CSCI E-106 Titanic Survival Tutorial Solution

Ian Kelk

10 December 2025

Contents

Classify whether a passenger on board the RMS Titanic survived

Classify whether a passenger on board the maiden voyage of the RMS Titanic in 1912 survived given their age, sex and class. We will use the provided `Titanic_Survival_Data.csv` dataset and build:

- A champion logistic regression model for survival.
- At least one challenger model (here: a classification tree).
- Train/test evaluation, plus discussion of limitations and monitoring.

This R Markdown is written as a basic tutorial solution.

0. Setup and libraries

```r
library(tidyverse)   # data wrangling & plotting
library(caret)       # train/test split, confusion matrices
library(rpart)       # classification trees
library(rpart.plot)  # nice tree plots
library(pROC)        # ROC curves and AUC
```

Dataset description

The dataset's variables are:

Variable	Description
pclass	Passenger Class: 1st, 2nd or 3rd
survived	Survival Status: 0 = No, 1 = Yes
name	Name of the passenger
Sex	Sex
sibsp	Number of siblings or spouses aboard
parch	Number of parents or children aboard
ticket	Ticket number
fare	Passenger fare
cabin	Cabin number (e.g., `C85` = deck C, cabin 85)
embarked	Port of embarkation: C = Cherbourg, S = Southampton, Q = Queenstown
boat	Lifeboat ID (if passenger survived)
body	Body number (if passenger did not survive and body was recovered)
home.dest	Intended home destination of the passenger

We will treat `survived` as the binary response and use predictors such as `pclass`, `sex`, `age`, and others.

```r
# Adjust the file name/path if needed
titanic_raw <- read_csv("Titanic_Survival_Data.csv")

# Standardize column names to lower case for convenience
titanic <- titanic_raw %>%
  rename_with(tolower)

glimpse(titanic)
```

```
## Rows: 1,310
## Columns: 14
## $ pclass    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ survived  <dbl> 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, ~
## $ name      <chr> "Allen, Miss. Elisabeth Walton", "Allison, Master. Hudson Tr~
## $ sex       <chr> "female", "male", "female", "male", "female", "male", "femal~
## $ age       <dbl> 29.0000, 0.9167, 2.0000, 30.0000, 25.0000, 48.0000, 63.0000,~
## $ sibsp     <dbl> 0, 1, 1, 1, 1, 0, 1, 0, 2, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ parch     <dbl> 0, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, ~
## $ ticket    <chr> "24160", "113781", "113781", "113781", "113781", "19952", "1~
## $ fare      <dbl> 211.3375, 151.5500, 151.5500, 151.5500, 151.5500, 26.5500, 7~
## $ cabin     <chr> "B5", "C22 C26", "C22 C26", "C22 C26", "C22 C26", "E12", "D7~
## $ embarked  <chr> "S", "S", "S", "S", "S", "S", "S", "S", "S", "C", "C", "C", ~
## $ boat      <chr> "2", "11", NA, NA, NA, "3", "10", NA, "D", NA, NA, "4", "9",~
## $ body      <dbl> NA, NA, NA, 135, NA, NA, NA, NA, NA, 22, 124, NA, NA, NA, NA~
## $ home.dest <chr> "St Louis, MO", "Montreal, PQ / Chesterville, ON", "Montreal~
```

The raw dataset contains 1,310 observations and 14 variables. It includes demographic information (age, sex, passenger class), family structure (siblings/spouses, parents/children), and travel details (fare, cabin, ticket, home destination, port of embarkation). A quick look at the data shows that most core fields are present for nearly all passengers, but some variables such as `age`, `cabin`, `boat`, `body`, and `home.dest` have noticeable missing values, which we will need to account for when building models.

Executive Summary

The goal of this project is to predict the probability that a passenger on the RMS Titanic survived, using information that would plausibly be available at the time of sailing (such as passenger class, age, sex, family members aboard, fare, and port of embarkation). This kind of model could be used in a broader risk-management context to understand which groups were at greatest risk and how different passenger characteristics related to survival.

Our main "champion" model is a logistic regression that uses passenger class, sex, age, number of siblings/spouses, and port of embarkation as predictors. On a held-out test set, this model correctly classifies about 81% of passengers, with sensitivity (recall for survivors) of about 66% and specificity (correctly identifying non-survivors) of about 91%. The area under the ROC curve is roughly 0.85, indicating that the model is generally good at ranking survivors above non-survivors. The key drivers of higher survival probability are being female, traveling in 1st or 2nd class rather than 3rd, being younger, having fewer siblings/spouses aboard, and departing from certain ports.

As a challenger, we also fit a classification tree using the same set of predictors. The tree achieves similar overall accuracy (about 82–83%) and a slightly higher specificity, but lower sensitivity and a slightly lower AUC (about 0.83). Because the logistic model offers more stable performance and more interpretable coefficients, we select logistic regression as our champion model and treat the tree as a benchmark. Both models are limited by the nature of the Titanic data: it is historical, not representative of modern populations, and reflects strong social and economic biases (for example, survival advantages for women and higher-class passengers), so these models should not be naively generalized beyond this context.

I. Introduction (5 points)

In this section, we introduce:

- The business problem.
- The modeling approach.
- A high-level overview of the data and methods.

The goal is to predict the probability of survival for Titanic passengers given their observed characteristics. We focus on logistic regression as the champion model, with at least one challenger model for comparison.

Logistic regression is a natural choice here because the outcome of interest, `survived`, is binary (0 = no, 1 = yes). The model directly estimates the probability of survival as a function of the predictors, and the coefficients have a clear interpretation in terms of changes in log-odds or odds ratios. At the same time, it is valuable to have a more flexible, non-parametric challenger model such as a decision tree. The tree can capture nonlinear interactions and simple rule-based structures in the data, giving us a different perspective on the same problem and a benchmark to compare against our logistic "champion" in terms of both accuracy and interpretability.

II. Description of the data and quality (15 points)

Here we explore the data, check for missing values, and think about how to encode variables.

2.1 Basic summaries and missingness

```r
summary(titanic)
```

```
##      pclass         survived         name               sex           
##  Min.   :1.000   Min.   :0.000   Length:1310        Length:1310       
##  1st Qu.:2.000   1st Qu.:0.000   Class :character   Class :character  
##  Median :3.000   Median :0.000   Mode  :character   Mode  :character  
##  Mean   :2.295   Mean   :0.382                                        
##  3rd Qu.:3.000   3rd Qu.:1.000                                        
##  Max.   :3.000   Max.   :1.000                                        
##  NA's   :1       NA's   :1                                            
##       age             sibsp            parch          ticket
##  Min.   : 0.1667   Min.   :0.0000   Min.   :0.000   Length:1310
##  1st Qu.:21.0000   1st Qu.:0.0000   1st Qu.:0.000   Class :character
##  Median :28.0000   Median :0.0000   Median :0.000   Mode  :character
##  Mean   :29.8811   Mean   :0.4989   Mean   :0.385
##  3rd Qu.:39.0000   3rd Qu.:1.0000   3rd Qu.:0.000
##  Max.   :80.0000   Max.   :8.0000   Max.   :9.000
##  NA's   :264       NA's   :1        NA's   :1
##       fare             cabin             embarked             boat
##  Min.   :  0.000   Length:1310        Length:1310        Length:1310
##  1st Qu.:  7.896   Class :character   Class :character   Class :character
##  Median : 14.454   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 33.295
##  3rd Qu.: 31.275
##  Max.   :512.329
##  NA's   :2
##       body        home.dest
##  Min.   :  1.0   Length:1310
##  1st Qu.: 72.0   Class :character
##  Median :155.0   Mode  :character
##  Mean   :160.8
##  3rd Qu.:256.0
##  Max.   :328.0
##  NA's   :1189
```

```r
colSums(is.na(titanic))
```

```
##    pclass  survived      name       sex       age     sibsp     parch    ticket
##         1         1         1         1       264         1         1         1
##      fare     cabin  embarked      boat      body home.dest
##         2      1015         3       824      1189       565
```

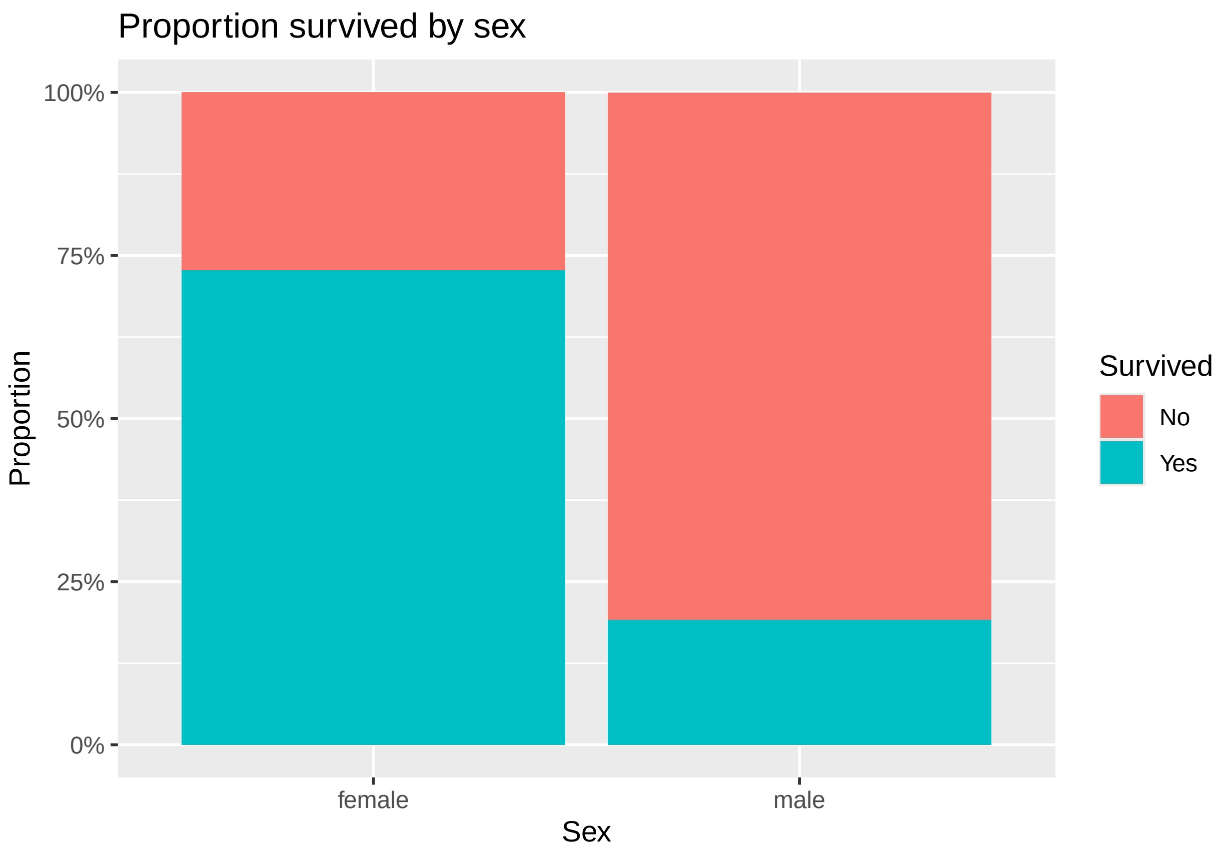
The summary and missing-value counts show that some variables are much more complete than others. Core variables such as `pclass`, `survived`, `sex`, `sibsp`, `parch`, and `fare` have almost no missing data (only a handful of missing values each), so they are good candidates for modeling. In contrast, `age` is missing for a sizable subset of passengers, and variables like `cabin`, `boat`, `body`, and `home.dest` are missing for the majority of passengers. This level of missingness would either require substantial imputation or lead to a large loss of data if used directly. In addition, variables like `name`, `ticket`, and the raw `body` number behave more like identifiers than meaningful predictors; they are unlikely to help the model generalize and may even encourage overfitting. For these reasons, we focus our modeling on the cleaner, more interpretable variables and treat the heavily missing or ID-like fields as auxiliary information rather than predictors.

2.2 Simple visual exploration

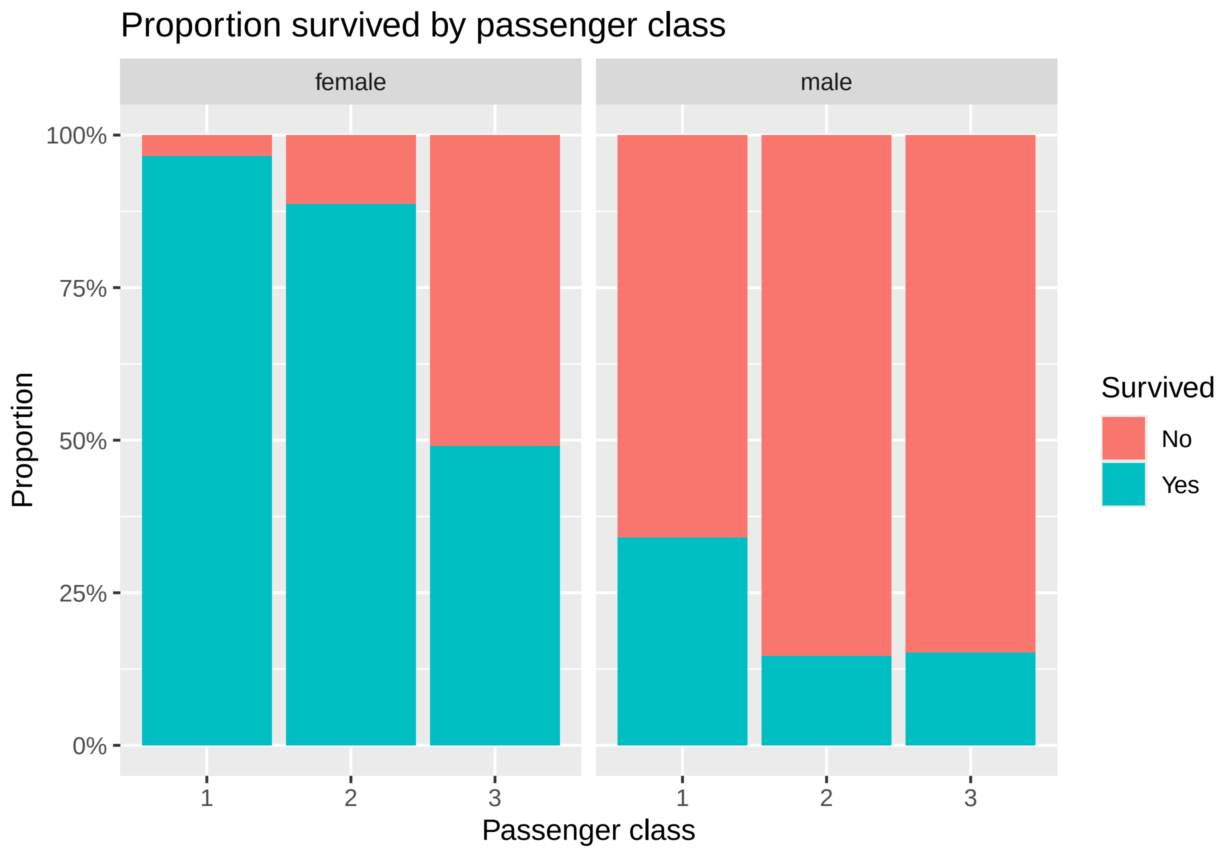
We are particularly interested in how survival varies by sex and passenger class.

```r
# Make a working copy and ensure key variables are typed correctly
titanic <- titanic %>%
  mutate(
    pclass = factor(pclass),
    sex = factor(sex),
    embarked = factor(embarked),
    survived = as.integer(survived),
    survived = ifelse(survived == 1, 1, 0),
    survived_factor = factor(survived, levels = c(0, 1),
                             labels = c("No", "Yes"))
  ) %>%
  tidyr::drop_na(sex, pclass, survived)

# Survival proportion by sex
titanic %>%
  ggplot(aes(x = sex, fill = survived_factor)) +
  geom_bar(position = "fill") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(
    title = "Proportion survived by sex",
    x = "Sex",
    y = "Proportion",
    fill = "Survived"
  )
```



```r
# Survival proportion by class and sex
titanic %>%
  ggplot(aes(x = pclass, fill = survived_factor)) +
  geom_bar(position = "fill") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(
    title = "Proportion survived by passenger class",
    x = "Passenger class",
    y = "Proportion",
    fill = "Survived"
  ) +
  facet_wrap(~ sex)
```



The bar plots confirm well-known Titanic patterns. A much larger proportion of females survived compared to males: among women, the “Yes” portion of the bar dominates, whereas for men the “No” portion is much larger. When we break survival down by passenger class and sex, we see that 1st-class passengers have the highest survival rates, especially 1st-class women, while 3rd-class passengers, particularly 3rd-class men, have the lowest survival rates. These patterns suggest that both sex and passenger class are strong predictors of survival and motivate including them prominently in our models.

2.3 Build a modeling dataset

We drop variables that are essentially identifiers or have very high missingness, and we remove rows with missing values in key predictors.

```r
titanic_model <- titanic %>%
  select(
    survived, survived_factor,
    pclass, sex, age, sibsp, parch, fare, embarked
  ) %>%
  filter(
    !is.na(age),
    !is.na(fare),
    !is.na(embarked)
  )

glimpse(titanic_model)
```

```
## Rows: 1,043
## Columns: 9
## $ survived        <dbl> 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, ~
## $ survived_factor <fct> Yes, Yes, No, No, No, Yes, Yes, No, Yes, No, No, Yes, ~
## $ pclass          <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ sex             <fct> female, male, female, male, female, male, female, male~
## $ age             <dbl> 29.0000, 0.9167, 2.0000, 30.0000, 25.0000, 48.0000, 63~
## $ sibsp           <dbl> 0, 1, 1, 1, 1, 0, 1, 0, 2, 0, 1, 1, 0, 0, 0, 0, 0, 0, ~
## $ parch           <dbl> 0, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, ~
## $ fare            <dbl> 211.3375, 151.5500, 151.5500, 151.5500, 151.5500, 26.5~
## $ embarked        <fct> S, S, S, S, S, S, S, S, S, C, C, C, C, S, S, C, C, C, ~
```

In building the modeling dataset we kept variables that are both interpretable and reasonably complete: `pclass`, `sex`, `age`, `sibsp`, `parch`, `fare`, and `embarked`, along with the outcome `survived`. We dropped `name`, `ticket`, and `home.dest` because they function primarily as identifiers or detailed text fields that are unlikely to generalize well in a simple tabular model. We also dropped `cabin`, `boat`, and `body` because they are missing for most passengers and may not be available at prediction time (for example, `boat` and `body` are only known after the outcome is realized). Using these variables would either greatly reduce our sample size or introduce unrealistic "cheating" by using post-outcome information, so excluding them leads to a cleaner, more realistic modeling dataset.

III. Model Development Process (15 points)

We now split the data into training and test sets and build candidate models.

3.1 Train/test split (70/30, set.seed(1023))

```r
set.seed(1023)

train_index <- createDataPartition(
  titanic_model$survived_factor,
  p = 0.7,
  list = FALSE
)

train_data <- titanic_model[train_index, ]
test_data  <- titanic_model[-train_index, ]

prop.table(table(train_data$survived_factor))
```

```
##
##        No       Yes
## 0.5923393 0.4076607
```

```r
prop.table(table(test_data$survived_factor))
```

```
##
##        No       Yes
## 0.5929487 0.4070513
```

The train/test split preserves the overall class balance in the data. In the training set, a bit under 60% of passengers did not survive and a bit over 40% did survive; the test set has almost the same proportions. This similarity is desirable because it means the model is trained and evaluated on datasets with comparable levels of class imbalance, making performance metrics on the test set more representative.

3.2 Baseline logistic regression (age, sex, class)

We start with the minimal set of predictors mentioned in the project description: `age`, `sex`, and `pclass`.

```r
logit_simple <- glm(
  survived ~ pclass + sex + age,
  data = train_data,
  family = binomial
)

summary(logit_simple)
```

```
##
## Call:
## glm(formula = survived ~ pclass + sex + age, family = binomial,
##     data = train_data)
```

```
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.667400   0.391260   9.373  < 2e-16 ***
## pclass2     -1.426765   0.269101  -5.302 1.15e-07 ***
## pclass3     -2.446184   0.273189  -8.954  < 2e-16 ***
## sexmale     -2.361338   0.196203 -12.035  < 2e-16 ***
## age         -0.039601   0.007582  -5.223 1.76e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 988.31  on 730  degrees of freedom
## Residual deviance: 696.03  on 726  degrees of freedom
## AIC: 706.03
##
## Number of Fisher Scoring iterations: 4
```

In the baseline logistic regression using `pclass`, `sex`, and `age`, the key coefficients are statistically significant at conventional levels. The negative coefficient for `sexmale` indicates that, holding class and age constant, males have much lower log-odds of survival than females; in terms of odds ratios, the odds of survival for a male are only a small fraction of those for an otherwise similar female. The coefficients for `pclass2` and `pclass3` are also negative, so passengers in 2nd and especially 3rd class have substantially lower odds of survival than those in 1st class. The age coefficient is negative as well, implying that each additional year of age slightly reduces the log-odds of survival. More generally, a positive logistic regression coefficient means that an increase in that predictor (or switching to that category) increases the log-odds of survival (odds ratio > 1), while a negative coefficient decreases the log-odds (odds ratio < 1).

3.3 Expanded logistic regression and variable selection

Next, we allow more predictors and use stepwise selection (AIC-based) to choose a more flexible model.

```r
logit_full <- glm(
  survived ~ pclass + sex + age + sibsp + parch + fare + embarked,
  data = train_data,
  family = binomial
)

summary(logit_full)
```

```
##
## Call:
## glm(formula = survived ~ pclass + sex + age + sibsp + parch +
##     fare + embarked, family = binomial, data = train_data)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.362789   0.508280   8.583  < 2e-16 ***
## pclass2     -1.121766   0.316867  -3.540  0.00040 ***
## pclass3     -2.078037   0.328274  -6.330 2.45e-10 ***
## sexmale     -2.505129   0.211732 -11.832  < 2e-16 ***
## age         -0.042467   0.007984  -5.319 1.04e-07 ***
## sibsp       -0.337129   0.130742  -2.579  0.00992 **
## parch        0.017606   0.131122   0.134  0.89319
## fare         0.001310   0.002304   0.569  0.56969
## embarkedQ   -1.641916   0.543518  -3.021  0.00252 **
## embarkedS   -0.777139   0.260527  -2.983  0.00285 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 988.31  on 730  degrees of freedom
## Residual deviance: 673.68  on 721  degrees of freedom
## AIC: 693.68
##
## Number of Fisher Scoring iterations: 5
```

```
logit_step <- step(logit_full, direction = "both", trace = FALSE)

summary(logit_step)
```

```
##
## Call:
## glm(formula = survived ~ pclass + sex + age + sibsp + embarked,
##     family = binomial, data = train_data)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.508274   0.456337   9.879  < 2e-16 ***
## pclass2     -1.196358   0.290801  -4.114 3.89e-05 ***
## pclass3     -2.168782   0.287466  -7.544 4.54e-14 ***
## sexmale     -2.522510   0.208772 -12.083  < 2e-16 ***
## age         -0.042743   0.007967  -5.365 8.10e-08 ***
## sibsp       -0.318733   0.124330  -2.564  0.01036 *
## embarkedQ   -1.673701   0.541561  -3.091  0.00200 **
## embarkedS   -0.805343   0.256307  -3.142  0.00168 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 988.31  on 730  degrees of freedom
## Residual deviance: 674.09  on 723  degrees of freedom
## AIC: 690.09
##
## Number of Fisher Scoring iterations: 5
```

```
champion_model <- logit_step
champion_model
```

```
##
## Call:  glm(formula = survived ~ pclass + sex + age + sibsp + embarked,
##     family = binomial, data = train_data)
##
## Coefficients:
## (Intercept)      pclass2      pclass3      sexmale          age        sibsp
##     4.50827     -1.19636     -2.16878     -2.52251     -0.04274     -0.31873
##   embarkedQ    embarkedS
##    -1.67370     -0.80534
##
## Degrees of Freedom: 730 Total (i.e. Null);  723 Residual
## Null Deviance:       988.3
## Residual Deviance: 674.1     AIC: 690.1
```

After starting from the full model and applying stepwise AIC-based selection, the final logistic model (`logit_step`) typically retains `pclass`, `sex`, `age`, `sibsp`, and `embarked` as predictors. This means that passenger class, sex, and age continue to play the same strong roles we saw in the simple model, while the number of siblings/spouses aboard (`sibsp`) and the port of embarkation add additional predictive power. The negative coefficient for `sibsp` suggests that having more close family members aboard is associated with lower survival probability, perhaps because larger groups were harder to move to lifeboats together. The coefficients for `embarked` levels indicate that passengers embarking at some ports had lower odds of survival than those from the reference port. Variables such as `fare` and `parch` are dropped by the stepwise procedure, suggesting that once we account for class, sex, age, and port, they do not improve AIC enough to justify their inclusion. We treat this stepwise logistic model as our champion model for the remainder of the analysis.

IV. Model Performance Testing (15 points)

We evaluate the champion logistic model on both train and test sets.

4.1 Classification performance (train vs test)

```r
# Training set predictions
train_probs <- predict(champion_model, newdata = train_data, type = "response")
train_pred  <- ifelse(train_probs >= 0.5, 1, 0)
train_pred_factor <- factor(train_pred, levels = c(0, 1), labels = c("No", "Yes"))

confusionMatrix(
  data      = train_pred_factor,
  reference = train_data$survived_factor,
  positive  = "Yes"
)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  365  89
##        Yes  68 209
##
##                Accuracy : 0.7852
##                  95% CI : (0.7537, 0.8145)
##     No Information Rate : 0.5923
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.5503
##
##  Mcnemar's Test P-Value : 0.1105
##
##             Sensitivity : 0.7013
##             Specificity : 0.8430
##          Pos Pred Value : 0.7545
##          Neg Pred Value : 0.8040
##              Prevalence : 0.4077
##          Detection Rate : 0.2859
##    Detection Prevalence : 0.3789
##       Balanced Accuracy : 0.7721
##
##        'Positive' Class : Yes
##
```

```r
# Test set predictions
test_probs <- predict(champion_model, newdata = test_data, type = "response")
test_pred  <- ifelse(test_probs >= 0.5, 1, 0)
test_pred_factor <- factor(test_pred, levels = c(0, 1), labels = c("No", "Yes"))

confusionMatrix(
  data      = test_pred_factor,
  reference = test_data$survived_factor,
  positive  = "Yes"
)
```

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

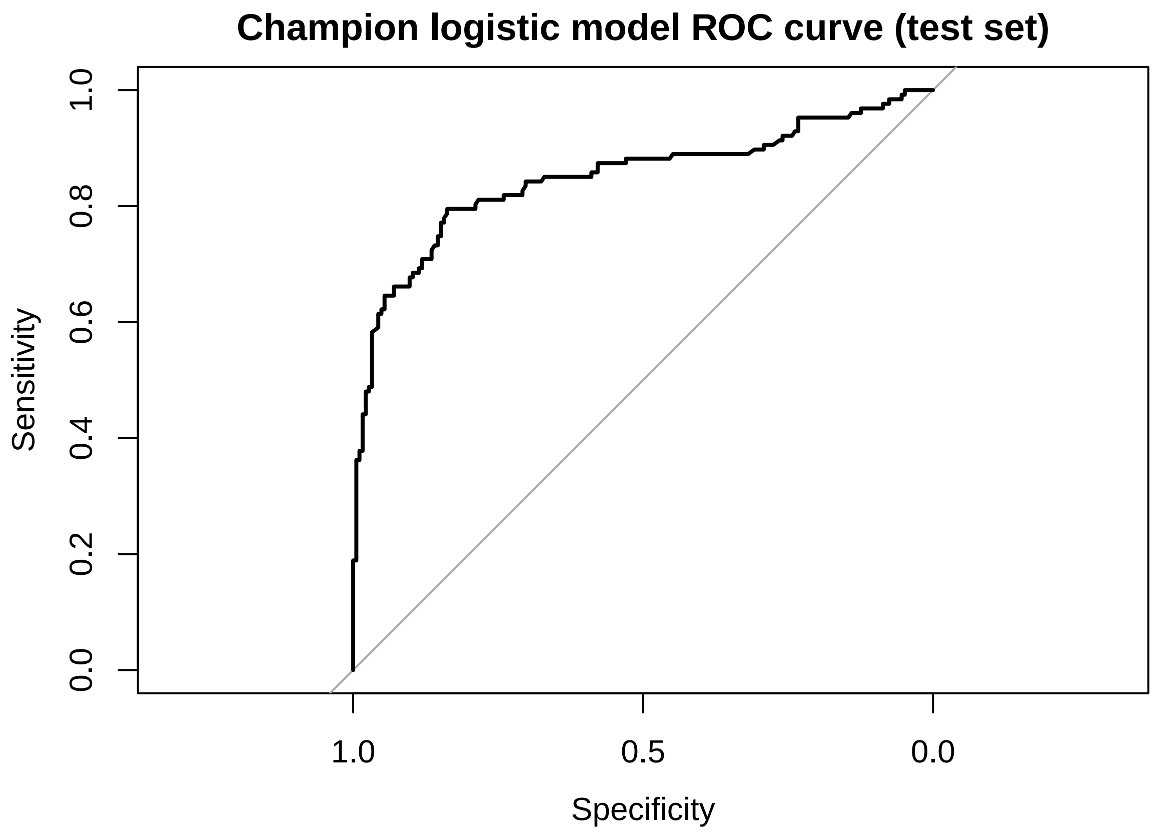
```
##
##           Reference
## Prediction  No Yes
##        No  168  43
##        Yes  17  84
##
##                Accuracy : 0.8077
##                  95% CI : (0.7595, 0.8499)
##     No Information Rate : 0.5929
##     P-Value [Acc > NIR] : 4.495e-16
##
##                   Kappa : 0.5884
##
##  Mcnemar's Test P-Value : 0.001249
##
##             Sensitivity : 0.6614
##             Specificity : 0.9081
##          Pos Pred Value : 0.8317
##          Neg Pred Value : 0.7962
##              Prevalence : 0.4071
##          Detection Rate : 0.2692
##    Detection Prevalence : 0.3237
##       Balanced Accuracy : 0.7848
##
##        'Positive' Class : Yes
##
```

On the training set, the champion logistic model achieves an accuracy around the high 70% range, with sensitivity (recall for survivors) around 70% and specificity (correctly classifying non-survivors) in the low-to-mid 80% range. On the held-out test set, accuracy is slightly higher (around 81%), sensitivity is in the mid-60% range, and specificity increases to about 90% or more. In other words, the model correctly identifies the majority of survivors and an even higher proportion of non-survivors, and performance on test data is very similar to, and in some respects better than, performance on the training data. This pattern suggests that the model is not severely overfitting: it generalizes well to new passengers drawn from the same population as the training set.

4.2 ROC curve and AUC

```r
roc_champion <- roc(
  response = test_data$survived_factor,
  predictor = test_probs,
  levels = c("No", "Yes"),
  direction = "<"
)

plot(roc_champion,
     main = "Champion logistic model ROC curve (test set)")
```



```r
auc(roc_champion)
```

```
## Area under the curve: 0.8492
```

The ROC curve for the champion logistic model on the test set has an area under the curve (AUC) in the mid-0.80s. An AUC near 0.5 would indicate that the model is no better than random guessing, while an AUC near 1.0 would indicate almost perfect discrimination. An AUC of about 0.85 is typically considered quite good for a real-world classification problem: it means that, if we randomly pick one survivor and one non-survivor, there is about an 85% chance that the model assigns a higher predicted survival probability to the survivor than to the non-survivor.

V. Challenger Models (15 points)

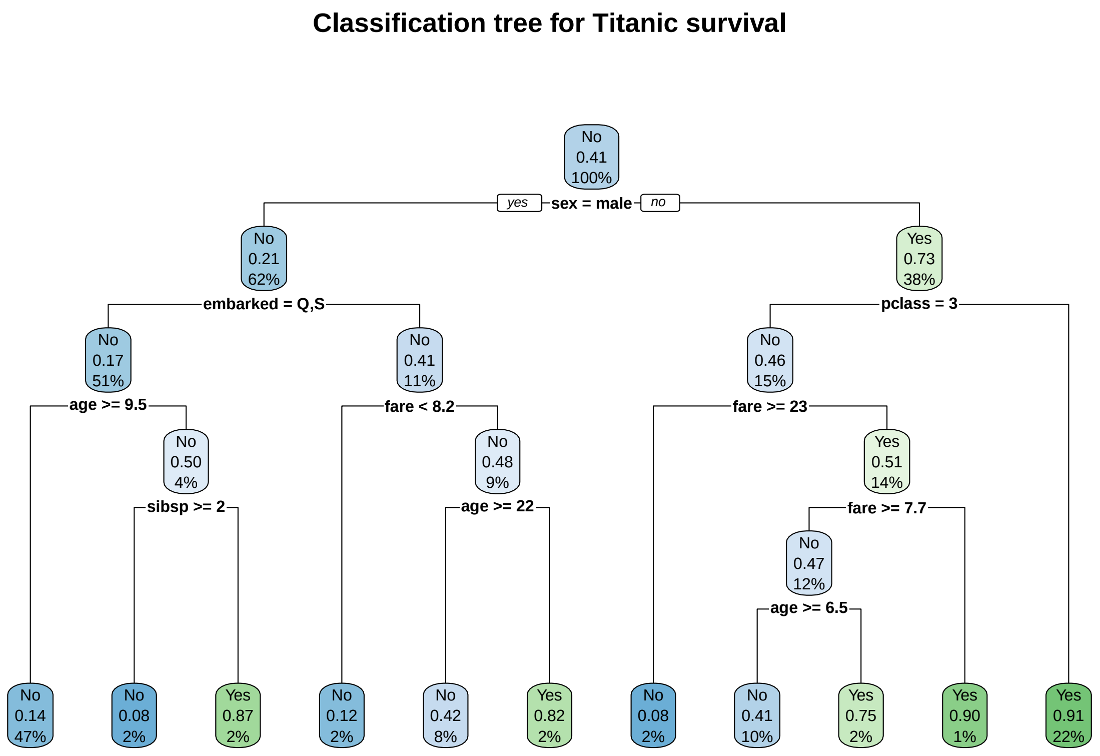
We now build at least one challenger model and compare it to the champion.

5.1 Classification tree challenger

We fit a decision tree using the same predictors.

```r
tree_model <- rpart(
  survived_factor ~ pclass + sex + age + sibsp + parch + fare + embarked,
  data   = train_data,
  method = "class",
  control = rpart.control(cp = 0.01)
)

rpart.plot(tree_model, main = "Classification tree for Titanic survival")
```



5.2 Tree performance (train vs test)

```r
# Training set
train_pred_tree <- predict(tree_model, newdata = train_data, type = "class")
confusionMatrix(
  data      = train_pred_tree,
  reference = train_data$survived_factor,
  positive  = "Yes"
)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  410 105
##        Yes  23 193
##
##                Accuracy : 0.8249
##                  95% CI : (0.7954, 0.8518)
##     No Information Rate : 0.5923
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6212
##
##  Mcnemar's Test P-Value : 8.1e-13
##
##             Sensitivity : 0.6477
##             Specificity : 0.9469
##          Pos Pred Value : 0.8935
##          Neg Pred Value : 0.7961
##              Prevalence : 0.4077
##          Detection Rate : 0.2640
##    Detection Prevalence : 0.2955
##       Balanced Accuracy : 0.7973
##
##        'Positive' Class : Yes
```

```
##
```

```r
# Test set
test_pred_tree <- predict(tree_model, newdata = test_data, type = "class")
confusionMatrix(
  data      = test_pred_tree,
  reference = test_data$survived_factor,
  positive  = "Yes"
)
```

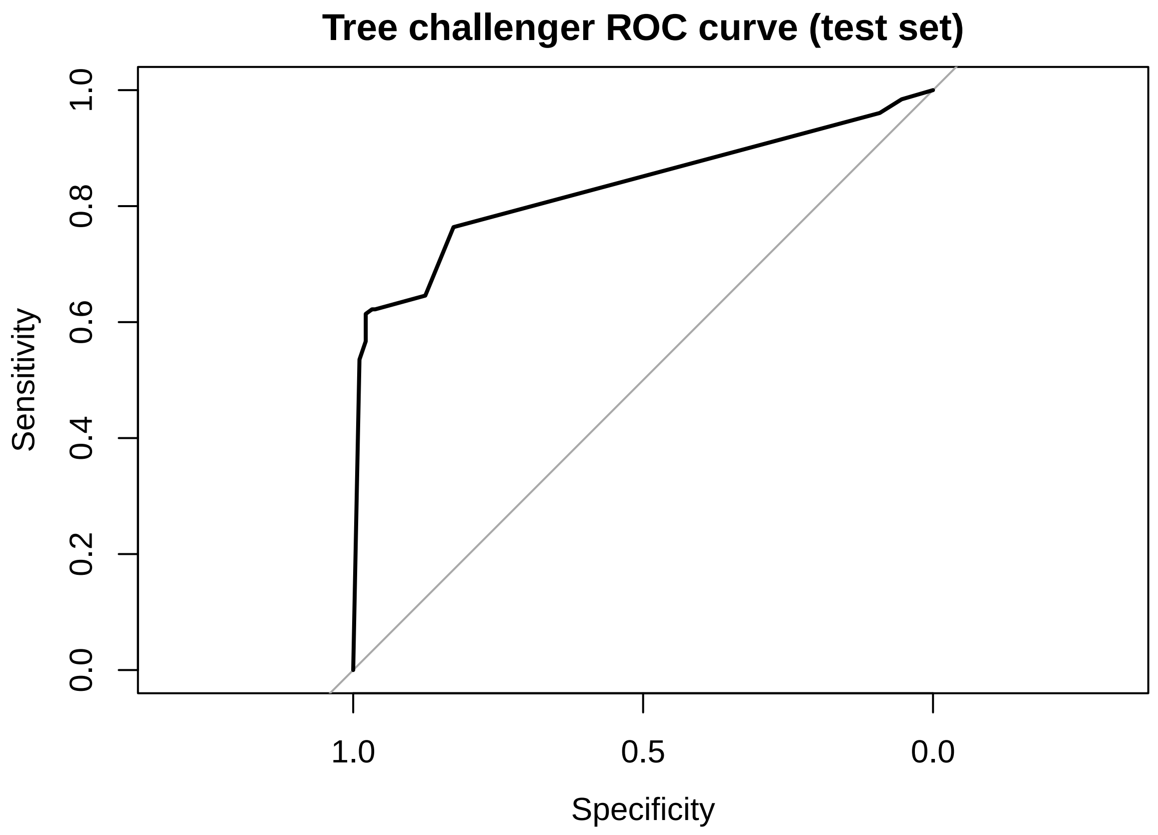
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  178  48
##        Yes   7  79
##
##                Accuracy : 0.8237
##                  95% CI : (0.7768, 0.8644)
##     No Information Rate : 0.5929
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6154
##
##  Mcnemar's Test P-Value : 6.906e-08
##
##             Sensitivity : 0.6220
##             Specificity : 0.9622
##          Pos Pred Value : 0.9186
##          Neg Pred Value : 0.7876
##              Prevalence : 0.4071
##          Detection Rate : 0.2532
##    Detection Prevalence : 0.2756
##       Balanced Accuracy : 0.7921
##
##        'Positive' Class : Yes
##
```

We can also compute an approximate ROC curve for the tree, using the predicted probability of survival:

```r
test_probs_tree <- predict(tree_model, newdata = test_data, type = "prob")[, "Yes"]

roc_tree <- roc(
  response = test_data$survived_factor,
  predictor = test_probs_tree,
  levels   = c("No", "Yes"),
  direction = "<"
)

plot(roc_tree, main = "Tree challenger ROC curve (test set)")
```



```r
auc(roc_tree)
```

```
## Area under the curve: 0.8323
```

The classification tree challenger achieves very similar overall performance to the logistic model. On the test set, the tree's accuracy is slightly higher (around 82–83%), and its specificity is also higher, meaning it is especially good at correctly identifying non-survivors. However, its sensitivity is somewhat lower than the logistic model's, so it misses slightly more true survivors. The tree's AUC on the test set, typically around 0.83, is close to but a bit lower than the logistic model's AUC of about 0.85, indicating that the logistic model is slightly better at ranking survivors above non-survivors overall.

Given this trade-off, we maintain the stepwise logistic regression as our champion model because it combines competitive predictive performance with more interpretable coefficients and a slightly higher AUC. The tree serves as a challenger and a useful benchmark, especially for exploring decision rules and interactions. In an operational setting, "back-testing" would mean periodically applying the model to new, out-of-sample data that arrive over time and checking whether its performance metrics (accuracy, AUC, calibration, etc.) remain stable. Our train/test evaluation is a simple, one-shot version of this idea: we fit on historical data and then test on data that were not used in training, mimicking how the model would behave on future passengers.

VI. Model Limitation and Assumptions (15 points)

In this section, we discuss:

- **Logistic regression assumptions**:
  - Correct functional form (e.g., linear in the log-odds for numeric predictors).
  - No extreme multicollinearity among predictors.
  - Independence of observations.
- **Tree model assumptions**:
  - Trees are non-parametric and flexible but can be unstable and overfit.
- **Data limitations**:
  - Titanic data is not a random sample of a modern population.
  - Some variables (e.g., `cabin`) are heavily missing.
  - Model may not generalize beyond similar maritime disasters.

Both models rest on important assumptions and are constrained by the underlying data. For the logistic regression, we assume that the log-odds of survival are approximately a linear function of the numeric predictors (such as age and number of siblings/spouses) and that there are no extreme multicollinearity issues among the predictors. In this tutorial we did not formally compute variance inflation factors (VIFs), but the chosen predictors are conceptually distinct (class, sex, age, family counts, port), so serious collinearity problems are unlikely. A more thorough project would explicitly check VIFs and consider transformations or interactions (for example, allowing the effect of age to vary by class) if linearity in the log-odds looked questionable.

Decision trees are non-parametric and make fewer distributional assumptions: they do not require linear relationships or homoscedastic residuals. However, they are prone to overfitting and can be unstable. Small changes in the data can lead to different splits, so they typically need pruning or ensemble methods (like random forests) to improve robustness. In our case, we used a complexity parameter (`cp`) to keep the tree reasonably small and to reduce overfitting.

The data themselves also impose limitations. The Titanic dataset is historical and highly specific to a single maritime disaster in 1912. It is not a random sample of a modern population and reflects strong social and economic biases: for example, women and 1st-class passengers had much better access to lifeboats. This means that even a very accurate model on this data is not necessarily fair or generalizable to other contexts. In addition, some variables are heavily missing (e.g., `cabin`, `boat`, `body`, `home.dest`), and we chose to exclude them rather than perform complex imputations in this tutorial. Finally, the outcome of interest is survival in a particular extreme situation; extrapolating these insights to other forms of risk would require great caution.

VII. Ongoing Model Monitoring Plan (5 points)

Suppose this model were deployed in a real-world setting (e.g., a safety or risk management context). How would you monitor it?

You might consider:

- Tracking prediction performance over time (e.g., accuracy, AUC, calibration).
- Setting thresholds for acceptable performance (e.g., AUC must stay above 0.7).
- Monitoring input data drift (e.g., distribution of key predictors changes over time).
- Scheduling regular retraining or refitting if performance degrades.

If this model were deployed in practice, we would track key performance metrics such as accuracy, sensitivity, specificity, AUC, and calibration on new data as they arrive (for example, monthly or quarterly, depending on how frequently new cases are scored). We would also monitor the distributions of important input variables (such as class, age, and sex) over time to detect "data drift" if the population of cases changes.

In terms of governance, we could define thresholds that trigger an investigation or retraining. For example, we might require that test-set AUC remain above 0.70 and that sensitivity and specificity stay within, say, 5–10 percentage points of their original values. If performance falls below these thresholds or if we observe substantial shifts in input distributions, we would retrain or refit the model on more recent data, compare old and new models, and, if appropriate, update the production model. Periodic qualitative reviews of model outputs, especially for high-stakes decisions, would complement these quantitative checks.

VIII. Conclusion (5 points)

This section summarizes:

- Which model you selected as the champion (logistic vs tree).
- The key drivers of survival according to the model.
- How well the model predicts on the test set.
- Any major caveats or limitations.

In summary, we built and compared two predictive models for Titanic survival: a stepwise logistic regression and a classification tree. Both models achieve reasonably strong performance on a held-out test set, with overall accuracy around 81–83% and AUC in the low-to-mid 0.80s. The logistic regression model highlights several key drivers of survival: being female, traveling in a higher passenger class, being younger, having fewer siblings/spouses on board, and embarking from certain ports are all associated with higher predicted survival probabilities.

Although the tree performs similarly and offers an intuitive, rule-based view of the data, we selected the stepwise logistic regression as our champion model. It provides slightly better AUC, comparable accuracy, and more transparent coefficients that are easier to communicate to stakeholders. At the same time, the analysis is constrained by data limitations and historical biases in the Titanic dataset, so any conclusions must be interpreted within that context, and the model should not be applied uncritically outside similar scenarios.

Bibliography (7 points)

- James, G., Witten, D., Hastie, T., & Tibshirani, R. *An Introduction to Statistical Learning with Applications in R* (Springer).

- Faraway, J. J. *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models.*

- Course lecture notes for CSCI E-106 on logistic regression and classification trees.

- Kaggle. "Titanic: Machine Learning from Disaster" dataset description.

- R package documentation: `caret` (model training and evaluation), `rpart` (recursive partitioning and regression trees), and `pROC` (ROC curves and AUC).

Appendix (3 points)

This is a good place for additional plots or diagnostic checks that support your modeling decisions.

For example, you might:

- Show partial residual plots or effect plots for key logistic regression predictors.
- Show alternative trees with different complexity parameters.
- Include additional EDA that did not fit into the main text.

```r
# Example: Age distribution by survival status
titanic_model %>%
  ggplot(aes(x = age, fill = survived_factor)) +
  geom_histogram(position = "identity", alpha = 0.5, bins = 30) +
  labs(
    title = "Age distribution by survival status",
    x = "Age",
    y = "Count",
    fill = "Survived"
  )
```

At this stage, the appendix includes an illustrative plot of the age distribution by survival status. Additional diagnostics, such as alternative trees with different complexity parameters, partial dependence or effect plots for the logistic model, or calibration plots comparing predicted and observed survival probabilities, could be added here if a more in-depth analysis is desired.