Predicting Hotel Cancellations

Art Tay

Abstract

Hotels are extremely interested in predicting room cancellations, as utilized capacity is a key driver of revenue. Data originally complied by *Antonio et. Al, 2019*, on over 100,000 hotel bookings was appropriated to build several logistic regression models. The final model had a predictive accuracy on a held-out test set of 81.5%. In conjunction with exploratory data analysis inferences beyond prediction were drawn. First, although the data was complied from two different types of hotels, and single predictive model was more powerful than a split model. This implies that the same factors that predict cancellations at Resort hotels can be used for City hotels and possibly other types of hotels. Second, all the feature present in the dataset were found to be useful predictors. On a forward basis, hotels should consider collecting and maintaining a similar database of customer data. Further investigation is required to identify exigent circumstances effecting Portugal bookings during the fall of 2015 as well as the rational behind canceled bookings where a non-refundable deposit was made.

Introduction

One of the primary concerns of a hotel is guests that end up canceling reservations. Cancelled reservations can dramatically impact revenue by reducing the overall utilization rate of rooms. *Antonio et Al., 2019*, collected bookings data from a resort hotel and a city hotel over a two year period. Over 100,000 bookings are represented in the dataset. The dataset contains categorical information about each booking such as the agent it was booked through, the date of arrival, and whether or not it was ultimately cancelled. The dataset also contained numeric information such as the number of previous cancellation a guest has made, and the number of guests on the bookings. Given the growing need to understand hotel cancellations, the goal of this analysis was to build a model that could predicted which bookings are most likely to be cancelled.

EDA

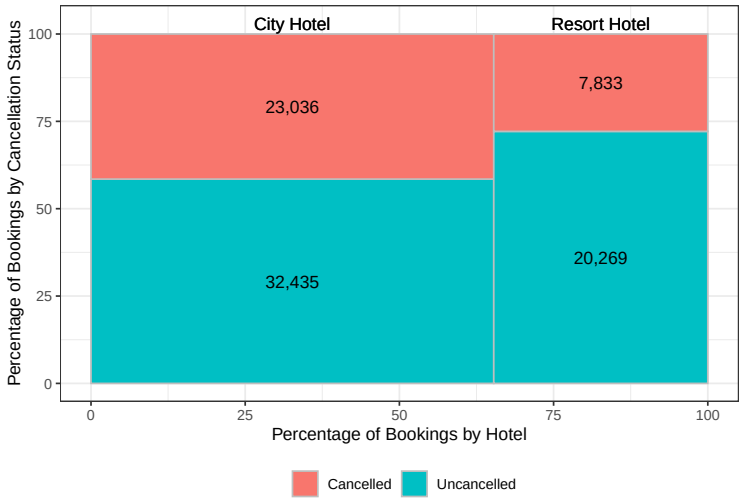


Figure 1: Number of bookings grouped by hotel type and cancellation status. The height on each box represents the percentage of bookings by cancellation status, while the width represents the percentage of bookings made in each type of hotel.

70% of the data was extract to conduct the exploratory data analysis as to not bias the final error rates. Based on this subset, the number of cancelled reservations is quite balanced with the number of uncancelled reservation. The figure above indicates that the data is more imbalanced for Resort Hotels, but the ratio in neither class exceeds 4:1.

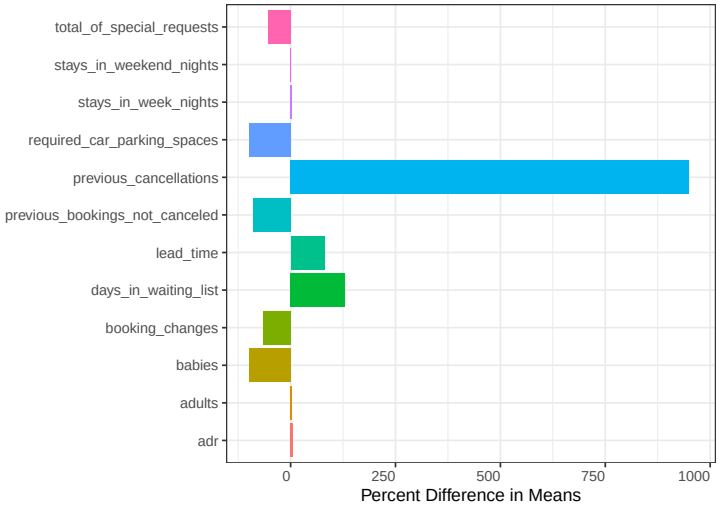


Figure 2: Percentage differences in means of numeric predictors for Cancelled v. Uncancelled reservations. Percentages were calculated relative to Cancelled reservations. Positive value indicate that the mean value of the Cancelled group is greater than the mean of the Uncancelled group.

The plot above clearly demonstrates the that guests who have previously canceled a bookings are far more likely to cancel another. Guest that cancelled reservations had nearly 1,000% more previous cancellation than guests that checked-in. Less significant indicators of cancellation included a longer lead time and a greater number of days on the waiting list.

On the contrary, guests that make special requests or require additional parking spaces are less likely to cancel. This may be caused by prepaid fee associated with these types of requests. Interestingly the duration and type of day that the booking covered did not differ between cancelled and uncanceled reservations. A long stay over a weekend seem just as likely to be cancelled as a short business trip during the week.

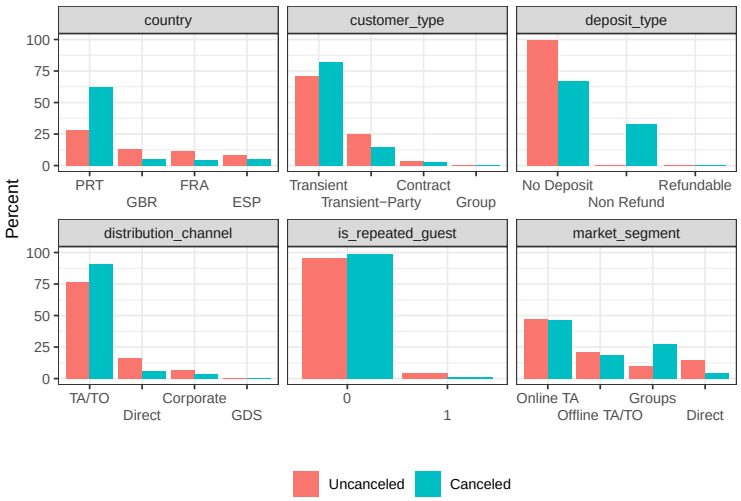


Figure 3: Percentage of different categories of bookings colored by cancellation status.

Although many different types of bookings are represented in the data, there is little separation in the proportion that get cancelled. Bookings in made Portugal are cancelled at a higher rate than any other country. The majority of these cancellation happened during the fall of 2015, which may indicate an event effected the country resulting cancellations; however, there were other time period with high rates of cancellations from Portugal. Strangely, the data seem to indicate that reservations that were made with a non-refundable deposit are more likely to be canceled. Smaller effects that can be seen on the above plot are: Groups are more likely to cancel, while, direct bookings are less likely to be cancelled.

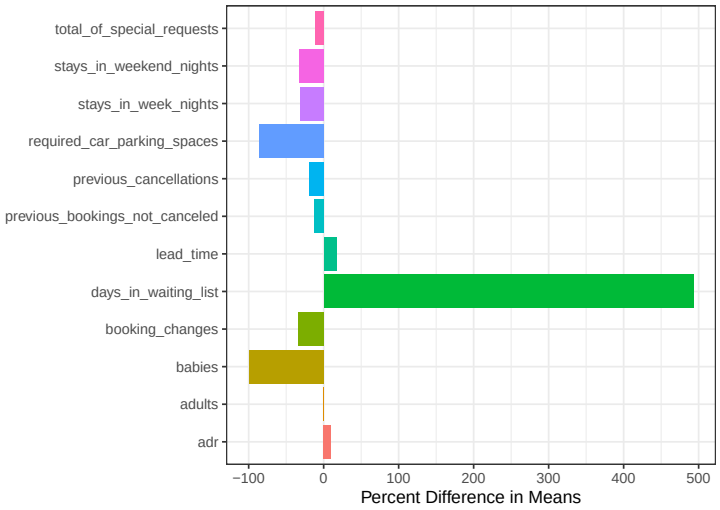


Figure 4: Percentage differences in means of numeric predictors for City v. Resort hotels. Percentages were calculated relative to the City group. Positive values indicate that the mean value of the City group is greater than the mean of the Resort group.

The plot above indicates that there is a much higher demand for City Hotels than there is for Resort Hotels as client spend almost 500% more time on the waiting list for City Hotels. This is probably a result of seasonality. City Hotels are used year around for business trips, whereas Resort Hotels are booking more infrequently. On the other hand, resort bookings tend to require more special accommodations.

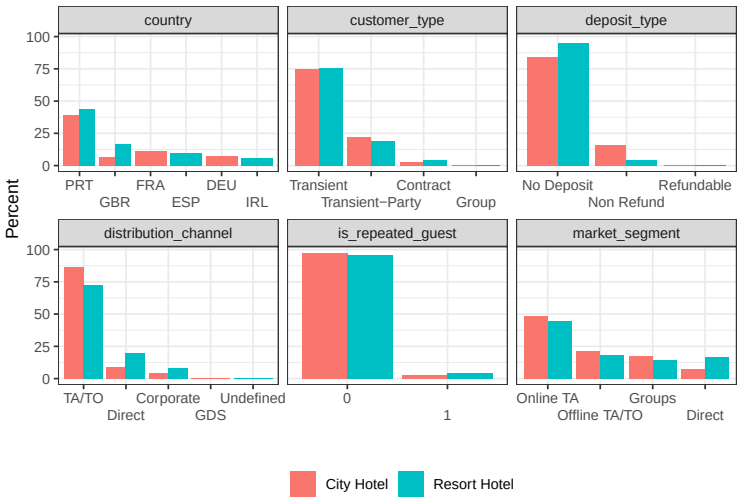


Figure 5: Percentage of different categories of bookings colored by type of hotel

Similar to the plot coded by cancellation status, Figure 5 indicates that although a variety of bookings types are represented in the data, there is very little separation on the basis of hotel.

Modeling Methodology

The data was initially split into a training data set and a testing data set (70% and 30% of the raw data respectively). Although some factors had `NULL` entries, it was generally meaningful as opposed to missing. For example, the majority of the agent data was `NULL`, but this was taken to mean that the customer simply booked a hotel without using an agent.

The `reservation_status` as well as `reservation_status_date` were both dropped from the feature space. The `reservation_status` variable was perfectly correlated with the outcome variable, the cancellation status, and therefore unlikely to be known prior to the actual cancellation. Although `reservation_status_date` is not perfectly correlated to the outcome, it is impossible to separate the intermediate status updates from the final outcome. The inclusion of this predictor in the model would most likely result in a highly optimistic error rate.

Additional, all factors were converted into $n - 1$ dummy variables. Then, near-zero-variance predictor as well as highly correlated variables were dropped. Finally, all originally numeric variables were centered and scaled to promote computational stability.

This data was then used to fit a standard logistic regression model, a penalized logistic regression model, and finally a split model based on hotel. The final model was selected on the basis of test set accuracy. Accuracy was calculated using the probability cut-point that maximized the kappa statistic on the training set for each given model. No models had Cook's Distance value greater than 1, therefore no observations were removed from the training sets of any of the models. Although regularization was applied, the optimal penalty was not significantly different from 0, and thus was unreported.

Results

Table 1: Summary of model metrics

	Combined Model				Split Model			
	City		Resort		City		Resort	
	Train	Test	Train	Test	Train	Test	Train	Test
Accuracy	0.8067	0.8038	0.8392	0.8369	0.7993	0.7981	0.8330	0.8302
Kappa	0.6062	0.6018	0.6036	0.5931	0.5896	0.5883	0.5941	0.5826
Sensitivity	0.8078	0.8034	0.8827	0.8848	0.8113	0.8113	0.8696	0.8712
Specificity	0.8051	0.8043	0.7265	0.7108	0.7824	0.7801	0.7384	0.7223
PPV	0.8537	0.8489	0.8931	0.8895	0.8400	0.8348	0.8959	0.8918
NPV	0.7484	0.7494	0.7053	0.7012	0.7465	0.7512	0.6863	0.6810

The overall accuracy of the standard logistic regression model (combined model) was 0.815. Model metrics were revaluate by hotel to make a direct comparison to the split model. Although, the metrics are close, the combined model dominates the split model in accuracy across both groups. This is mostly likely due to the higher data utilization of the combined model. This might also be the result of the two hotels having a very similar distributions of features.

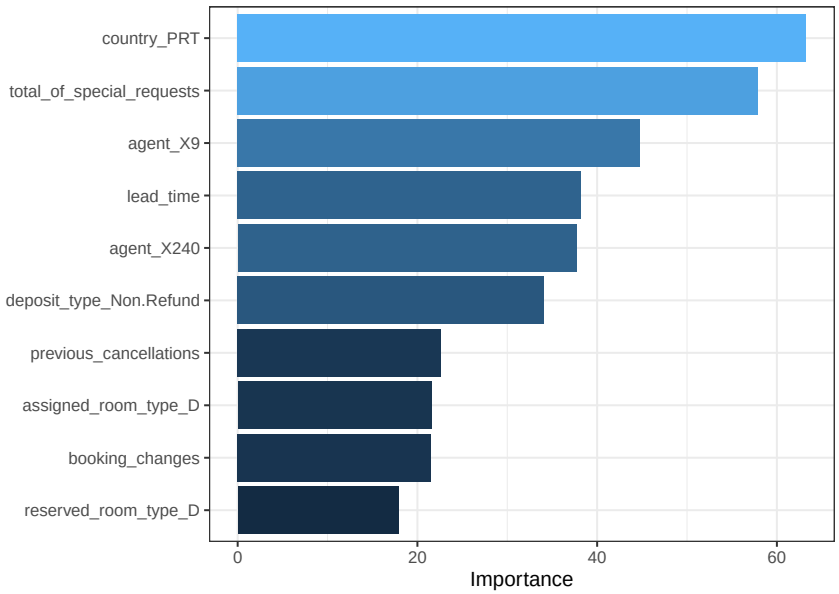


Figure 6: Ten Most Important Variable for the Final Model

Based on the variable importance plot, the most important categorical variable was country, and the most important numeric variable was the number of special requests. Based on the sign of the associated coefficients bookings from Portugal were more likely to be cancel, while the more special requests a customer made the less likely they were to cancel (for a full list of model coefficients refer the the *Appendix*). Interesting various agent variables were flag as significant to the model despite the high dispersion of agents with most having a low level of separation between the canceled and uncanceled bookings.

Conclusion

The overall goal of this analysis was to produce a predictive model that could estimate the probability of a client cancelling a bookings with a high degree of accuracy. The ultimate model chosen was a standard logistic regression model. The final model had a test accuracy of 81.5%. During the exploratory data analysis the implication of several variables on the ultimate outcome were hypothesized. Many on these hypothesizes were consistent with the model coefficients. For example the model predicts a greater cancellation rate for customers with a high number of previous cancellations. There are two issues that may require further investigation. Firstly, there were a high number of cancellations from Portugal made during the fall of 2015. It is possible that this trend will continue into the future, but exigent circumstances should be identified to increase the applicability of the model on future data. Second, the data seems to indicate that bookings made with a non-refundable deposit are more likely to be canceled. This seems logically backwards, as it is usually the case that customers are less likely to forego a partially prepaid room.

Appendix - Additional Plots

Table 2: Final Model Coefficients

term	estimate	std.error	statistic	p.value
Intercept	-1.764	2.947	-0.598	0.550
lead_time	0.561	0.015	38.194	0.000
stays_in_weekend_nights	0.125	0.012	10.804	0.000
stays_in_week_nights	0.181	0.012	15.168	0.000
adults	0.095	0.013	7.531	0.000
previous_cancellations	0.954	0.042	22.653	0.000
booking_changes	-0.270	0.013	-21.545	0.000
adr	0.195	0.015	13.170	0.000
required_car_parking_spaces	-5.568	11.613	-0.479	0.632
total_of_special_requests	-0.681	0.012	-57.873	0.000
hotel_Resort.Hotel	-0.144	0.016	-8.981	0.000
arrival_date_year_X2016	0.191	0.018	10.832	0.000
arrival_date_year_X2017	0.326	0.020	16.127	0.000
arrival_date_month_X2	0.053	0.015	3.468	0.001
arrival_date_month_X3	-0.007	0.016	-0.406	0.685
arrival_date_month_X4	0.067	0.017	3.947	0.000
arrival_date_month_X5	0.029	0.018	1.620	0.105
arrival_date_month_X6	-0.020	0.017	-1.169	0.242
arrival_date_month_X7	-0.029	0.019	-1.557	0.120
arrival_date_month_X8	0.019	0.020	0.979	0.327
arrival_date_month_X9	0.049	0.019	2.577	0.010
arrival_date_month_X10	0.132	0.019	7.123	0.000
arrival_date_month_X11	0.109	0.016	7.023	0.000
arrival_date_month_X12	0.102	0.016	6.580	0.000
meal_HB	-0.031	0.011	-2.731	0.006
meal_SC	0.031	0.010	3.230	0.001
country_DEU	-0.194	0.011	-17.219	0.000
country_ESP	0.084	0.011	7.760	0.000
country_FRA	-0.136	0.011	-12.318	0.000
country_GBR	-0.099	0.011	-8.722	0.000
country_PRT	0.922	0.015	63.255	0.000
market_segment_Direct	0.058	0.035	1.636	0.102
market_segment_Groups	0.205	0.037	5.529	0.000
market_segment_Offline.TA.TO	-0.023	0.041	-0.569	0.570
market_segment_Online.TA	0.067	0.052	1.281	0.200
distribution_channel_Direct	-0.204	0.034	-5.976	0.000
distribution_channel_TA.TO	-0.223	0.034	-6.647	0.000
reserved_room_type_D	0.339	0.019	17.916	0.000
reserved_room_type_E	0.236	0.020	11.854	0.000
assigned_room_type_D	-0.427	0.020	-21.572	0.000
assigned_room_type_E	-0.259	0.021	-12.518	0.000
deposit_type_Non.Refund	1.526	0.045	34.056	0.000
agent_X240	0.610	0.016	37.693	0.000
agent_X9	0.838	0.019	44.747	0.000
agent_NULL.	-0.087	0.018	-4.752	0.000
company_NULL.	0.208	0.022	9.528	0.000
customer_type_Transient	0.377	0.028	13.317	0.000

customer_type_Transient.Party	0.078	0.028	2.729	0.006

Appendix - Code

Project Description

- Perform an EDA – Brief EDA on that hotel booking data (city room vs. resort)
- Build a logistic regression model if one would cancel a reservation or not
- Check class imbalance
- Provide an evaluation Metric
- Show the feature needed any hot encoding

EDA

- Slight class imbalance. Roughly 2:1 Uncancelled:Cancelled. Unlike to cause modeling problems.
- Some variable difference identified between outcome classes.
- Slight differences between variables between hotel groups. Most likely insufficient to justify separate models.

```
data_full <- read_csv(file = "hotel_bookings.csv")
```

```
str(data_full)
glimpse(data_full)
```

Data Cleaning

```
# Train-Test Split (70/30)
set.seed(123)
data_split <- initial_split(data_full, prop = 0.7)

train <- training(data_split)
test <- testing(data_split)
```

```
# start a recipe
cleaning_recipe <- train %>%
                   recipe(is_canceled ~ .)
```

```
# Change all character variables to factors.
cleaning_recipe %<>%
  step_mutate(across(where(is.character), as.factor))

# Change outcome variable to be a factor.
cleaning_recipe %<>%
    step_mutate(is_canceled = as.factor(is_canceled), skip = T)

# Change month string to corresponding month number
cleaning_recipe %<>%
    step_mutate(arrival_date_month = match(arrival_date_month, month.name))
```

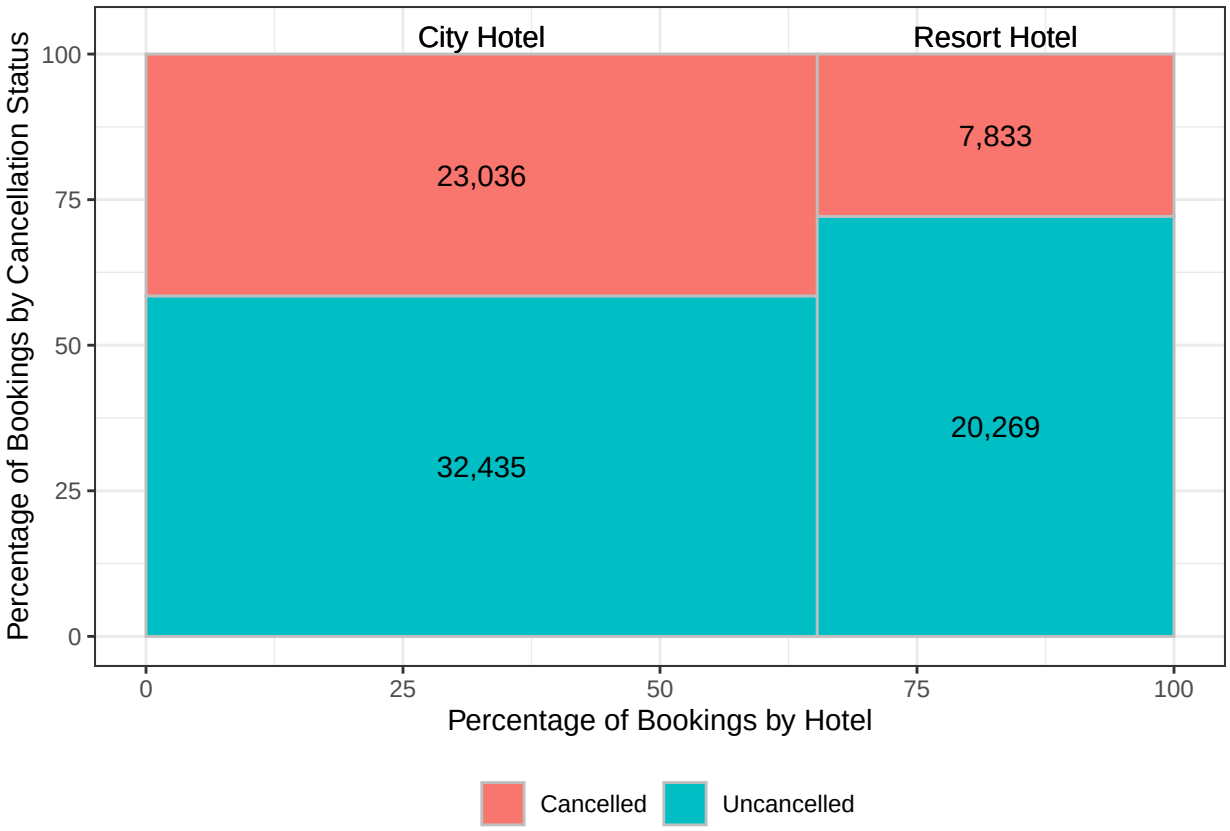
```
# Change arrival (date) variables to factors.
cleaning_recipe %<>%
    step_mutate_at(starts_with("arrival"), fn = as.factor)

# Change is_repeated_guest to be a factor variable.
cleaning_recipe %<>%
    step_mutate(is_repeated_guest = as.factor(is_repeated_guest))
```

Tables

```
mean_not <- train_clean %>% filter(is_canceled == 0) %>%
                            summarise(across(where(is.numeric), mean)) %>%
                            round(digits = 2)

mean_canceled <- train_clean %>% filter(is_canceled == 1) %>%
                            summarise(across(where(is.numeric), mean)) %>%
                            round(digits = 2)

table_1 <- as.data.frame(t(rbind(mean_canceled, mean_not)))
colnames(table_1) <- c("Uncancelled", "Cancelled")
table_1 %<>% mutate("%_Diff" = round(Uncancelled / Cancelled - 1, 2))

table_2 <- train_clean %>%
           group_by(is_canceled) %>%
           summarize(across(where(is.factor), Mode)) %>%
           t() %>% as.data.frame()

colnames(table_2) <- c("Uncancelled", "Cancelled")

table_2 %<>% filter(Uncancelled != Cancelled)
```

```
mean_city <- train_clean %>% filter(hotel == "City Hotel") %>%
                            summarise(across(where(is.numeric), mean)) %>%
                            round(digits = 2)

mean_resort <- train_clean %>% filter(hotel == "Resort Hotel") %>%
                            summarise(across(where(is.numeric), mean)) %>%
                            round(digits = 2)

table_3 <- as.data.frame(t(rbind(mean_city, mean_resort)))
colnames(table_3) <- c("City", "Resort")
table_3 %<>% mutate("%_Diff" = round(City / Resort - 1, 2))

table_4 <- train_clean %>% group_by(hotel) %>%
                            summarize(across(where(is.factor), Mode)) %>%
                            t() %>% as.data.frame()

colnames(table_4) <- c("City", "Resort")

table_4 %<>% filter(City != Resort)
```

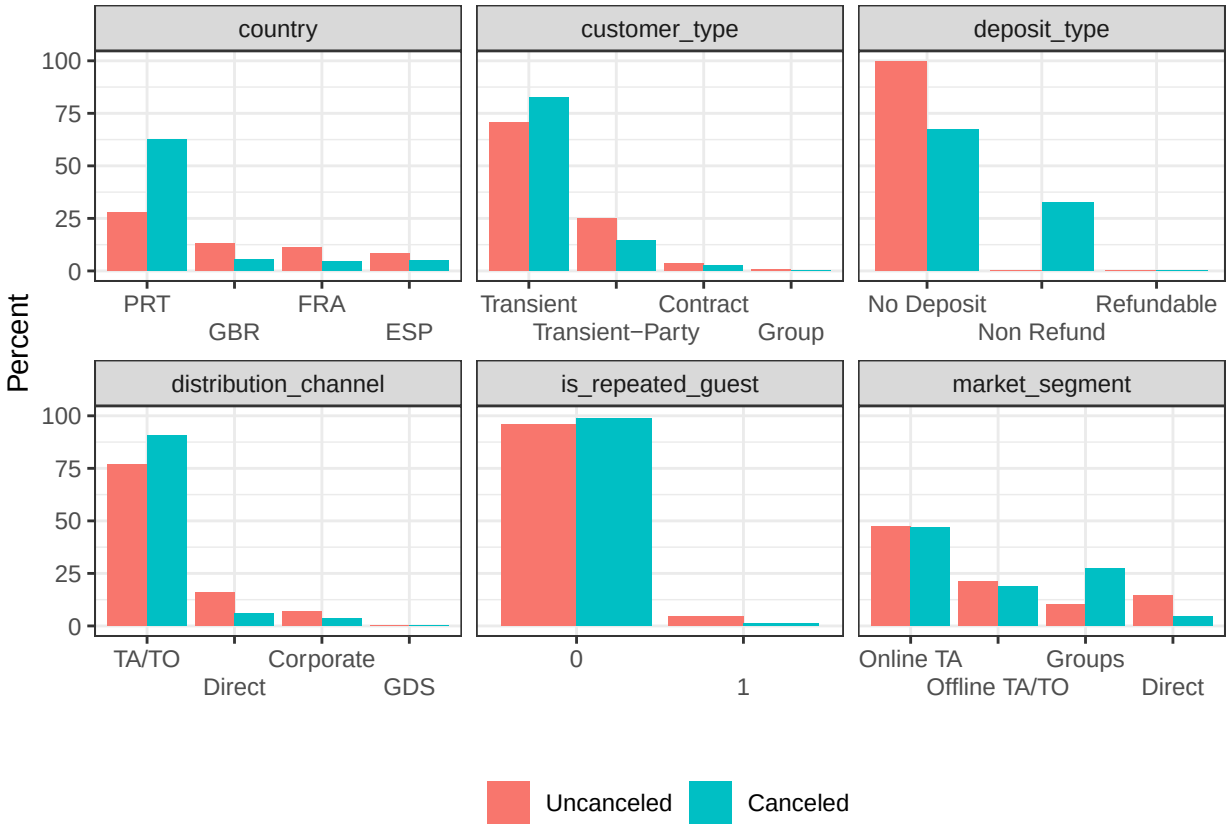
```r
marimekko_data <- train_clean %>%
                  # 2 by 2 count table
                  count(hotel, is_canceled) %>%
                  group_by(hotel) %>%
                  # percentage of cancelled in each hotel group
                  mutate(Percent_row = n * 100 / sum(n), n = NULL) %>%
                  pivot_wider(names_from = is_canceled,
                              values_from = Percent_row) %>%
                  ungroup() %>%
                  # Percent each hotel of the entire dataset
                  mutate(Percent_col = c(sum(`0`), sum(`1`))) %>%
                  # Define xmax and xmin for rectangle plots
                  mutate(Percent_col = Percent_col * 100 / sum(Percent_col)) %>%
                  mutate(xmax = cumsum(Percent_col),
                         xmin = xmax - Percent_col) %>%
                  pivot_longer(col = c(`0`, `1`)) %>%
                  # Define ymax and ymin for rectangle plots
                  group_by(hotel) %>%
                  mutate(ymax = cumsum(value), ymin = ymax - value) %>%
                  # Define text positions for labels
                  mutate(xtext = xmin + (xmax - xmin) / 2,
                         ytext = ymin + (ymax - ymin) / 2) %>%
                  mutate(name = ifelse(name == 0, "Uncancelled", "Cancelled")) %>%
                  ungroup()

#marimekko_data

# 2 by 2 counts for labels
counts <- train_clean %>% count(hotel, is_canceled) %>% select(n)

marimekko_plot <- marimekko_data %>% mutate(counts = counts$n) %>%
ggplot(
  aes(ymin = ymin, ymax = ymax, xmin = xmin, xmax = xmax, fill = name)
) + geom_rect(col = I("grey")) +
    geom_text(aes(x = xtext, y = ytext,
                  label = scales::comma(counts))) +
    geom_text(aes(x = xtext, y = 103, label = hotel), size = 4) +
    labs(x = "Percentage of Bookings by Hotel",
         y = "Percentage of Bookings by Cancellation Status",
         fill = "") +
    theme_bw() +
    theme(legend.position = "bottom")

marimekko_plot
```




```
save(file = "Figures/marimekko_plot.rds", marimekko_plot)
```

```
train_factor_plots <- train_clean %>%
                      select(is_canceled, is_repeated_guest, customer_type,
                             deposit_type,
                             distribution_channel,
                             market_segment,
                             country)

counts_by_level <- function(colname){
  train_factor_plots %>% group_by(is_canceled) %>%
                  count(!!as.name(colname)) %>%
                  mutate(Percent = n * 100 / sum(n)) %>%
                  mutate(Factor = colname) %>%
                  rename(Level = !!as.name(colname)) %>%
                  arrange(desc(Percent)) %>%
                  slice(1:4)
}

names <- colnames(train_factor_plots %>% select(-is_canceled))

train_factor_plot_data <- do.call("rbind", lapply(names, FUN = counts_by_level))

factor_plot <- ggplot(data = train_factor_plot_data,
  aes(x = reorder(Level, -Percent), y = Percent, fill = is_canceled)) +
```

```
  geom_bar(stat = "identity", position = "dodge") +
  facet_wrap(vars(Factor), scales = "free_x") +
  theme_bw() +
  labs(x = "") +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  scale_fill_discrete(labels = c("Uncanceled", "Canceled")) +
  theme(legend.title = element_blank(), legend.position = "bottom")

factor_plot
```

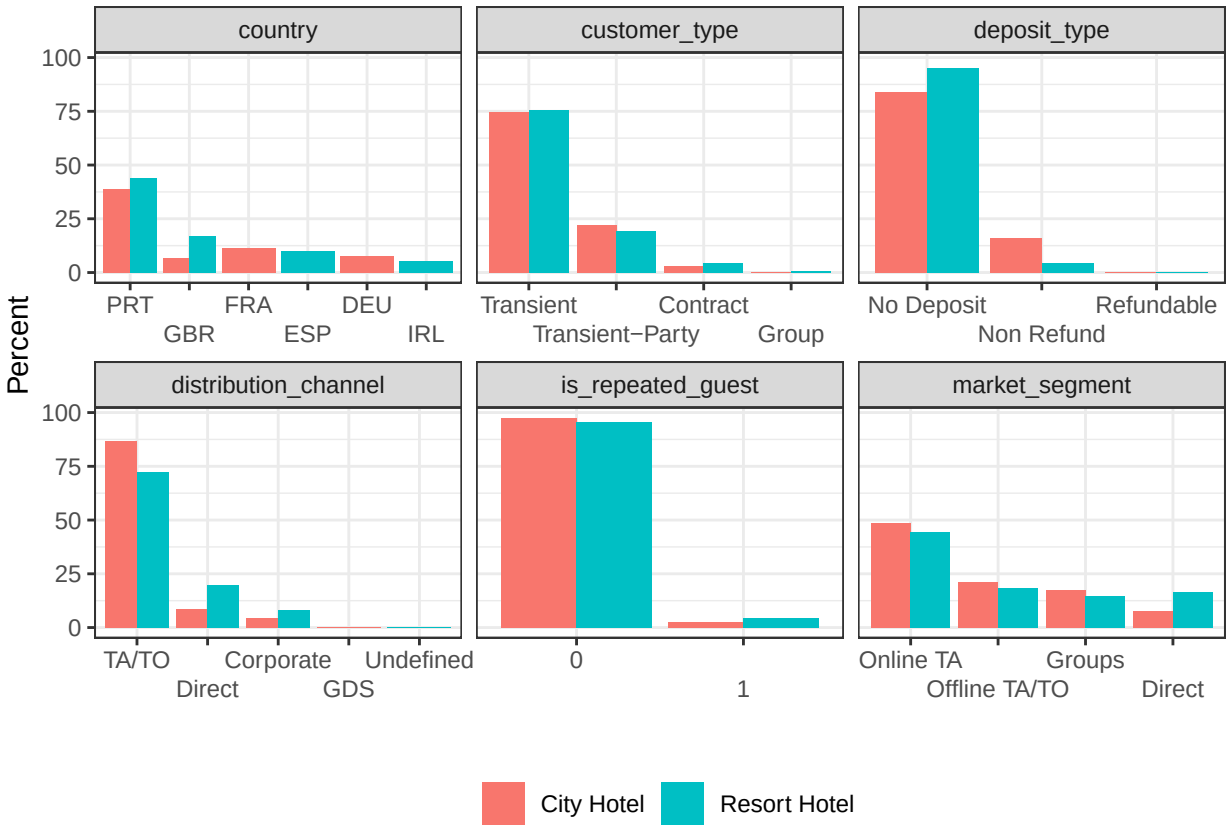



```
train_clean %>% select(arrival_date_year,
  arrival_date_month, country, is_canceled) %>%
  filter(country == "PRT" & is_canceled == 1) %>%
  count(arrival_date_year, arrival_date_month) %>%
  arrange(desc(n)) %>% print(n = 26)
```

```
## # A tibble: 26 x 3
##    arrival_date_year arrival_date_month     n
##    <fct>             <fct>              <int>
##  1 2015              9                   1324
##  2 2015              10                  1124
##  3 2016              6                   1082
##  4 2015              8                   1067
##  5 2017              5                   1034
##  6 2016              10                   991
```

```
##  7 2016              4                   955
##  8 2017              6                   936
##  9 2016              5                   934
## 10 2016              9                   867
## 11 2017              4                   846
## 12 2015              7                   842
## 13 2016              2                   747
## 14 2016              11                  711
## 15 2016              3                   657
## 16 2015              12                  646
## 17 2017              3                   623
## 18 2016              8                   533
## 19 2016              7                   526
## 20 2017              7                   496
## 21 2017              2                   463
## 22 2017              1                   429
## 23 2016              12                  383
## 24 2017              8                   382
## 25 2015              11                  342
## 26 2016              1                   308
```

```
train_factor_plots <- train_clean %>%
                      select(is_repeated_guest, customer_type,
                             deposit_type, hotel,
                             distribution_channel,
                             market_segment,
                             country)

counts_by_level_hotel <- function(colname){
  train_factor_plots %>% group_by(hotel) %>%
                  count(!!as.name(colname)) %>%
                  mutate(Percent = n * 100 / sum(n)) %>%
                  mutate(Factor = colname) %>%
                  rename(Level = !!as.name(colname)) %>%
                  arrange(desc(Percent)) %>%
                  slice(1:4)
}

names <- colnames(train_factor_plots %>% select(-hotel))

train_factor_plot_hotel <- do.call("rbind",
  lapply(names, FUN = counts_by_level_hotel))

factor_plot_hotel <- ggplot(data = train_factor_plot_hotel,
  aes(x = reorder(Level, -Percent), y = Percent, fill = hotel)) +
  geom_bar(stat = "identity", position = "dodge") +
  facet_wrap(vars(Factor), scales = "free_x") +
  theme_bw() +
  labs(x = "") +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  theme(legend.title = element_blank(), legend.position = "bottom")

factor_plot_hotel
```

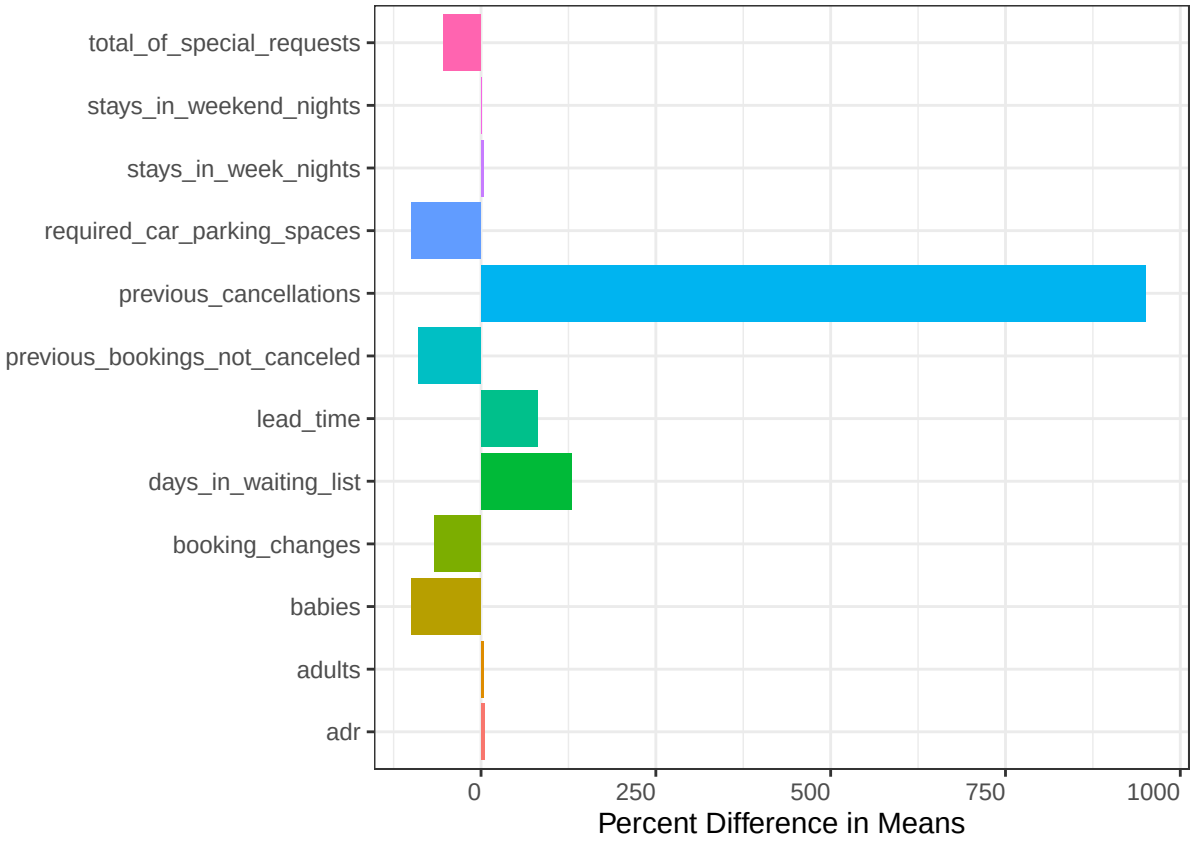



```
save(file = "Figures/factor_plot.rds", factor_plot)
save(file = "Figures/factor_plot_hotel.rds", factor_plot_hotel)
```

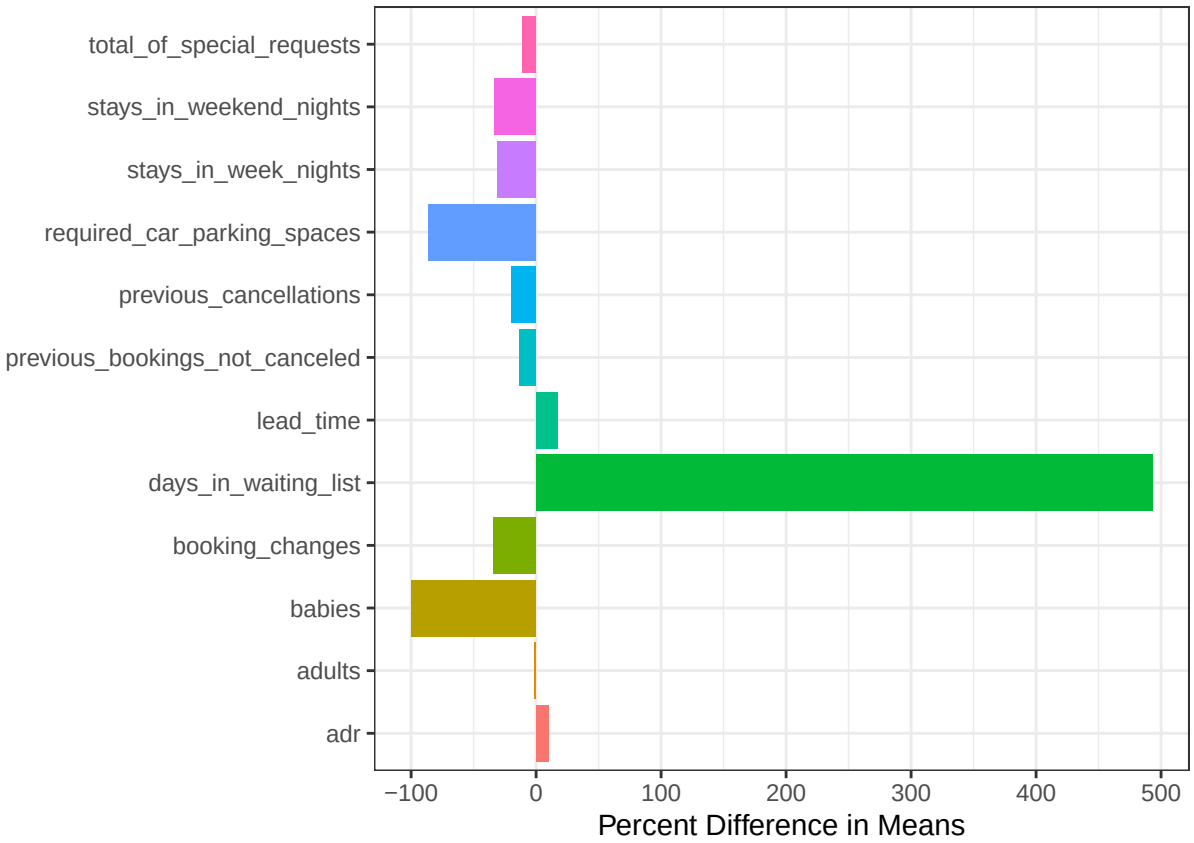
```
train_clean %>% group_by(is_canceled, reservation_status) %>%
                summarise(n = n())
train_test_perfect <- train_clean %>% select(is_canceled, reservation_status)

library(fastDummies)
train_test_perfect %<>% dummy_cols(remove_first_dummy = T,
                                   remove_selected_columns = T)

test_glm <- glm(is_canceled_1~ ., family = binomial,
                    data = train_test_perfect)

train_test_perfect$pred <- predict(test_glm, type = "response")
train_test_perfect %<>% mutate(pred = ifelse(pred >= 0.5, 1, 0))

train_test_perfect %>% ggplot(
  aes(x = is_canceled_1, y = pred, col = "blue")
) + geom_point()
```

Plots

```r
plot_1 <- table_1 %>%
   mutate("variables" = rownames(.), "diff" = `%_Diff` * 100) %>%
   filter(variables != "children") %>%
  ggplot(
    aes(x = diff, y = variables, fill = variables)
  ) + geom_bar(stat = 'identity') +
      xlim(-100, 960) +
      labs(y = "", x = "Percent Difference in Means") +
      theme_bw() + theme(legend.position = "none",
                         axis.text.x = element_text(hjust = 1))
plot_1
```




```r
plot_2 <- table_3 %>%
   mutate("variables" = rownames(.), "diff" = `%_Diff` * 100) %>%
   filter(variables != "children") %>%
  ggplot(
    aes(x = diff, y = variables, fill = variables)
  ) + geom_bar(stat = 'identity') +
      labs(y = "", x = "Percent Difference in Means") +
      theme_bw() + theme(legend.position = "none")

plot_2
```




```
caption <- "Factor modes of Cancelled v. Uncancelled reservations"
table_2_latex <- table_2 %>%
  kbl(format = 'latex', booktab = T, longtable = T, caption = caption,
    linesep = "", escape = T) %>%
    kable_styling(full_width = F)
```

```
caption <- "Factor modes of City v. Resort hotels"
table_4_latex <- table_4 %>%
  kbl(format = 'latex', booktab = T, longtable = T, caption = caption,
    linesep = "", escape = T) %>%
    kable_styling(full_width = F)
```

Model Fitting

```
# Guard against multicollinearity and overfitting.
# Not concerned about normality of skewness for logistic regression.
cleaning_recipe %<>% step_rm(reservation_status, reservation_status_date) %>%
                     step_dummy(all_nominal_predictors()) %>%
                     step_nzv(all_predictors()) %>%
                     step_corr(all_numeric_predictors()) %>%
                     step_normalize(all_numeric_predictors())
```

Model 1

```
log_Reg_mod <- logistic_reg() %>% set_engine("glm")

log_Reg_wflow <- workflow() %>%
                 add_model(log_Reg_mod) %>%
                 add_recipe(cleaning_recipe)

log_Reg_wflow
```

```
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 10 Recipe Steps
##
## * step_mutate()
## * step_mutate()
## * step_mutate()
## * step_mutate_at()
## * step_mutate()
## * step_rm()
## * step_dummy()
## * step_nzv()
## * step_corr()
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Computational engine: glm
```

```
# Saved and reloaded as rds to save runtime.
#log_Reg_fit <- log_Reg_wflow %>%
               #fit(data = train)
```

```
log_Reg_glm <- log_Reg_fit %>% extract_fit_parsnip()

#summary(log_Reg_glm)

#par(mfrow = c(2, 2))
#plot(log_Reg_glm$fit)#, which = c(1,2))
```

```
Id <- 1:length(train$hotel)
Leverage <- hatvalues(log_Reg_glm$fit)
StudRes <- studres(log_Reg_glm$fit)
CookD <- cooks.distance(log_Reg_glm$fit)

inful_data <- cbind(Id, Leverage, StudRes, CookD)
inful_data <- as.data.frame(inful_data)
```

```r
##Plots
##Leverage
lev <- ggplot(data = inful_data, aes(x = Id, y = Leverage)) + geom_point() +
            geom_hline(yintercept = 2 * length(log_Reg_glm$fit$coefficients) /
                length(inful_data$Id), col = "red") +
            labs(x = "Index") +
            theme_bw()

##Studentized Residuals
studres <- ggplot(data = inful_data, aes(x = Id, y = StudRes)) + geom_point() +
                geom_hline(yintercept = 2, col = "red") +
                geom_hline(yintercept = -2, col = "red") +
                labs(y = "Studentized Residuals", x = "Index") +
                theme_bw()

##Cooks distance
cooks <- ggplot(data = inful_data, aes(x = Id, y = CookD)) + geom_point() +
             geom_hline(yintercept = 1, col = "red") +
             labs(y = "Cook's Distance", x = "Index") +
             theme_bw()

inful <- ggarrange(lev, studres, cooks, ncol = 3, nrow = 1)

inful <- annotate_figure(inful, top = text_grob("Influential Point Analysis"))

inful

save(inful, file = "Figures/inful.rds")
```

Model 2 (Remove Outliers)

```r
train_2 <- train %>%
    mutate(m1cooksD = cooks.distance(log_Reg_glm$fit))
dim(train_2)
train_2 %<>% filter(m1cooksD < 1)
```

Model 3 (Split Model)

```r
# Define data splits
train_city <- train %>% filter(hotel == "City Hotel")
train_resort <- train %>% filter(hotel == "Resort Hotel")
test_city <- test %>% filter(hotel == "City Hotel")
test_resort <- test %>% filter(hotel == "Resort Hotel")
```

```r
recipe_3 <- cleaning_recipe
rm_hotel <- train %>% recipe(formula = is_canceled ~ . ) %>%
                      step_rm(hotel)
recipe_3$steps <- append(recipe_3$steps, list(rm_hotel$steps[[1]]), after = 6)
```

```
model_3_wflow <- workflow() %>%
                  add_model(log_Reg_mod) %>%
                  add_recipe(recipe_3)
```

```
model_3_fit_city <- model_3_wflow %>% fit(data = train_city)
model_3_fit_resort <- model_3_wflow %>% fit(data = train_resort)
```

Model 4 (Penalized)

```
# penalty = lambda
# mixture = alpha
# Define elastic net model.
pen_log_mod <- logistic_reg(penalty = tune(), mixture = tune()) %>%
               set_engine("glmnet")

# Define resampling method for hyperparameter tunning.
resamples <- train %>% vfold_cv(v = 10, repeats = 2)

# Define grid of parameters to test.
param_grid <- grid_regular(penalty(), mixture(),
                           levels = list(penalty = 100,
                                         mixture = 10))

model_4_wflow <- workflow() %>%
                 add_model(pen_log_mod) %>%
                 add_recipe(cleaning_recipe)
#tic()
#model_4_fit <- model_4_wflow %>%
               #tune_grid(resamples = resamples, grid = param_grid,
                         #metrics = metric_set(roc_auc))
#toc()
# model fit was saved an reloaded as a rds to save runtime.
```

```
model_4_best_params <- select_best(model_4_fit)
```

Model Predictions and Evaluation

Model 1

```
model_1_train <- predict(log_Reg_fit, new_data = train, type = "prob")[2]
model_1_train$obs <- train$is_canceled
model_1_cut <- model_1_train %>%
  cutpointr(x = .pred_1, class = obs,
            method = maximize_metric, metric = cohens_kappa,
            pos_class = 1, direction = ">=")
model_1_train %<>%
  mutate(pred.class = ifelse(.pred_1 >= model_1_cut$optimal_cutpoint, 1, 0)) %>%
  mutate(obs = as.factor(obs), pred.class = as.factor(pred.class))
```

```
model_1_train_cmat <- model_1_train %>%
    conf_mat(truth = obs, estimate = pred.class)

model_1_train_metrics <- summary(model_1_train_cmat)
```

```
model_1_test <- predict(log_Reg_fit, new_data = test, type = "prob")[2]
model_1_test$obs <- test$is_canceled

model_1_test %<>%
  mutate(pred.class = ifelse(.pred_1 >= model_1_cut$optimal_cutpoint, 1, 0)) %>%
  mutate(obs = as.factor(obs), pred.class = as.factor(pred.class))


model_1_test_cmat <- model_1_test %>%
    conf_mat(truth = obs, estimate = pred.class)

summary(model_1_test_cmat)
```

```
## # A tibble: 13 x 3
##    .metric              .estimator .estimate
##    <chr>                <chr>          <dbl>
##  1 accuracy             binary         0.815
##  2 kap                  binary         0.609
##  3 sens                 binary         0.835
##  4 spec                 binary         0.781
##  5 ppv                  binary         0.865
##  6 npv                  binary         0.738
##  7 mcc                  binary         0.610
##  8 j_index              binary         0.616
##  9 bal_accuracy         binary         0.808
## 10 detection_prevalence binary         0.605
## 11 precision            binary         0.865
## 12 recall               binary         0.835
## 13 f_meas               binary         0.850
```

```
model_1_train %<>% mutate(hotel = train$hotel)
model_1_test %<>% mutate(hotel = test$hotel)

model_1_train_city <- model_1_train %>% filter(hotel == "City Hotel")
model_1_test_city <- model_1_test %>% filter(hotel == "City Hotel")
model_1_train_resort <- model_1_train %>% filter(hotel == "Resort Hotel")
model_1_test_resort <- model_1_test %>% filter(hotel == "Resort Hotel")


model_1_train_cmat_city <- model_1_train_city %>%
    conf_mat(truth = obs, estimate = pred.class)
model_1_train_metrics_city <- summary(model_1_train_cmat_city)

model_1_test_cmat_city <- model_1_test_city %>%
    conf_mat(truth = obs, estimate = pred.class)
model_1_test_metrics_city <- summary(model_1_test_cmat_city)
```

```
model_1_train_cmat_resort <- model_1_train_resort %>%
    conf_mat(truth = obs, estimate = pred.class)
model_1_train_metrics_resort <- summary(model_1_train_cmat_resort)

model_1_test_cmat_resort <- model_1_test_resort %>%
    conf_mat(truth = obs, estimate = pred.class)
model_1_test_metrics_resort <- summary(model_1_test_cmat_resort)
```

Model 3

```
model_3_train_city <- predict(model_3_fit_city,
    new_data = train_city, type = "prob")[2]

model_3_train_city$obs <- train_city$is_canceled

model_3_cut <- model_3_train_city %>%
  cutpointr(x = .pred_1, class = obs,
            method = maximize_metric, metric = cohens_kappa,
            pos_class = 1, direction = ">=")

model_3_train_city %<>%
  mutate(pred.class = ifelse(.pred_1 >= model_1_cut$optimal_cutpoint, 1, 0)) %>%
  mutate(obs = as.factor(obs), pred.class = as.factor(pred.class))

model_3_train_city_cmat <- model_3_train_city %>%
    conf_mat(truth = obs, estimate = pred.class)

model_3_train_city_metrics <- summary(model_3_train_city_cmat)

model_3_train_resort <- predict(model_3_fit_resort,
    new_data = train_resort, type = "prob")[2]

model_3_train_resort$obs <- train_resort$is_canceled

model_3_cut <- model_3_train_resort %>%
  cutpointr(x = .pred_1, class = obs,
            method = maximize_metric, metric = cohens_kappa,
            pos_class = 1, direction = ">=")
model_3_train_resort %<>%
  mutate(pred.class = ifelse(.pred_1 >= model_1_cut$optimal_cutpoint, 1, 0)) %>%
  mutate(obs = as.factor(obs), pred.class = as.factor(pred.class))

model_3_train_resort_cmat <- model_3_train_resort %>%
    conf_mat(truth = obs, estimate = pred.class)

model_3_train_resort_metrics <- summary(model_3_train_resort_cmat)
```

```
model_3_test_city <- predict(model_3_fit_city,
                             new_data = test_city, type = "prob")[2]
model_3_test_city$obs <- test_city$is_canceled
```

```
model_3_test_city %<>%
  mutate(pred.class = ifelse(.pred_1 >= model_1_cut$optimal_cutpoint, 1, 0)) %>%
  mutate(obs = as.factor(obs), pred.class = as.factor(pred.class))

model_3_test_city_cmat <- model_3_test_city %>%
    conf_mat(truth = obs, estimate = pred.class)

model_3_test_city_metrics <- summary(model_3_test_city_cmat)

model_3_test_resort <- predict(model_3_fit_resort,
                               new_data = test_resort, type = "prob")[2]
model_3_test_resort$obs <- test_resort$is_canceled

model_3_test_resort %<>%
  mutate(pred.class = ifelse(.pred_1 >= model_1_cut$optimal_cutpoint, 1, 0)) %>%
  mutate(obs = as.factor(obs), pred.class = as.factor(pred.class))

model_3_test_resort_cmat <- model_3_test_resort %>%
    conf_mat(truth = obs, estimate = pred.class)

model_3_test_resort_metrics <- summary(model_3_test_resort_cmat)
```

More Figures

```
table_5 <- cbind(model_1_train_metrics_city$.estimate,
                 model_1_test_metrics_city$.estimate,
                 model_1_train_metrics_resort$.estimate,
                 model_1_test_metrics_resort$.estimate,
                 model_3_train_city_metrics$.estimate,
                 model_3_test_city_metrics$.estimate,
                 model_3_train_resort_metrics$.estimate,
                 model_3_test_resort_metrics$.estimate)

table_5 <- as.data.frame(table_5) %>% slice(1:6)

rownames(table_5) <- c("Accuracy", "Kappa", "Sensitivity", "Specificity",
                       "PPV", "NPV")

colnames(table_5) <- c("Train", "Test",
                       "Train", "Test",
                       "Train", "Test",
                       "Train", "Test")


caption <- "Summary of model metrics"

table_5_latex <- table_5 %>%
  kbl(format = "latex", booktabs = T, longtable = T, caption = caption,
    linesep = "", escape = T, digits = 4) %>%
    kable_styling(full_width = F) %>%
    add_header_above(c(" ", "City" = 2, "Resort" = 2,
```

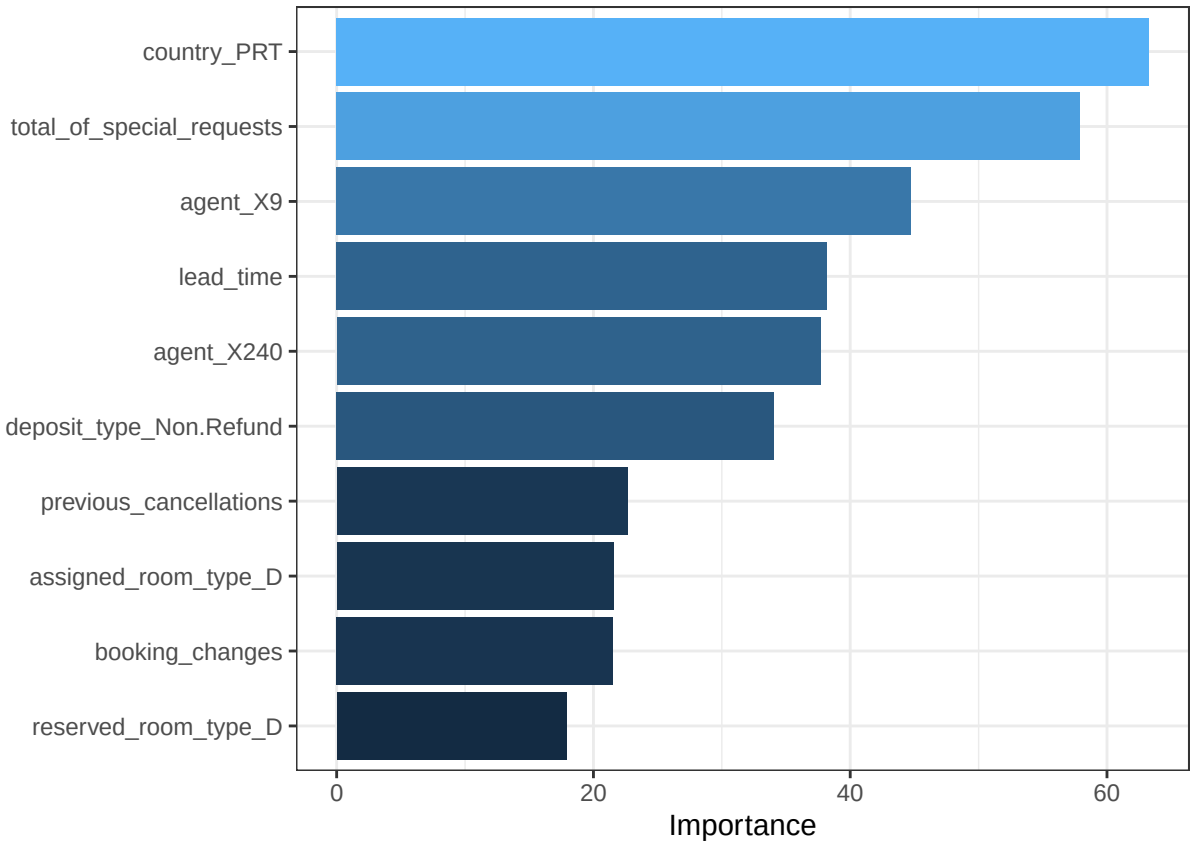
```
                              "City" = 2, "Resort" = 2)) %>%
    add_header_above(c(" ", "Combined Model" = 4, "Split Model" = 4))

table_5_latex
```

Table 3: Summary of model metrics

	Combined Model				Split Model			
	City		Resort		City		Resort	
	Train	Test	Train	Test	Train	Test	Train	Test
Accuracy	0.8067	0.8038	0.8392	0.8369	0.7993	0.7981	0.8330	0.8302
Kappa	0.6062	0.6018	0.6036	0.5931	0.5896	0.5883	0.5941	0.5826
Sensitivity	0.8078	0.8034	0.8827	0.8848	0.8113	0.8113	0.8696	0.8712
Specificity	0.8051	0.8043	0.7265	0.7108	0.7824	0.7801	0.7384	0.7223
PPV	0.8537	0.8489	0.8931	0.8895	0.8400	0.8348	0.8959	0.8918
NPV	0.7484	0.7494	0.7053	0.7012	0.7465	0.7512	0.6863	0.6810

```
save(file = "Figures/table_5.rds", table_5_latex)
```

```
# Variable Importance
plot_4 <- log_Reg_glm %>%
          vip(num_features = 10) %+%
          aes(x = reorder(Variable, Importance), y = Importance,
              fill = Importance) +
          scale_fill_gradient() +
          labs(x = "") +
          theme_bw() + theme(legend.position = "none")

plot_4
```




```
save(file = "Figures/plot_4.rds", plot_4)
```

```
caption <- "Final Model Coefficients"
model_1_coef <- log_Reg_glm %>% tidy()
# Fixes a latex table issue
model_1_coef$term[1] <- "Intercept"
model_1_coef %<>%
    kbl(format = "latex", booktabs = T, longtable = T, caption = caption,
      linesep = "", escape = T, digits = 3) %>%
      kable_styling(full_width = F)

model_1_coef
```

Table 4: Final Model Coefficients

term	estimate	std.error	statistic	p.value
Intercept	-1.764	2.947	-0.598	0.550
lead_time	0.561	0.015	38.194	0.000
stays_in_weekend_nights	0.125	0.012	10.804	0.000
stays_in_week_nights	0.181	0.012	15.168	0.000
adults	0.095	0.013	7.531	0.000
previous_cancellations	0.954	0.042	22.653	0.000
booking_changes	-0.270	0.013	-21.545	0.000
adr	0.195	0.015	13.170	0.000

required_car_parking_spaces	-5.568	11.613	-0.479	0.632
total_of_special_requests	-0.681	0.012	-57.873	0.000
hotel_Resort.Hotel	-0.144	0.016	-8.981	0.000
arrival_date_year_X2016	0.191	0.018	10.832	0.000
arrival_date_year_X2017	0.326	0.020	16.127	0.000
arrival_date_month_X2	0.053	0.015	3.468	0.001
arrival_date_month_X3	-0.007	0.016	-0.406	0.685
arrival_date_month_X4	0.067	0.017	3.947	0.000
arrival_date_month_X5	0.029	0.018	1.620	0.105
arrival_date_month_X6	-0.020	0.017	-1.169	0.242
arrival_date_month_X7	-0.029	0.019	-1.557	0.120
arrival_date_month_X8	0.019	0.020	0.979	0.327
arrival_date_month_X9	0.049	0.019	2.577	0.010
arrival_date_month_X10	0.132	0.019	7.123	0.000
arrival_date_month_X11	0.109	0.016	7.023	0.000
arrival_date_month_X12	0.102	0.016	6.580	0.000
meal_HB	-0.031	0.011	-2.731	0.006
meal_SC	0.031	0.010	3.230	0.001
country_DEU	-0.194	0.011	-17.219	0.000
country_ESP	0.084	0.011	7.760	0.000
country_FRA	-0.136	0.011	-12.318	0.000
country_GBR	-0.099	0.011	-8.722	0.000
country_PRT	0.922	0.015	63.255	0.000
market_segment_Direct	0.058	0.035	1.636	0.102
market_segment_Groups	0.205	0.037	5.529	0.000
market_segment_Offline.TA.TO	-0.023	0.041	-0.569	0.570
market_segment_Online.TA	0.067	0.052	1.281	0.200
distribution_channel_Direct	-0.204	0.034	-5.976	0.000
distribution_channel_TA.TO	-0.223	0.034	-6.647	0.000
reserved_room_type_D	0.339	0.019	17.916	0.000
reserved_room_type_E	0.236	0.020	11.854	0.000
assigned_room_type_D	-0.427	0.020	-21.572	0.000
assigned_room_type_E	-0.259	0.021	-12.518	0.000
deposit_type_Non.Refund	1.526	0.045	34.056	0.000
agent_X240	0.610	0.016	37.693	0.000
agent_X9	0.838	0.019	44.747	0.000
agent_NULL.	-0.087	0.018	-4.752	0.000
company_NULL.	0.208	0.022	9.528	0.000
customer_type_Transient	0.377	0.028	13.317	0.000
customer_type_Transient.Party	0.078	0.028	2.729	0.006

```
save(file = "Figures/model_1_coef.rds", model_1_coef)
```

```
model_1_roc <- layer_data(plot_roc(model_1_cut), 2) %>%
               select(x, y) %>% mutate(Model = "Model 1 Full")

model_1_roc <- layer_data(plot_roc(model_1_cut), 2) %>%
               select(x, y) %>% mutate(Model = "Model 1 Full")
```