Hands on logistic regression and diagnostics of Titanic Dataset

2024-03-15

Today's Gameplan

- Revisit the titanic data. Code along with me as I run the logistic regression model.
- **Primary learning goal**: understanding assessment of model fit for logistic regression.
  - **Different than assessing model fit for linear regression.**
  - **Different from doing odds ratio interpretation of coefficients, as we did previously.**
  - Instead of predicting the level of some continuous outcome variable, we are trying to predict whether an event happened or didn't.
    - In other words, probability of survival. Can we accurately predict who survives and who doesn't?
    - Generally, how accurately can we predict $Y = 1$.
    - **Recall: The outcome of a logistic regression model is not Y, it's P(Y=1), the probability of the event occurring.**
    - If $P(Y = 1) > 0.5$ in our model, then we "predict" $Y = 1$, and predict $Y = 0$ otherwise.

Data Exploration

```
data(titanic_train)
head(titanic_train)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                  Name    Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                              Heikkinen, Miss. Laina female  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                            Allen, Mr. William Henry   male  35     0     0
## 6                                    Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500             S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250             S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500             S
## 6           330877  8.4583             Q
```

Fitting a Logistic Regression Model

The term on the left here is literally the "log odds".

- It's the natural log of the probability of the event occurring divided by the probability it doesn't occur.

$ln\frac{P(Y=1)}{1-P(Y=1)} = \beta_0 + \beta_1 Class + \beta_2 Sex + \beta_3 Age + \epsilon$

When this formula is rearranged, it becomes

$P(Y=1) = \frac{1}{1+e^{-(\beta_0+\beta_1 Class+\beta_2 Sex+\beta_3 Age+\epsilon)}}$

- We use command `glm()` instead of `lm()` and set an option of `family = "binomial"`.
  - In statistics, a "binomial" is a variable that takes on one of two values. A *BI**NARY variable!

```
model <- glm(Survived ~ Pclass + Sex + Age,data = titanic_train, family = 'binomial')
summary(model)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age, family = "binomial",
##     data = titanic_train)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.056006   0.502128  10.069  < 2e-16 ***
## Pclass      -1.288545   0.139259  -9.253  < 2e-16 ***
## Sexmale     -2.522131   0.207283 -12.168  < 2e-16 ***
## Age         -0.036929   0.007628  -4.841 1.29e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 964.52  on 713  degrees of freedom
## Residual deviance: 647.29  on 710  degrees of freedom
##   (177 observations deleted due to missingness)
## AIC: 655.29
##
## Number of Fisher Scoring iterations: 5
```

The model output.

- We don't interpret these the same way, though! This is not linear regression!

Recall, we exponentiate coefficients if we want to interpret them properly.

```
exp(model$coefficients)
```

```
##  (Intercept)       Pclass      Sexmale          Age
## 156.96238449   0.27567157   0.08028834   0.96374454
```

Two (of several) Diagnostics of our Logistic Regression Model: Confusion Matrix and ROC Curve

We will need these two packages

```
library(caret) # For confusion matrix
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(plotROC) # for ROC plot
```

We are going to see how well the model predicts survival. (Y = 1)

1. We need to develop "predicted probabilities" of Survival from our model.
   - The `predict()` command predicts outcomes on any data you feed it, based on a model you built.
   - You could use new data in `predict()`! We're just going to feed it our current data.
   - **You must specify `type = "response"`**
     - You can read more about why if you do run `?predict.glm`

```
# Predict probabilities on the dataset
titanic_train$predicted_survival <- predict.glm(model, newdata = titanic_train,
                                                type = 'response')
```

2. We then convert these probabilities to outcomes.
   - If probability of survival > 0.5, we predict survival.
   - If probability of survival <= 0.5, we predict "not" survival.

```
# Convert probabilities to binary outcome based on threshold 0.5
titanic_train$predicted_survival <- ifelse(titanic_train$predicted_survival > 0.5,1,0)

# we can immediately check the accuracy by taking the average amount of times the predicted_survival =
# since there is at least one NA in our data, we must use `na.rm=TRUE`, else we get an NA mean.
mean(titanic_train$Survived==titanic_train$predicted_survival,na.rm=T)
```

```
## [1] 0.7885154
```

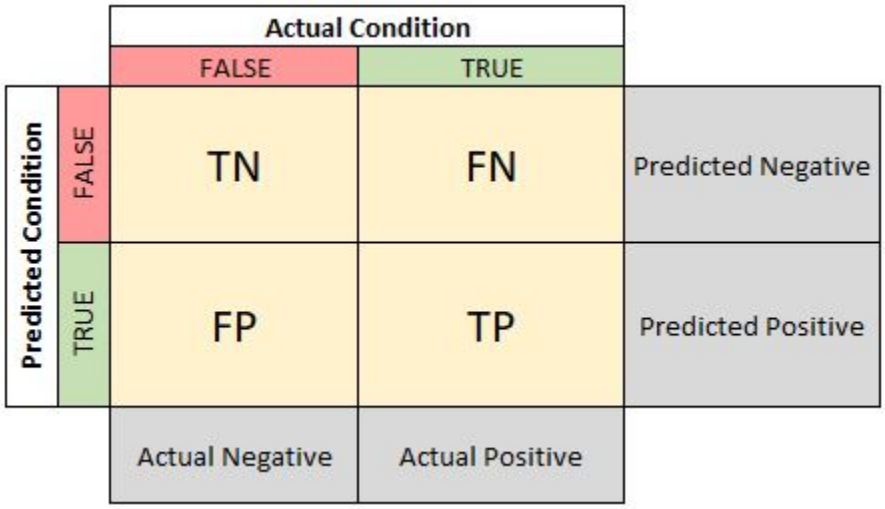
3. We create our confusion matrix
   - **You must read the predicted Y = 1 and actual Y = 1 as factors.**
   - YOU MUST IDENTIFY WHAT THE "YES" IS IN YOUR DATA USING 'positive=' OR ELSE IT WILL ASSUME THE FIRST THING IT SEES IS "YES"
   - The first time I did this, it treated 0 as the "positive" outcome and mixed up its measurements.
   - This is because it will put the first thing it sees first. The first observation is one where Y = 0 (a death).

```
# Confusion Matrix
conf_matrix <- confusionMatrix(factor(titanic_train$predicted_survival),
                               factor(titanic_train$Survived),
                               positive = '1')
conf_matrix
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction   0   1
##          0 356  83
##          1  68 207
##
##                Accuracy : 0.7885
##                  95% CI : (0.7567, 0.8179)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.558
##
##  Mcnemar's Test P-Value : 0.2546
##
##             Sensitivity : 0.7138
##             Specificity : 0.8396
##          Pos Pred Value : 0.7527
##          Neg Pred Value : 0.8109
##              Prevalence : 0.4062
##          Detection Rate : 0.2899
##    Detection Prevalence : 0.3852
##       Balanced Accuracy : 0.7767
##
##        'Positive' Class : 1
##
```

Interpreting a Confusion Matrix



- Row-wise (prediction), 0 is prediction of death, 1 is prediction of survival. - Top row predicted to have died, bottom row predicted to have survived.
- Column-wise (reference), 0 is actually died, 1 is actually survived.
  - Left column died, right column survived.
- Accuracy: How often the model is correct.

Recall Conditional Probabilities: $P(A|B) = P(A)/P(AandB)$

- Sensitivity: The "True positive rate"
  - Conditional probability $P(\hat{Y} = 1|Y = 1)$

  - Probability it predicts survival given survival actually occurred.
  - $207/(207 + 83) = .7138$
- Specificity: The "True negative rate"
  - Conditional probability $P(\hat{Y} = 0|Y = 0)$
  - Probability it predicts non-survival given non-survival actually occured.
  - $356/(356 + 68) = .8396$
- True positive (sensitivity)
- False positive (1 - specificity)
  - Specificity is "true negative".
  - Intuition behind this math:
    * Total Negatives (deaths) = people properly identified as deaths + people mistakenly identified as survived
    * AKA Total Negatives = true negatives + false positives

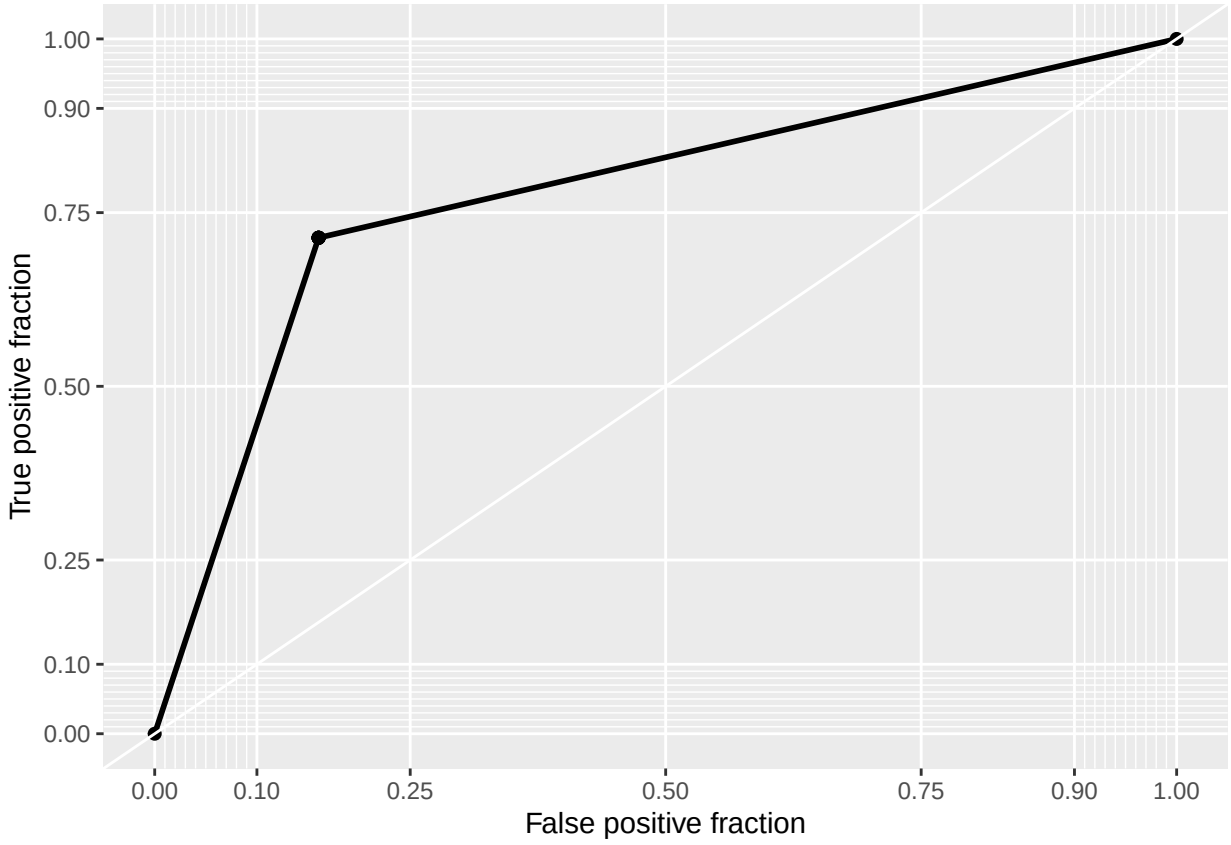
A confusion matrix with high values on the diagonal (true positives and true negatives) and low values on the off-diagonal (false positives and false negatives) indicate a good model.

Building and Interpreting an ROC Curve

5. We build our ROC Curve as another test of how well our logistic regression model fits.

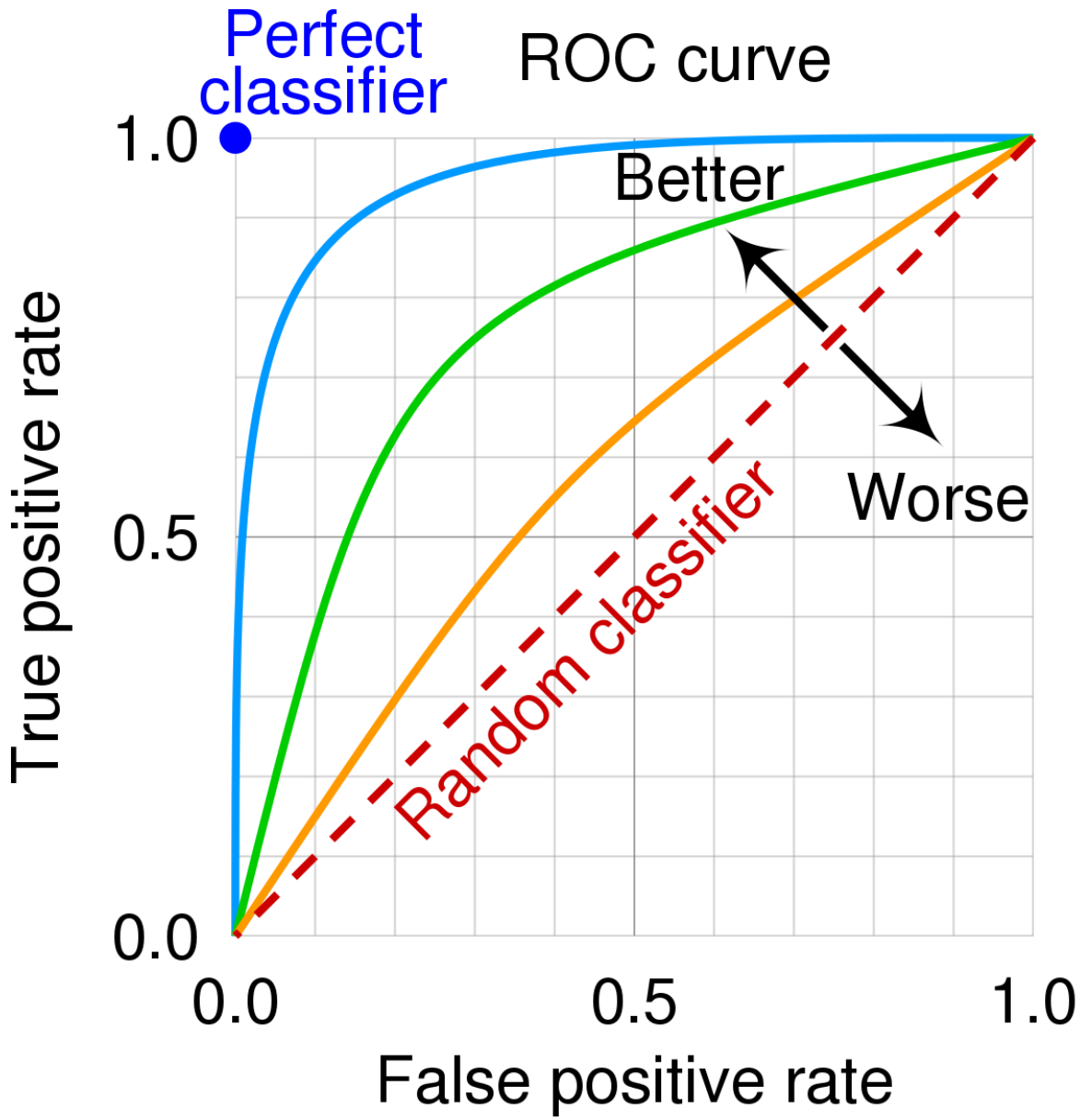
- plotROC is a package that works with ggplot, but it requires that we define, inside our `aes()` like
  - `aes(m = "predicted value variable", d = "actual value variable")`

```
library(plotROC)
rocplot <- ggplot(titanic_train,aes(m=predicted_survival,d=Survived)) + geom_roc(labels=F) + style_roc(
rocplot
```



Interpreting our ROC Curve

- Helps us visualize how well our model discriminates between Y = 1 and Y = 0 (survived or not).
- "Receiver Operating Characteristics" (ROC) is a graph where - True positive rate is plotted on the y axis. - False positive rate is plotted on the x axis. - The point on the line from our ggplot corresponds to our actual Sensitivity (true positive) and our actual (1-Specificity) (false positive)
  - y on the point = .7138, same sensitivity shown in the confusion matrix
  - x on the line = 1 - .8396 = .1604, 1 - specificity from our confusion matrix



- Fully diagonal would mean our model is no better at predicting than a random guess.
- The further bowed up to the left it looks, the better it is at predicting.
- If it were bowed down to the right, it would be worse at predicting than a random guess.