Lab 11 - Predicting ICU survival

TENSORFLOW 2.0

11/19/2020

Load packages and data

```
library(tidyverse)
library(tidymodels)
library(knitr)
```

```
icu <- read_csv("data/icu.csv")
```

Exercise 1

At first, lets convert the binary variables in our dataset into factors.

```
icu <- icu %>%
  mutate(
    survive = as_factor(survive),
    sex = as_factor(sex),
    infection = as_factor(infection),
    emergency = as_factor(emergency)
  )

glimpse(icu)
```

```
## Rows: 200
## Columns: 8
## $ id        <dbl> 881, 462, 517, 154, 401, 889, 100, 73, 502, 639, 208, 763...
## $ survive   <fct> 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, ...
## $ age       <dbl> 89, 69, 34, 53, 40, 62, 78, 66, 55, 46, 70, 55, 64, 32, 6...
## $ sex       <fct> 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...
## $ infection <fct> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, ...
## $ sysbp     <dbl> 190, 150, 110, 148, 140, 110, 180, 206, 190, 110, 168, 13...
## $ pulse     <dbl> 114, 85, 80, 128, 65, 78, 75, 90, 136, 128, 122, 140, 88,...
## $ emergency <fct> 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, ...
```

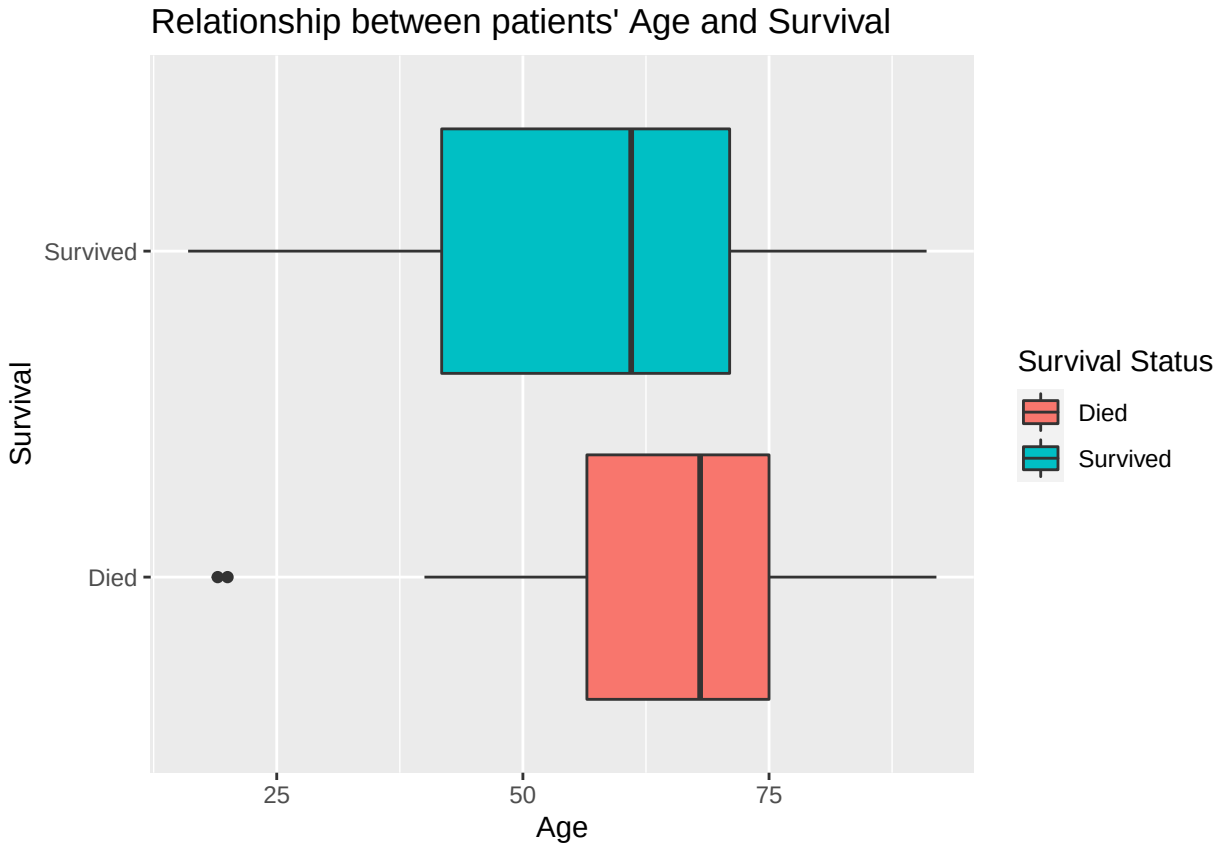
It is evident from the dataset above that we have successfully converted our targeted variables into factors.

Exercise 2

Now, lets visualize different relationships in our dataset.

At first, lets see the relationship between Age and Survival Status of the patient.

```
icu %>%
  ggplot(aes(x = age, y = survive, fill = survive)) +
  geom_boxplot() +
  scale_y_discrete(labels = c("0" = "Died",
                              "1" = "Survived")) +
  scale_fill_discrete(labels = c("0" = "Died",
                                 "1" = "Survived")) +
  labs(
    x = "Age",
    y = "Survival",
    title = "Relationship between patients' Age and Survival",
    fill = "Survival Status"
  )
```

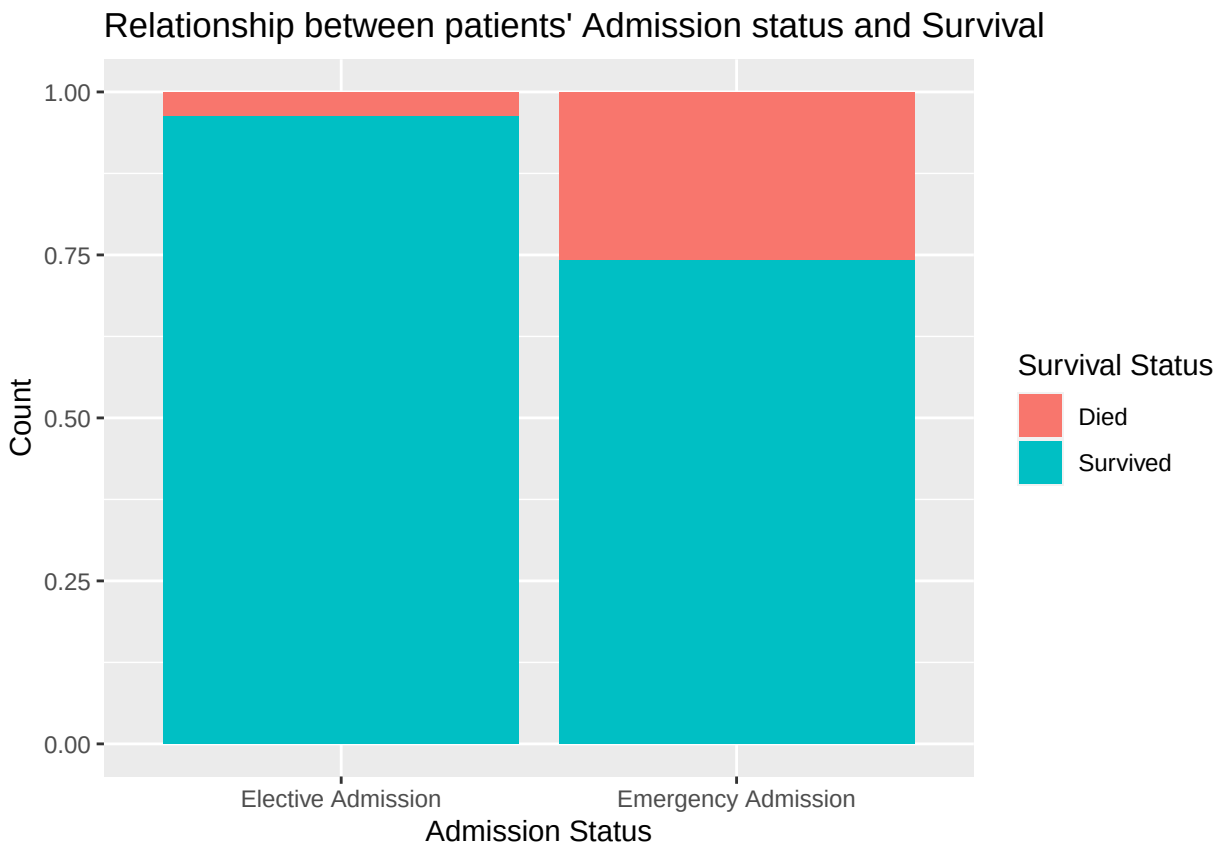


It is evident from the visualization above that the median age for patients who died was higher than the median age for patients who survived.

Now, lets see the relationship between Admission Status and the Survival Status of the patient.

```
icu %>%
  ggplot(aes(x = emergency, fill = survive)) +
  geom_bar(position = "fill") +
  scale_x_discrete(labels = c("0" = "Elective Admission",
                              "1" = "Emergency Admission")) +
  scale_fill_discrete(labels = c("0" = "Died",
                                 "1" = "Survived")) +
  labs(
    x = "Admission Status",
```

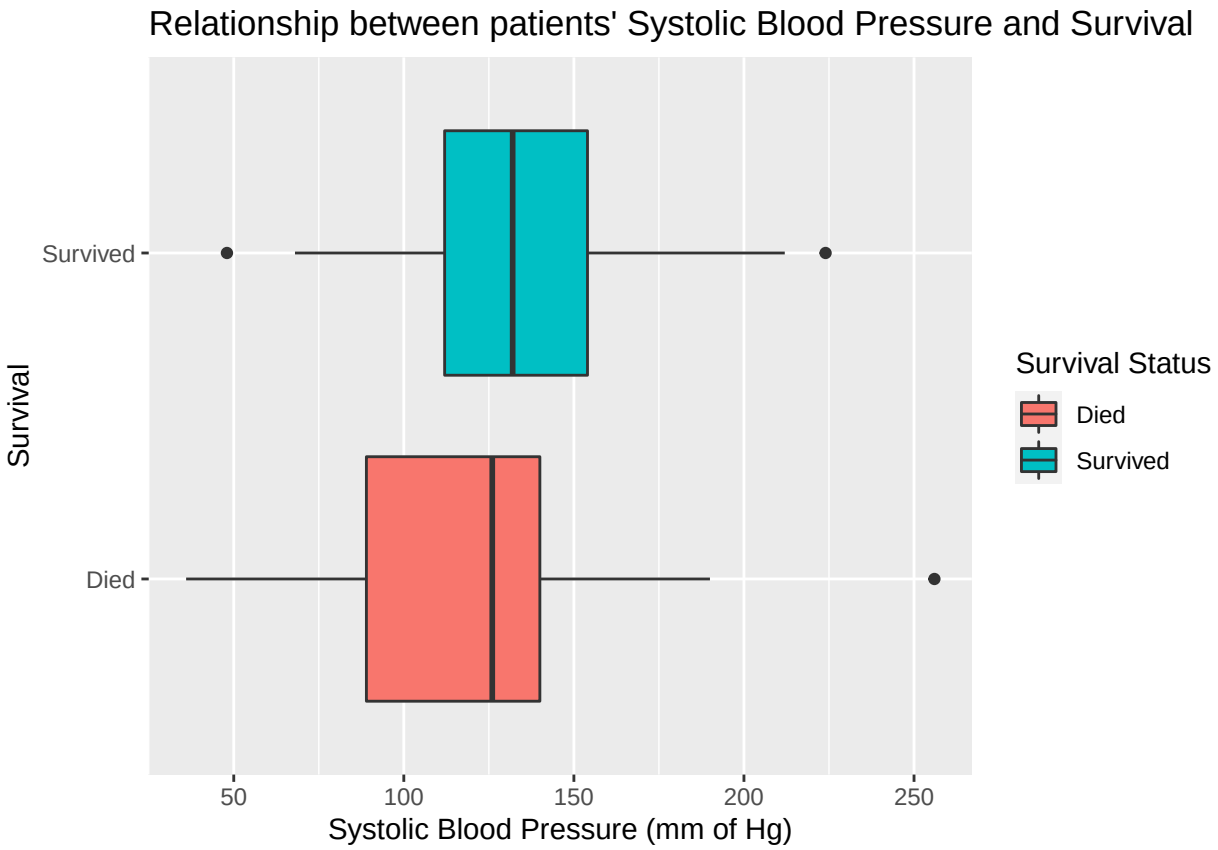
```
    y = "Count",
    title = "Relationship between patients' Admission status and Survival",
    fill = "Survival Status"
  )
```



It is evident from the graph above that proportion of patients who died was higher in the emergency admission compared to that in the elective admission.

Finally, lets see the relationship between Systolic Blood Pressure and Survival Status of the patient.

```
icu %>%
  ggplot(aes(x = sysbp, y = survive, fill = survive)) +
  geom_boxplot() +
  scale_y_discrete(labels = c("0" = "Died",
                              "1" = "Survived")) +
  scale_fill_discrete(labels = c("0" = "Died",
                                 "1" = "Survived")) +
  labs(
    x = "Systolic Blood Pressure (mm of Hg)",
    y = "Survival",
    title = "Relationship between patients' Systolic Blood Pressure and Survival",
    fill = "Survival Status"
  )
```



It is evident from the visualization above that the median systolic blood pressure for the patients who died was lower than that for the patients who survived.

Exercise 3

First, lets split our dataset into training and testing set.

```
# Fix random numbers by setting the seed
# Enables analysis to be reproducible when random numbers are used
set.seed(1119)
# Put 80% of the data into the training set
icu_split <- initial_split(icu, prop = 0.80)
# Create data frames for the two sets:
train_data <- training(icu_split)
test_data  <- training(icu_split)
```

Exercise 4

Now, lets fit a logistic regression model on our training set.

```
icu_fit <- logistic_reg() %>%
  set_engine("glm") %>%
  fit(survive ~ age + sysbp + emergency, data = train_data, family = "binomial")

tidy(icu_fit)
```

```
## # A tibble: 4 x 5
```

```
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)   3.35     1.45         2.32 0.0205
## 2 age          -0.0379   0.0127      -2.99 0.00279
## 3 sysbp         0.0154   0.00692      2.23 0.0258
## 4 emergency1   -2.03     0.774       -2.63 0.00862
```

Exercise 5

The equation of the fitted logistic regression model can be written as:
$\log \frac{p}{1-p} = 3.35 - 0.0379 * age + 0.0154 * sysbp - 2.03 * emergency_{emergency\ admission}$

Exercise 6

The patient's predicted log-odds of survival is -0.5155.

Exercise 7

The patient's probability of survival is 0.37.

Exercise 8

This is not a surprising outcome as our model might not necessarily be 100% accurate. Our model might have false positives and false negatives and this particular case might be one of those incidents where our model failed to predict correctly. To be more specific, taking ".pred_0/dying" as the event, this is an instance of a False Positive(ie. a patient was predicted to die but he/she survived).

Exercise 9

Now, lets use our model to make predictions on our testing data.

```
icu_pred <- predict(icu_fit, test_data, type = "prob") %>%
  bind_cols(test_data %>% select(survive, id))

icu_pred
```

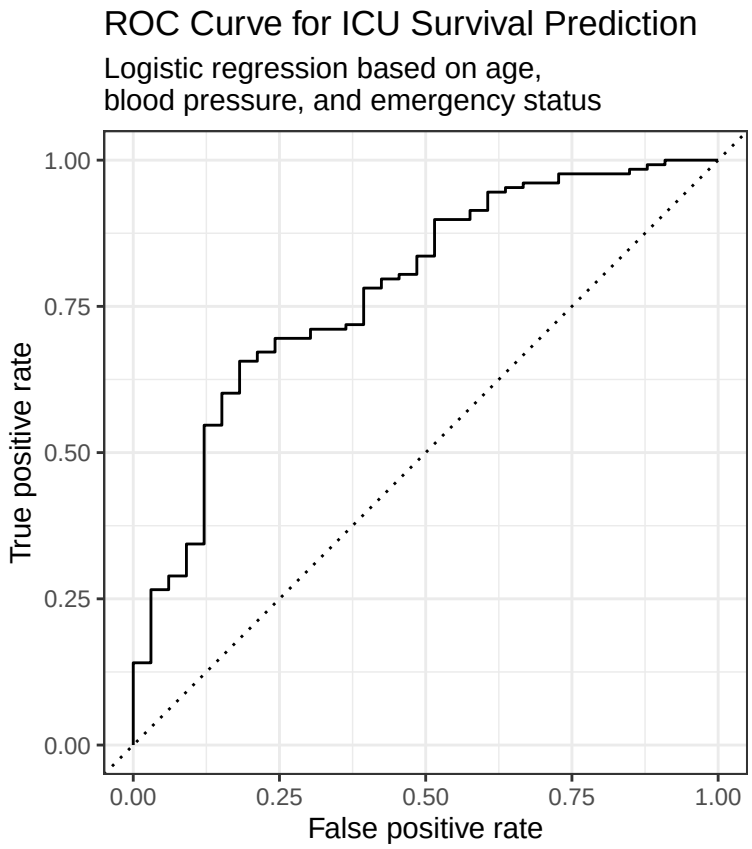
```
## # A tibble: 161 x 4
##    .pred_0 .pred_1 survive    id
##      <dbl>   <dbl> <fct>   <dbl>
##  1  0.292    0.708 1         881
##  2  0.264    0.736 1         462
##  3  0.123    0.877 1         401
##  4  0.338    0.662 1         889
##  5  0.0400   0.960 1         100
##  6  0.119    0.881 1          73
##  7  0.102    0.898 1         502
##  8  0.0353   0.965 1         639
##  9  0.0357   0.964 0         208
## 10  0.0323   0.968 1         763
## # ... with 151 more rows
```

The first row of the tibble above explains that for the person with ID 881 for whom the truth was that he/she survived, our model predicted that he/she had a probability of 0.708 of surviving and a probability of 0.292 of dying.

Exercise 10

Now, lets produce the ROC curve for our model.

```
icu_pred %>%
  roc_curve(
    truth = survive,
    .pred_1,
    event_level = "second"
  ) %>%
  autoplot() +
  labs(x = "False positive rate",
       y = "True positive rate",
       title = "ROC Curve for ICU Survival Prediction",
       subtitle = "Logistic regression based on age,\nblood pressure, and emergency status")
```



If we decide to set our threshold at a level that will achieve a true positive rate of 80%, our false positive rate will be 44%.

The True Positive Rate(TPR) of 80% means that the probability of our model predicting the patient to survive accurately is 80% and a False Positive Rate(FPR) of 44% means the probability of our model predicting the patient to survive even though the patient dies is 44%.

Exercise 11

Now, lets produce the confusion matrix using a threshold score of 0.5.

```
# Threshold of 0.5
cutoff_prob <- 0.5
icu_pred %>%
  mutate(
    survive      = if_else(survive == 1, "Patient survived", "Patient died"),
    survive_pred = if_else(.pred_1 > cutoff_prob, "Patient predicted to survive",
                           "Patient predicted to die")
    ) %>%
  count(survive_pred, survive) %>%
  pivot_wider(names_from = survive, values_from = n) %>%
  kable(col.names = c("", "Patient died", "Patient survived"))
```

	Patient died	Patient survived
Patient predicted to die	9	3
Patient predicted to survive	24	125

Similarly, lets produce the confusion matrix using a threshold score of 0.3.

```
# Threshold of 0.3
cutoff_prob <- 0.3
icu_pred %>%
  mutate(
    survive      = if_else(survive == 1, "Patient survived", "Patient died"),
    survive_pred = if_else(.pred_1 > cutoff_prob, "Patient predicted to survive",
                           "Patient predicted to die")
    ) %>%
  count(survive_pred, survive) %>%
  pivot_wider(names_from = survive, values_from = n) %>%
  kable(col.names = c("", "Patient died", "Patient survived"))
```

	Patient died	Patient survived
Patient predicted to survive	33	128

Finally, lets produce the confusion matrix using a threshold score of 0.8.

```
# Threshold of 0.8
cutoff_prob <- 0.8
icu_pred %>%
  mutate(
    survive      = if_else(survive == 1, "Patient survived", "Patient died"),
    survive_pred = if_else(.pred_1 > cutoff_prob, "Patient predicted to survive",
                           "Patient predicted to die")
    ) %>%
  count(survive_pred, survive) %>%
  pivot_wider(names_from = survive, values_from = n) %>%
  kable(col.names = c("", "Patient died", "Patient survived"))
```

	Patient died	Patient survived
Patient predicted to die	25	40
Patient predicted to survive	8	88

Comparing all the confusion matrix for different threshold scores above, it becomes evident that a threshold score of 0.3 seems to be performing the worst as it doesnt predict any patients to die at all.As for the other two scores, in the threshold score of 0.5, more number of patients died who were predicted to survive compared to that for 0.8. This means that if we use our model with threshold score of 0.5, then we have higher chance of removing a patient from the ICU and letting him/her to die without proper medical resources. We cannot fail when it comes to someone's life. While in the case of threshold score of 0.8, at its worst, we might have patients who can survive still in the ICU who will still be using critical resources. We wont lose lives just because our model fails in this case. Hence, I would recommend the doctors to use the threshold score of 0.8.