Classification

Load the data; split train/test

Dataset: Hotel Bookings/Cancellations via Kaggle.

Data is loaded and cleaned here. I put these steps together since it was easier for me to work with all consolidated in one place, but most of the time I spent with data exploration involved making observations about the dataframe as I was handling factors, subset, etc.

```
df <- read.csv("hotel_booking.csv", header=TRUE)
set.seed(1234)
df <- df[sample(1:nrow(df), 10000, replace=FALSE),]
df$hotel <- factor(df$hotel)
df$arrival_date_month <- factor(df$arrival_date_month)
df$meal <- factor(df$meal)
df$market_segment <- factor(df$market_segment)
df$distribution_channel <- factor(df$distribution_channel)
df$reserved_room_type <- factor(df$reserved_room_type)
df$assigned_room_type <- factor(df$assigned_room_type)
df$deposit_type <- factor(df$deposit_type)
df$customer_type <- factor(df$customer_type)
df$reservation_status <- factor(df$reservation_status)
df$reservation_status_date <- as.Date(df$reservation_status_date)
df <- subset(df, select=-c(agent, company, country, credit_card, email, name, phone.number, reservation

i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Explore data

Unfortunately the hotel cancellation data is difficult to explore graphically, as none of the attributes of various cancellations make intuitive sense in a visual space. Nevertheless, we can get a good idea of the types of data we're dealing with by simply looking at some of the entries in the dataframe.

```
str(df)
```

```
## 'data.frame':    10000 obs. of  28 variables:
##  $ hotel                         : Factor w/ 2 levels "City Hotel","Resort Hotel": 1 1 1 2 2 1 1 1 1
##  $ is_canceled                   : int  0 0 0 0 0 0 0 1 0 1 ...
##  $ lead_time                     : int  48 17 1 0 27 95 1 176 8 20 ...
##  $ arrival_date_year             : int  2017 2017 2015 2017 2017 2016 2016 2016 2016 2017 ...
##  $ arrival_date_month            : Factor w/ 12 levels "April","August",..: 4 4 2 5 4 10 5 1 8 9 ...
##  $ arrival_date_week_number      : int  8 8 35 2 9 46 3 15 11 20 ...
##  $ arrival_date_day_of_month     : int  22 24 28 10 28 7 13 9 11 14 ...
```

```
##  $ stays_in_weekend_nights       : int  0 0 1 0 0 1 0 1 1 1 ...
##  $ stays_in_week_nights          : int  2 1 2 3 1 2 1 1 2 0 ...
##  $ adults                        : int  1 2 2 1 2 2 1 2 1 2 ...
##  $ children                      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ babies                        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ meal                          : Factor w/ 5 levels "BB","FB","HB",..: 1 4 1 1 1 1 1 3 1 1 ...
##  $ market_segment                : Factor w/ 7 levels "Aviation","Complementary",..: 5 7 7 3 7 7 4 6
##  $ distribution_channel          : Factor w/ 4 levels "Corporate","Direct",..: 4 4 4 1 4 4 2 4 4 1 .
##  $ is_repeated_guest             : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ previous_cancellations        : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ previous_bookings_not_canceled: int  0 0 0 8 0 0 0 0 0 0 ...
##  $ reserved_room_type            : Factor w/ 8 levels "A","B","C","D",..: 1 1 1 1 4 4 1 1 2 4 ...
##  $ assigned_room_type            : Factor w/ 10 levels "A","B","C","D",..: 1 1 1 4 4 4 1 1 2 4 ...
##  $ booking_changes               : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ deposit_type                  : Factor w/ 3 levels "No Deposit","Non Refund",..: 1 1 1 1 1 1 1 2 
##  $ days_in_waiting_list          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ customer_type                 : Factor w/ 4 levels "Contract","Group",..: 4 3 1 3 4 3 3 3 4 3 ...
##  $ adr                           : num  65 97 106 35 56 ...
##  $ required_car_parking_spaces   : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ total_of_special_requests     : int  1 2 2 0 2 1 0 0 1 0 ...
##  $ reservation_status_date       : Date, format: "2017-02-24" "2017-02-25" ...
```

Logistic regression

```
glm1 <- glm(is_canceled~., data=train, family="binomial")
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
glm1_probs <- predict(glm1, newdata=test, type="response")
glm1_pred <- ifelse(glm1_probs > 0.5, 1, 0)
glm1_acc <- mean(glm1_pred == test$is_canceled)
print(paste("accuracy = ", glm1_acc))
```

```
## [1] "accuracy =  0.876"
```

```
table(glm1_pred, test$is_canceled)
```

```
##
## glm1_pred    0    1
##         0 1209  139
##         1  109  543
```

kNN

```
library(class)
library(fastDummies)
knn1_train <- dummy_cols(train, remove_selected_columns=TRUE)
knn1_train$reservation_status_date <- as.numeric(knn1_train$reservation_status_date)
knn1_train_scaled <- scale(knn1_train)
knn1_test <- dummy_cols(test, remove_selected_columns=TRUE)
knn1_test$reservation_status_date <- as.numeric(knn1_test$reservation_status_date)
knn1_test_scaled <- scale(knn1_test)
knn1_pred <- knn(knn1_train_scaled, knn1_test_scaled, cl=knn1_train$is_canceled, k=3)
knn1_results <- knn1_pred == knn1_test$is_canceled
knn1_acc <- length(which(knn1_results == TRUE)) / length(knn1_results)
print(paste("accuracy = ", knn1_acc))
```

```
## [1] "accuracy =  0.9095"
```

```
table(knn1_results, knn1_pred)
```

```
##             knn1_pred
## knn1_results    0    1
##        FALSE  112   69
##        TRUE  1249  570
```

Decision trees

```
library(rpart)
tree1 <- rpart(is_canceled~., data=train, method="class")
tree1_pred <- predict(tree1, newdata=test, type="class")
tree1_acc <- mean(tree1_pred == test$is_canceled)
print(paste("accuracy = ", tree1_acc))
```

```
## [1] "accuracy =  0.807"
```

```
table(tree1_pred, test$is_canceled)
```

```
##
## tree1_pred    0    1
##          0 1235  303
##          1   83  379
```

Comparison and analysis

The best algorithm was kNN, followed by logistic regression. The decision tree algorithm performed worst. My hypothesis is that the decision tree becomes overly complex from being trained with so many attributes and "learns" patterns that aren't really relevant to the data. kNN seems to be a great, general-purpose algorithm, and makes sense for predicting hotel cancellations as the situation surrounding one cancellation would likely be similar to that of other cancellations, so it is reasonable to simply cluster cancellations together and predict the value of each test point based on similarity to the training data. Logistic regression works surprisingly well, probably for a similar reason to kNN, but because the separation between the various canceled/non-canceled bookings lends itself to being "cut" by a logistic equation.