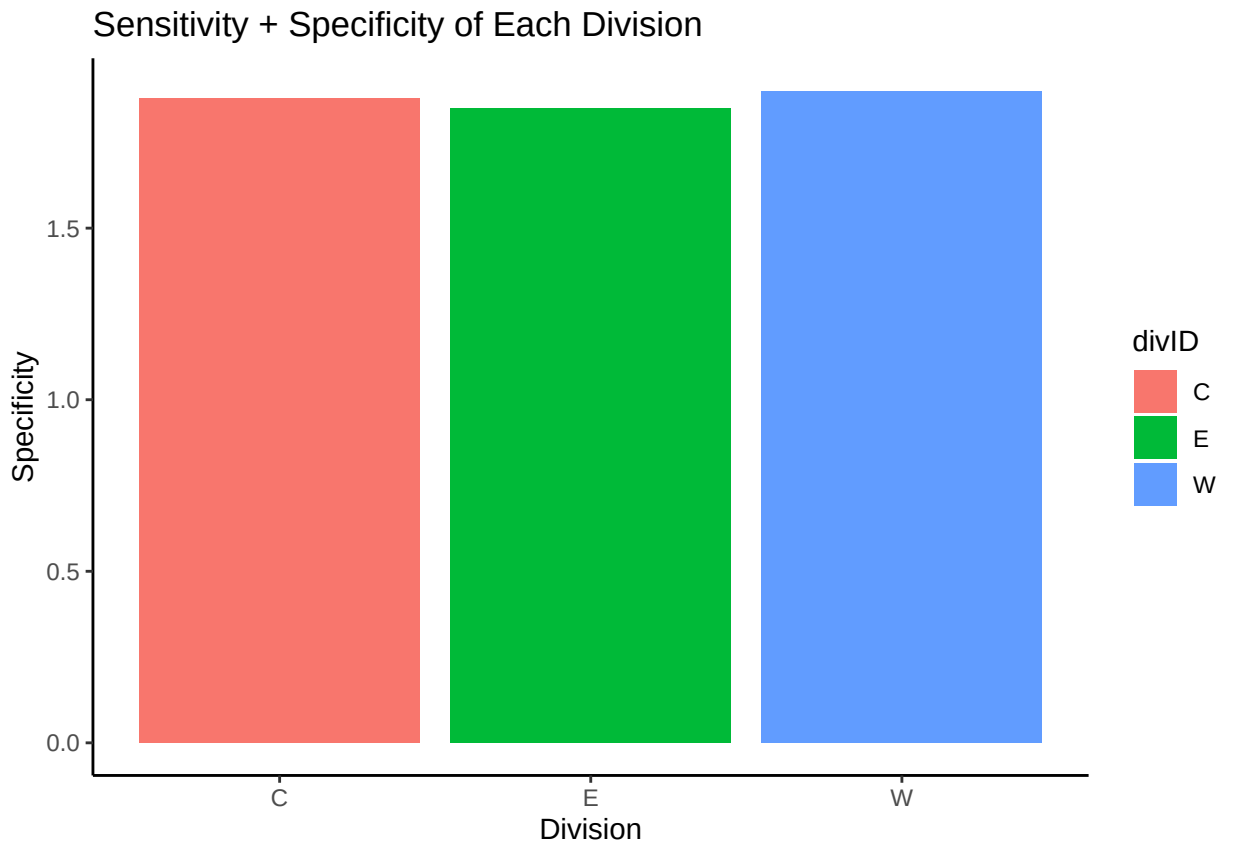
```
## specificity
##         1.9
```

Plotting sensitivity + specificity of each division

```
DivWinners2 %>%
  mutate(test_specificity = case_when(            # adding test_specificity column
    divID == "C" ~ 1.878049,
    divID == "E" ~ 1.848485,
    TRUE ~ 1.9
  )) %>%
  group_by(divID) %>%
  summarise(test_specificity) %>%                 # isolating test_specificity
  unique() %>%
  ggplot(aes(x = divID,                         # plotting sensitivity + specificity of each division
             y= test_specificity,
             fill = divID)
         ) +
  geom_col() +
```

```
  labs(title = "Sensitivity + Specificity of Each Division",
       x = "Division",
       y = "Specificity") +
  theme_classic()
```

```
## `summarise()` has grouped output by 'divID'. You can override using the `.groups` argument.
```



Here we can see that the sensitivity and specificity on the testing data for each division are almost equal. This is positive as it reveals that the testing data does not perform much better on one group than another.