```
library(tidymodels)
library(lubridate)
library(patchwork)
hotels <- readr::read_csv('https://raw.githubusercontent.com/
rfordatascience/tidytuesday/master/data/2020/2020-02-11/hotels.csv')
theme_set(theme_light())

hotels <- hotels %>%
  mutate(
    check_in_date = ymd(paste(arrival_date_year, arrival_date_month,
                              arrival_date_day_of_month)),
    reservation_date = check_in_date,
    booking_date = check_in_date - days(lead_time),
    stay_length = stays_in_week_nights + stays_in_weekend_nights,
    check_out_date = if_else(reservation_status != "No-Show",
                             reservation_date + days(stay_length),
ymd("NA"))) %>%
  select(booking_date, is_canceled, hotel,customer_type, adults, children,
babies, lead_time, stay_length, deposit_type, check_in_date,
check_out_date) %>%
  mutate(not_canceled = (is_canceled - 1) * -1)

city_hotel <- hotels %>%
  filter(hotel == "City Hotel")

training_dates <- city_hotel %>%
  count(check_in_date) %>%
  arrange(check_in_date) %>%
  slice(round(0.8*n()):n()) %>%  # 80% trian set
  pull(check_in_date)

train <- city_hotel %>%
  filter(!check_in_date %in% training_dates)

test <- city_hotel %>%
  filter(check_in_date %in% training_dates)

logistic_fit <- glm(not_canceled ~ customer_type +
                      adults +
                      children +
                      babies +
                      lead_time +
                      stay_length +
                      deposit_type, data = train,
                    family = binomial(link = logit))

capacity_frac <- function (p, rooms=100, loss=4, min_p=0.01) {
  E <- 0; E_m1 <- 0; Fx <- 0;
  P_overbook <- 0; bookings <- rooms;

  if(p > min_p) {
    while(E_m1 >= E) {
      E <-  bookings * Fx - loss * (bookings - rooms) * bookings *
P_overbook
```

```
      Fx <-  pbinom(rooms, bookings, p, lower.tail = TRUE)
      bookings <-  bookings + 1
      P_overbook <-  1 - Fx
      E_m1 <-  bookings * Fx - loss * (bookings - rooms) * bookings *
P_overbook
    }
  } else {
    bookings = rooms * 2.4
  }
  return(bookings)
}

test_pred <- test %>%
  mutate(p_hat = predict(logistic_fit,
                         newdata = test, type = "response"),
         cap = map_dbl(p_hat, capacity_frac),
         cap_fra = 1/cap)

sample_fn_overbook <- function(data_input, p_threash = 0.99) {

  success <- FALSE; i <- 1; samp <- data_input[1,]
  while(!success) {
    samp[i,] <- slice_sample(data_input, n = 1)
    success <- sum(samp$cap_fra) > p_threash
    i <- i + 1
  }
  samp <- samp[1:i,]

  return(samp)
}

sample_capacity <- function(data_input, capacity = 100) {
  output <- slice_sample(data_input, n = capacity, replace = TRUE)
  return(output)
}

set.seed(1999)

sim_results <- test_pred %>%
  group_by(check_in_date) %>%
  nest() %>%
  mutate(
    resampled_overbookings = map(data, sample_fn_overbook),
    resampled_capacity = map(data, sample_capacity),

    over_nbookings = map_dbl(resampled_overbookings, ~nrow(.x)),
    over_canceled = map_dbl(resampled_overbookings, ~sum(.x$is_canceled,
na.rm = TRUE)),
    over_booking_arrivals = over_nbookings - over_canceled,

    booking_n = map_dbl(resampled_capacity, ~nrow(.x)),
    booking_canceled = map_dbl(resampled_capacity, ~sum(.x$is_canceled,
na.rm = TRUE)),
    booking_arrivals = booking_n - booking_canceled
  )

sim_results %>%
  ungroup() %>%
  select(check_in_date, over_nbookings:ncol(.)) %>%
  mutate(penalty = if_else(over_booking_arrivals > 100,
(over_booking_arrivals - 100) * 4, 0),
         over_empty_rooms = if_else(over_booking_arrivals < 100, 100 -
over_booking_arrivals, 0),
         booking_empty_rooms = if_else(booking_arrivals < 100, 100 -
booking_arrivals, 0)) %>%
  summarise(overbooking_arrivals = sum(over_booking_arrivals),
            booking_arrivals = sum(booking_arrivals),
            overbooking_penalty = sum(penalty),
            overbooking_empty_rooms = sum(over_empty_rooms),
            booking_empty_rooms = sum(booking_empty_rooms),
  )
```