ECON 490 Project: Predicting Hotel Cancelations

Adam Wisowaty (adamtw2)

# Introduction

For this project I investigated if we can predict hotel cancelations based on other factors. One reason why I think this is important is that it shows the general sentiment regarding travel around the country. If a lot people are canceling their hotel reservations, then this means people aren't traveling and the economy might not be doing very well. I also think it would be important for hotel management to be able to predict which reservations have a high probability of being canceled as this would allow them to adjust staff hours and project revenue.

# Data Wrangling and Cleaning

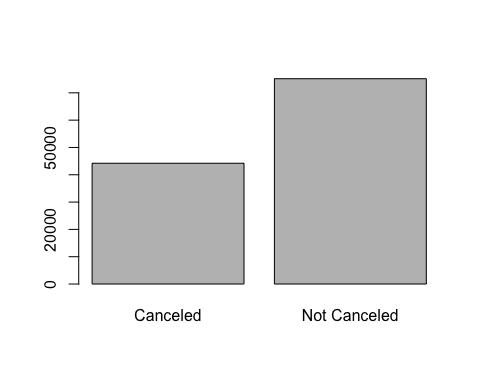
The data set I used was the Hotel Booking Demand Dataset from Kaggle (<https://www.kaggle.com/jessemostipak/hotel-booking-demand>). The first thing I had to do when creating the final data set was remove NA values using na.omit(). Next I created dummy variables for some of the data. For example, the children variable had values ranging from 0 to 10, but I just wanted an indicator that would show if any kids were part of the reservation. After creating the final data set, I set the seed to 490 and created the train/test split.

library(tidyverse)  
library(glmnet)  
library(dplyr)  
library(caret)  
library(e1071)  
library(keras)  
library(plotROC)  
library(class)  
library(MASS)   
  
project\_data = read\_csv("hotel\_bookings.csv")  
project\_data = na.omit(project\_data)  
  
project\_data$dummy\_children = if\_else(project\_data$children>0 | project\_data$babies>0, 1, 0)  
project\_data$dummy\_requests = if\_else(project\_data$total\_of\_special\_requests>0, 1, 0)  
project\_data$dummy\_previous\_cancellations = if\_else(project\_data$previous\_cancellations>0,1,0)  
project\_data$dummy\_parking = if\_else(project\_data$required\_car\_parking\_spaces>0,1,0)  
project\_data$is\_canceled\_factor = as.factor(if\_else(project\_data$is\_canceled == 1, "Canceled", "Not Canceled"))  
  
set.seed(490)  
train = sample\_frac(project\_data, .8)  
test = anti\_join(project\_data, train)

# EDA

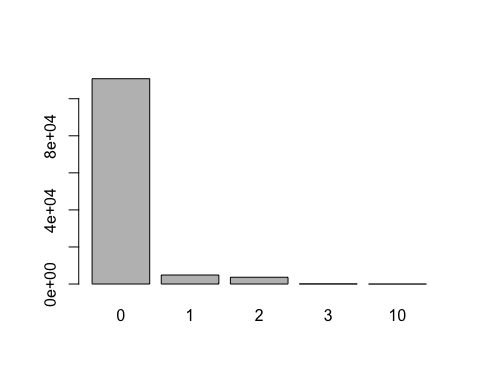
This barplot shows the distribution of the canceled and not canceled hotel reservations. Clearly there are significantly more reservations that are not canceled.

barplot(table(project\_data$is\_canceled\_factor))



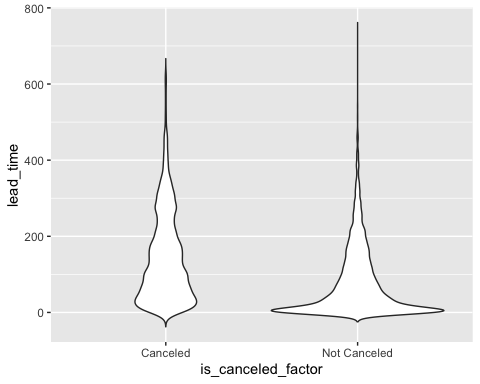
This barplot shows the number of children in each booking. Not many bookings had kids, but the dummy variable for children was one of the most significant variables chosen by the lasso. Bookings with kids were less likely to be canceled.

barplot(table(project\_data$children))



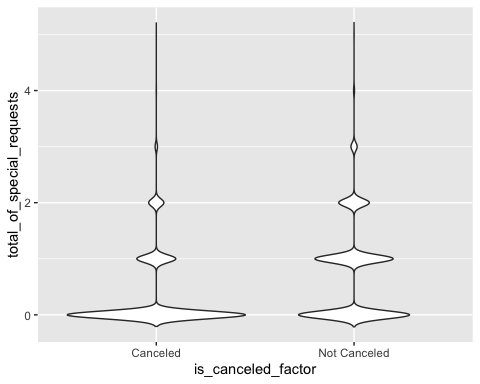
This violin plot shows that bookings that weren't canceled were likely to have shorter lead time.

ggplot(project\_data, aes(x=is\_canceled\_factor, y=lead\_time)) +   
 geom\_violin(trim=FALSE)



This violin plot shows that the reservations without special request were more likely to be canceled.

ggplot(project\_data, aes(x=is\_canceled\_factor, y=total\_of\_special\_requests)) +   
 geom\_violin(trim=FALSE)



# Inference

The top 3 significant coefficients for this model are the dummy variables for children, requests and previous cancellations. The coefficients for the children and previous cancelations are negative, meaning that these people are less likely to cancel their reservations. The coefficient for requests is positive which suggests that these people who make special requests are more likely to cancel their reservation.

trControl = trainControl(method = 'repeatedcv',  
 repeats = 5,  
 number = 5)  
  
x = model.matrix(is\_canceled\_factor ~ hotel + dummy\_children + lead\_time + dummy\_requests + as.factor(arrival\_date\_year) + is\_repeated\_guest + dummy\_previous\_cancellations + dummy\_parking, data = train)  
y = train$is\_canceled\_factor  
  
fitl = train(x,  
 y,  
 method = 'glmnet',  
 family = "binomial",  
 trControl = trControl,  
 tuneGrid = expand.grid(alpha = 1, lambda = 10^seq(-5,2,length = 50)))  
fitl$bestTune

## alpha lambda  
## 10 1 0.0001930698

a = fitl$bestTune$alpha  
l = fitl$bestTune$lambda  
best\_model = glmnet(x, y, alpha = a, lambda = l, family = "binomial")  
  
final\_model = glm(is\_canceled\_factor ~ I(hotel == 'Resort Hotel') + dummy\_children + lead\_time + dummy\_requests + I(as.factor(arrival\_date\_year) == '2016') + I(as.factor(arrival\_date\_year) == '2017') + is\_repeated\_guest + dummy\_previous\_cancellations + dummy\_parking, family = "binomial", data = train)  
  
summary(final\_model)

##   
## Call:  
## glm(formula = is\_canceled\_factor ~ I(hotel == "Resort Hotel") +   
## dummy\_children + lead\_time + dummy\_requests + I(as.factor(arrival\_date\_year) ==   
## "2016") + I(as.factor(arrival\_date\_year) == "2017") + is\_repeated\_guest +   
## dummy\_previous\_cancellations + dummy\_parking, family = "binomial",   
## data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.2981 -1.0412 0.5664 0.8839 2.8373   
##   
## Coefficients:  
## Estimate Std. Error z value  
## (Intercept) 1.026e+00 2.298e-02 44.645  
## I(hotel == "Resort Hotel")TRUE 3.115e-01 1.677e-02 18.572  
## dummy\_children -3.518e-01 2.834e-02 -12.415  
## lead\_time -4.134e-03 7.526e-05 -54.922  
## dummy\_requests 1.109e+00 1.629e-02 68.088  
## I(as.factor(arrival\_date\_year) == "2016")TRUE -5.365e-01 2.396e-02 -22.390  
## I(as.factor(arrival\_date\_year) == "2017")TRUE -7.561e-01 2.492e-02 -30.346  
## is\_repeated\_guest 3.299e+00 1.157e-01 28.510  
## dummy\_previous\_cancellations -4.481e+00 1.039e-01 -43.110  
## dummy\_parking 1.665e+01 4.681e+01 0.356  
## Pr(>|z|)   
## (Intercept) <2e-16 \*\*\*  
## I(hotel == "Resort Hotel")TRUE <2e-16 \*\*\*  
## dummy\_children <2e-16 \*\*\*  
## lead\_time <2e-16 \*\*\*  
## dummy\_requests <2e-16 \*\*\*  
## I(as.factor(arrival\_date\_year) == "2016")TRUE <2e-16 \*\*\*  
## I(as.factor(arrival\_date\_year) == "2017")TRUE <2e-16 \*\*\*  
## is\_repeated\_guest <2e-16 \*\*\*  
## dummy\_previous\_cancellations <2e-16 \*\*\*  
## dummy\_parking 0.722   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 125981 on 95508 degrees of freedom  
## Residual deviance: 100510 on 95499 degrees of freedom  
## AIC: 100530  
##   
## Number of Fisher Scoring iterations: 16

# Prediction

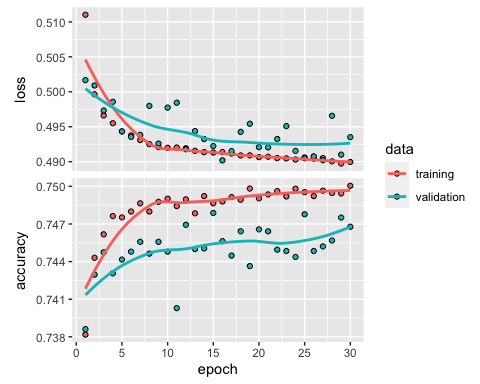
final\_predicted\_probabilities <- predict(final\_model, newdata = test, type = "response")  
final\_predicted\_classes <- ifelse(final\_predicted\_probabilities > 0.5, "Canceled", "Not Canceled")  
  
mean(final\_predicted\_classes == test$is\_canceled\_factor)

## [1] 0.2563121

f = is\_canceled\_factor ~ hotel + as.factor(arrival\_date\_year) + dummy\_children + lead\_time + dummy\_requests + is\_repeated\_guest + dummy\_previous\_cancellations + dummy\_parking - 1  
  
x\_train = model.matrix(f, train) %>% scale  
x\_test = model.matrix(f, test) %>% scale  
  
# Recall if you are predicting a categorical variable, it needs to be zero indexed  
# and you need to initialize it with the keras function to\_categorical()  
y\_train = to\_categorical(y = train$is\_canceled - 1, num\_classes = 2)  
y\_test = to\_categorical(y = test$is\_canceled - 1, num\_classes = 2)  
  
  
n = ncol(x\_train)  
neural\_model = keras\_model\_sequential()  
neural\_model %>%  
 layer\_dense(units = 70, activation = 'relu', input\_shape = n) %>%  
 layer\_dense(units = 35, activation = 'relu') %>%  
 layer\_dense(units = 2, activation = 'softmax')  
  
neural\_model %>% compile(  
 loss = "categorical\_crossentropy",  
 optimizer = optimizer\_rmsprop(),  
 metrics = c("accuracy")  
)  
  
history1 <- neural\_model %>% fit(  
 x\_train, y\_train,  
 epochs = 30, batch\_size = 50,  
 validation\_split = .2  
)

plot(history1)

## `geom\_smooth()` using formula 'y ~ x'



neural\_model %>% evaluate(x\_test, y\_test)

## $loss  
## [1] 0.6006147  
##   
## $accuracy  
## [1] 0.724983

x\_train = model.matrix(is\_canceled\_factor ~ hotel + dummy\_children + lead\_time + dummy\_requests + as.factor(arrival\_date\_year) + is\_repeated\_guest + dummy\_previous\_cancellations + dummy\_parking, data = train)  
x\_test = model.matrix(is\_canceled\_factor ~ hotel + dummy\_children + lead\_time + dummy\_requests + as.factor(arrival\_date\_year) + is\_repeated\_guest + dummy\_previous\_cancellations + dummy\_parking, data = test)  
y\_train = train$is\_canceled\_factor  
  
K = c(2:6, 10, 15, 20, 30, 50)  
for(k in K){  
 kfit = knn(x\_train, x\_test, y\_train, k)  
 accuracy = mean(kfit == test$is\_canceled\_factor)  
 cat('k =', k, ' Accuracy =', accuracy, '\n')  
}

## k = 2 Accuracy = 0.7329465   
## k = 3 Accuracy = 0.7327613   
## k = 4 Accuracy = 0.7320822   
## k = 5 Accuracy = 0.732823   
## k = 6 Accuracy = 0.7326378   
## k = 10 Accuracy = 0.7338725   
## k = 15 Accuracy = 0.732823   
## k = 20 Accuracy = 0.7314649   
## k = 30 Accuracy = 0.7270819   
## k = 50 Accuracy = 0.7231311

# Comparison

In terms of interpretability, it would be easy enough to calculate the marginal effect of each variable on the probability of the logistic regression, and KNN just looks at the surrounding observations to make a prediction, but the neural network has a bit more going on behind the scenes, so I think it might be hard for someone who doesn't know how it works to understand it. In terms of flexibility, I think the logistic regression and neural network are superior to KNN because KNN isn't particularly complex (though it still managed to get pretty good accuracy). None of these models performed to the level I would have hoped. The best model was the neural network which managed to correctly classify 73.45% of cancelations, it was closely followed by the KNN with k=4 which managed to correctly classify 73.06% of cancelations. The logistic regression model had a horrific accuracy of just 26%. I think it is difficult to say that hotel cancelations could be predicted with any certainty given the variables I used.

# Conclusion

To summarize, I investigated whether hotel cancelations can be correctly classified based on other factors by using logistic regression, KNN and neural network models. To examine this question I used the Hotel Booking Demand Dataset from Kaggle. The best performing model turned out to be the neural network with two hidden layers(70 and 35 neurons) that used ReLU and an output layer that used softmax. The three most important predictors turned out to be the dummy variables for children, special requests and previous cancellations. I think this was a really interesting problem to look at, even though I wasn't able to achieve the accuracy that I wanted from my models. Once the data is available for 2020, I think it would be fascinating to go back and see if COVID-19 had a major impact on hotel cancelations.