Midterm

Akshaya Ravichandran

```
head(hotel_door)
```

```
head(hotel_elevator)
```

```
head(hotel_frontdesk)
```

Where are people going from and to?

```
table(hotel_elevator$from, hotel_elevator$to)
```

Insights:

People have used the elevator to go from floor 1 to floor 18 followed 8 and vice versa for almost double the number of times compared to other floors. On observing the from and to columns in hotel_elevator we can see that the guests have visited only the ground floor and the floor in which their room is situated.

Recommendation:

More info about other/all floors visited could help us find out if elevators were used more during the time of false openings and get more insights about the security flaws.

What cars are they using to go from?

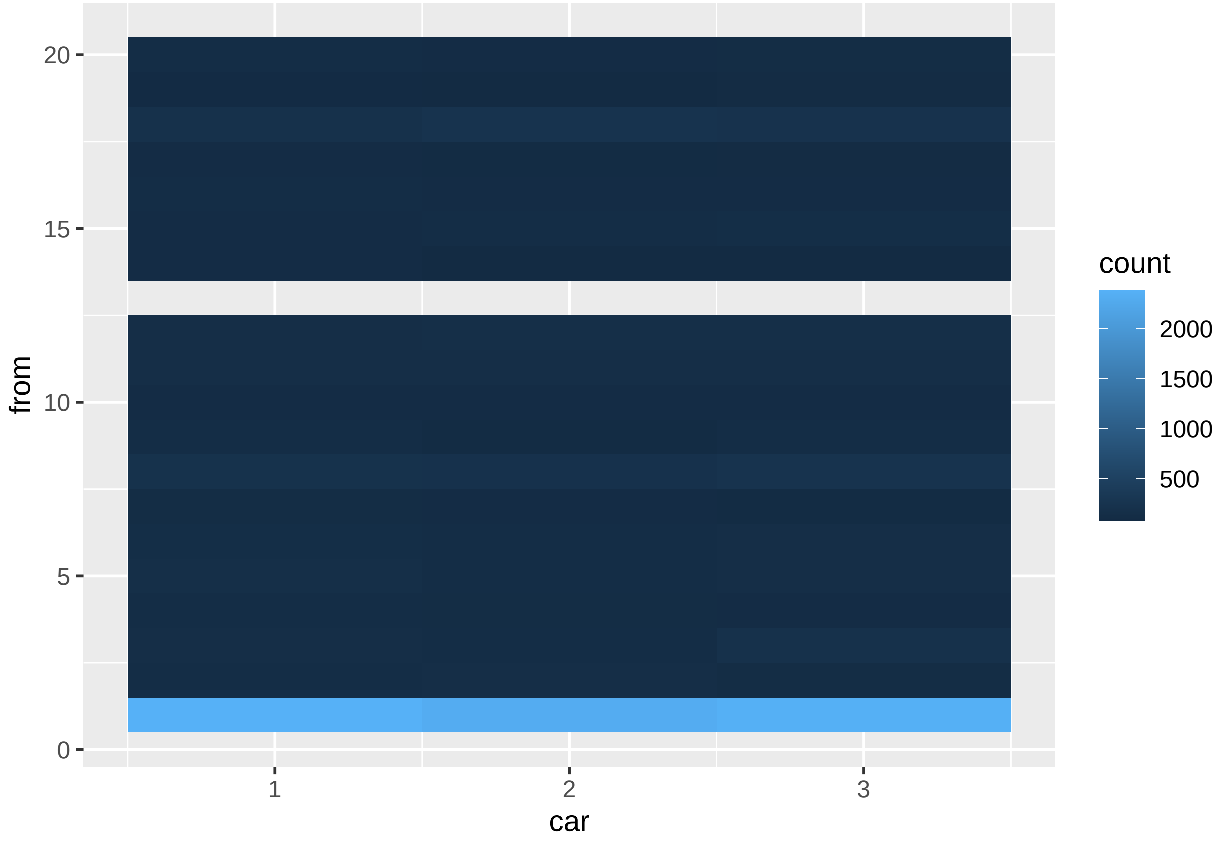
```
#There are 20 floors in total
table(hotel_elevator$from, hotel_elevator$car)
```

Most guests seem to use car 1, which could be either because car 1 is the closest to most rooms or because guests are more familiar with where it's located.

HEATMAP OF CAR-FLOOR RELATIONSHIP:

```
library(ggplot2)

count = c(table(hotel_elevator$from, hotel_elevator$car))
car = rep(1:3, each=19)
from = rep(c(1:12,14:20), times=3)
dat_counts = data.frame(count, car, from)
```

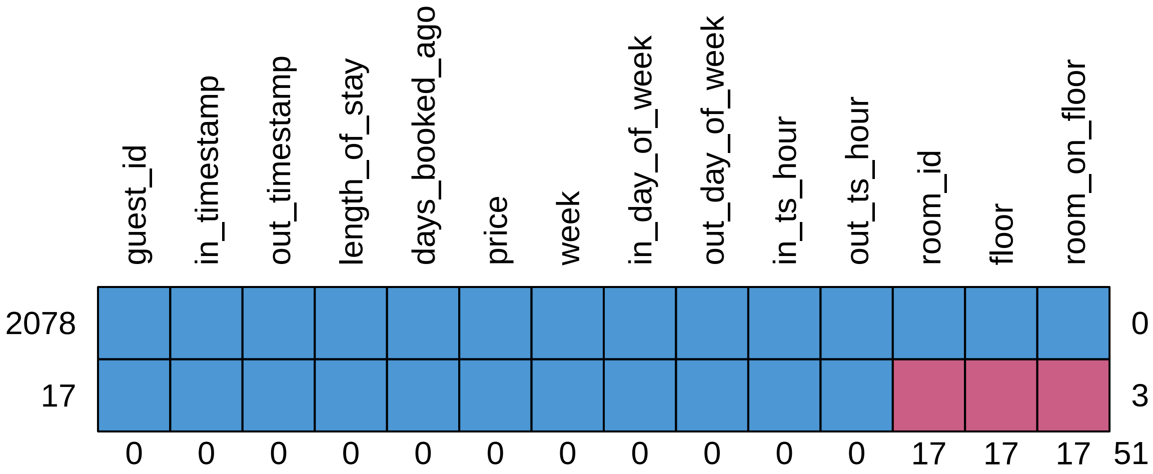
```
g = ggplot(dat_counts, aes(y=from, x=car, fill=count)) +
     geom_tile()

plot(g)
```



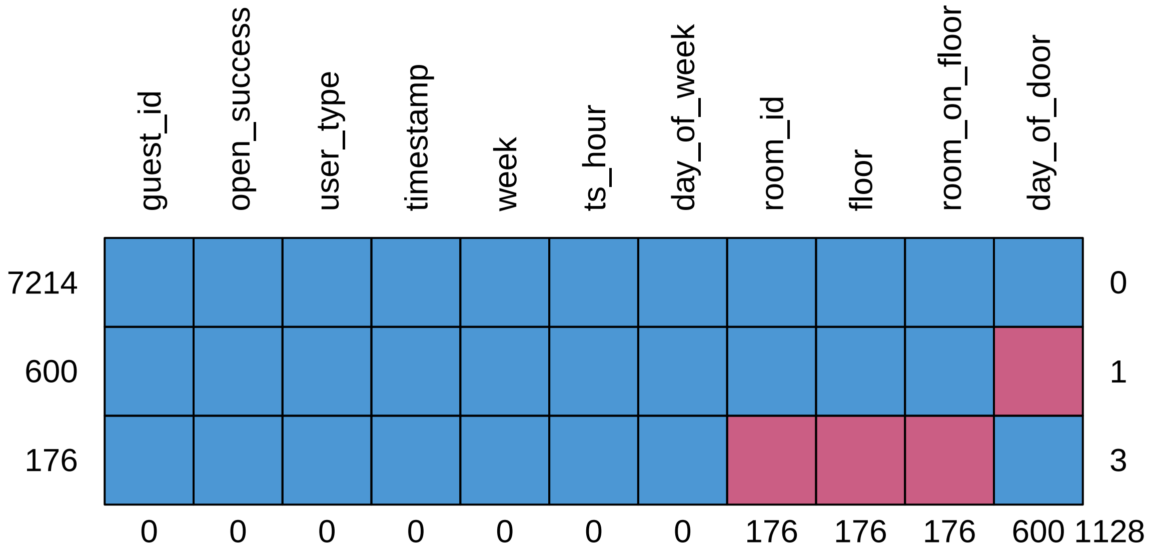
Floor 1 (ground floor) is the most visited floor i.e. it could be the reception so all guests arrive here initially and need to get back to the reception everytime they wish to leave the hotel. We can observe that the greyish white area in the middle of the heatmap is because there is no floor 13, which could be because the number 13 is considered unlucky (superstitious belief) in many cultures.

ANALYSING MISSING DATA:

```
md.pattern(hotel_frontdesk, rotate.names = TRUE)
```

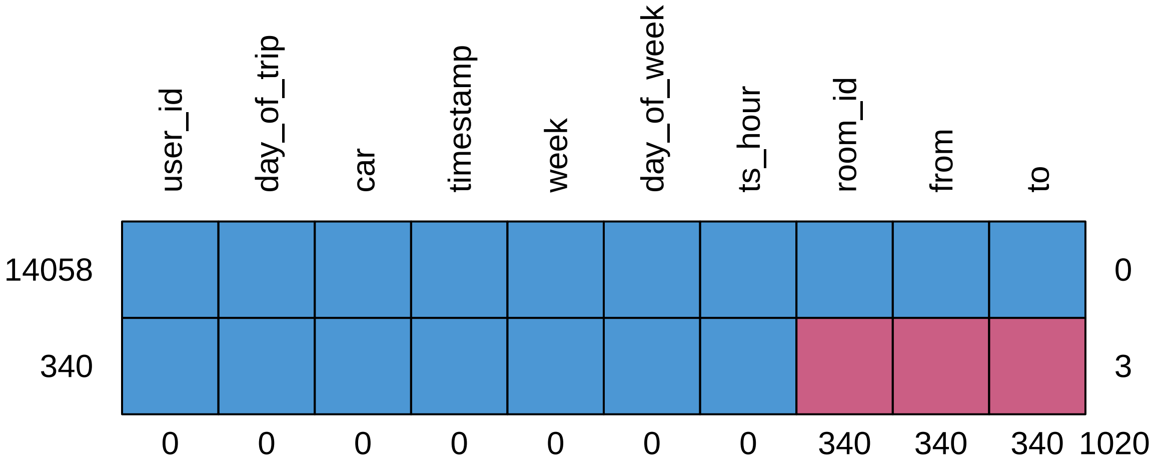


```
md.pattern(hotel_door,rotate.names = TRUE)
```



floor and room_on_floor are dependent on the room_id which itself is missing. Therefore, room_id is MCAR while floor & room_on_floor are missing at random. day_of_door is MAR since, it'd dependent on the room_id (missing).

```
md.pattern(hotel_elevator, rotate.names = TRUE)
```



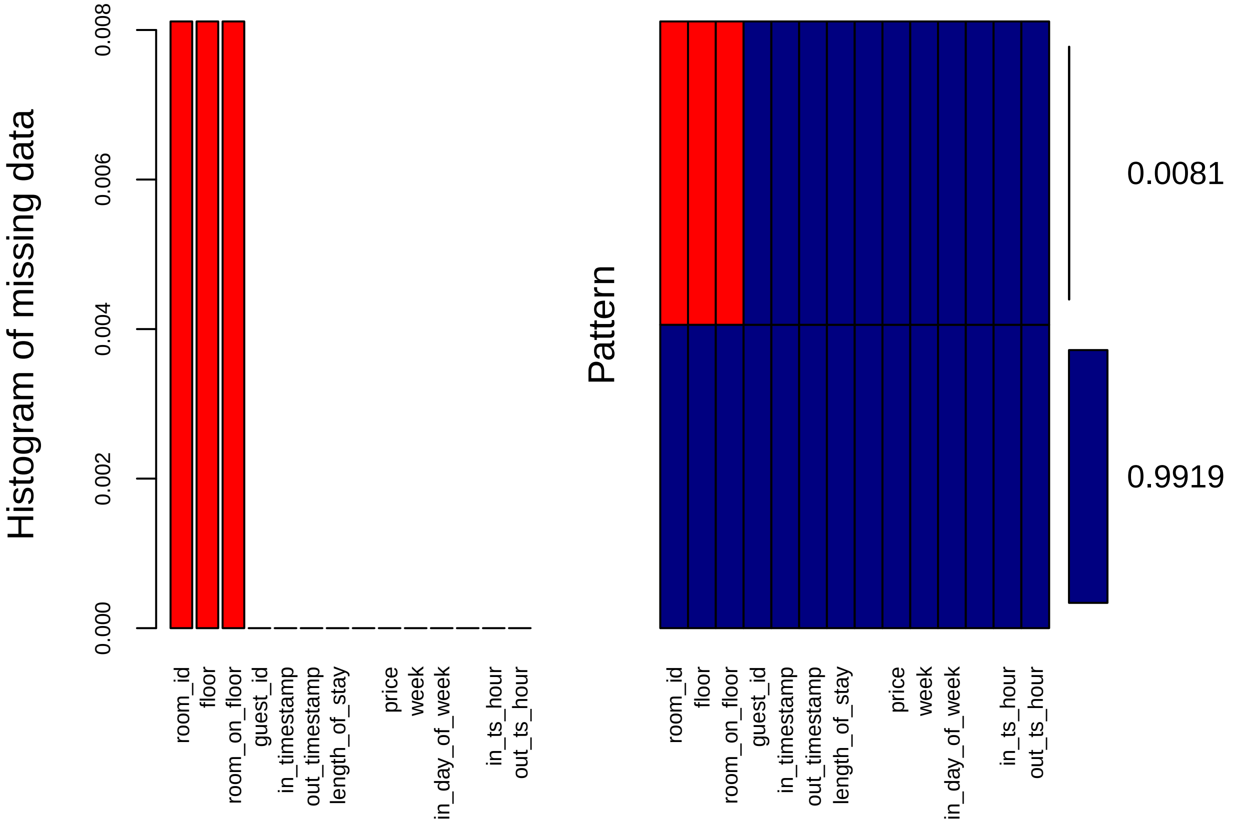
from and to are missing at random as they are dependent on the room_id which itself is missing.

```r
df <- tibble(x=hotel_door$day_of_door)

dplyr::count(df, x, sort = TRUE)
```

```
## # A tibble: 8 x 2
##       x     n
##   <int> <int>
## 1     1  2421
## 2     0  2160
## 3     2  1434
## 4     3   937
## 5    NA   600
## 6     4   310
## 7     5   107
## 8     6    21
```

Most guests opened their rooms either on the first day or immediately which is the usual expected pattern.

```r
#below shows that 99% percentage of data is available(not missing)
aggr_plot <- aggr(hotel_frontdesk, col=c('navyblue','red'),
numbers=TRUE, sortVars=TRUE, labels=names(hotel_frontdesk), cex.axis=.7, gap=3,
ylab=c("Histogram of missing data","Pattern"))
```



```
#dropping rows from hotel front desk where roomid =NA

 hotel_frontdesk_roomid_dropped<- drop_na(hotel_frontdesk)
#hotel_frontdesk_roomid_dropped<-hotel_frontdesk[Reduce("&",data.frame(!sapply(hotel_frontdesk[5],is.na

hotel_frontdesk <- hotel_frontdesk_roomid_dropped
```

About 99% of the data is available only 0.8% / 1% (approx) of entire data is missing because of these 3 variables. Since, it does not make sense to impute room_id's we have dropped all rows with missing room_id's. We also need to note that the same 17 guests, room_id's are missing from all 3 tables.

```
#removing rows with missing room_id's in hotel_door
#hotel_door <- drop_na(hotel_door)
#sapply(hotel_door, function(x) sum(is.na (x)))
```

```
defaultW <- getOption("warn")
options(warn = -1)
hotel_elevator = read.csv("hotel_elevator.csv")
hotel_frontdesk = read.csv("hotel_frontdesk.csv")
hotel_door = read.csv("hotel_door.csv")
if(hotel_door$ts_hour == hotel_frontdesk$in_ts_hour)
  {
  hotel_door$day_of_door=0
  }
sapply(hotel_door, function(x) sum(is.na (x)))
```

```
##      guest_id   day_of_door       room_id         floor room_on_floor
##             0             0           176           176           176
##  open_success     user_type     timestamp          week       ts_hour
##             0             0             0             0             0
##   day_of_week
##             0
```

```
hotel_door <- drop_na(hotel_door)
sapply(hotel_door, function(x) sum(is.na (x)))
```

```
##      guest_id   day_of_door       room_id         floor room_on_floor
##             0             0             0             0             0
##  open_success     user_type     timestamp          week       ts_hour
##             0             0             0             0             0
##   day_of_week
##             0
```

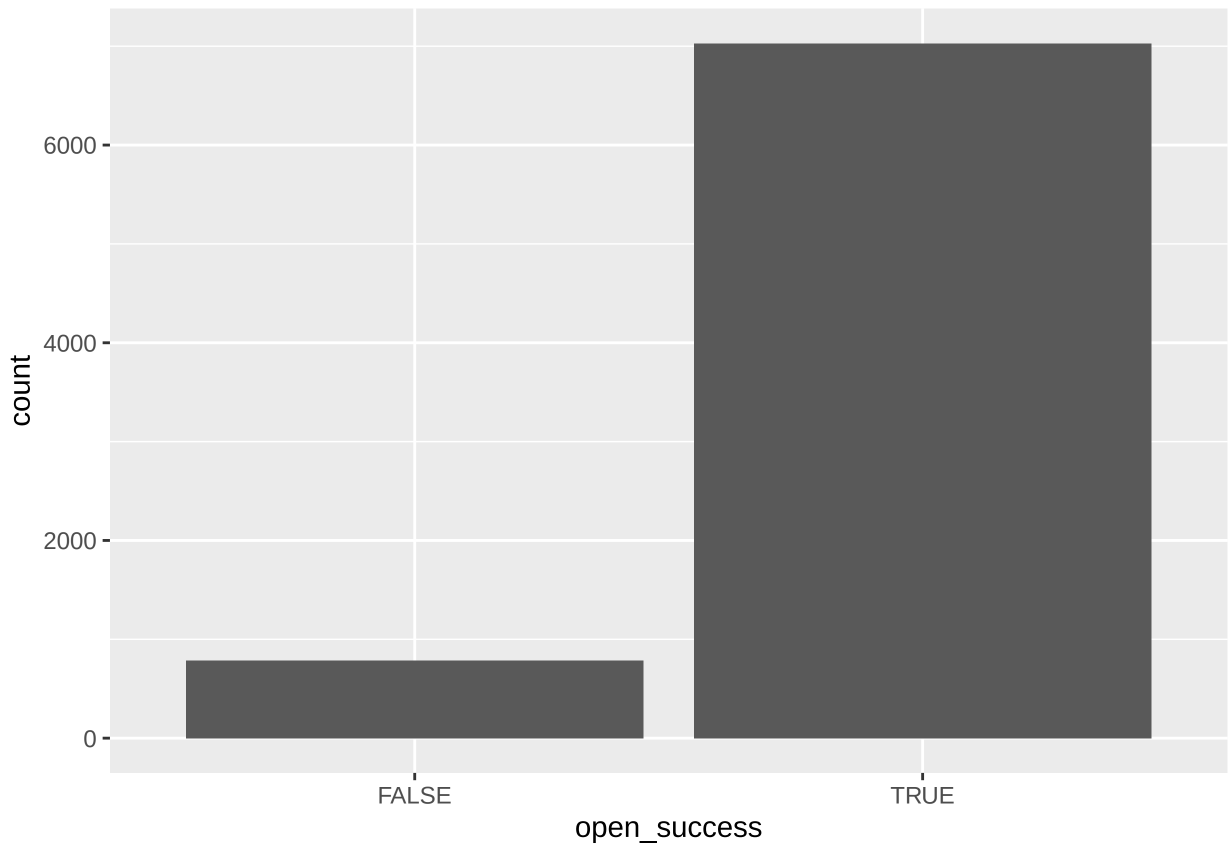
```
options(warn = defaultW)
hotel_door <- drop_na(hotel_door)
```

```
#frontdesk_door_merged
frontdesk_door_merged <- merge(hotel_frontdesk, hotel_door, by.x = "guest_id",
             by.y = "guest_id", all.x = TRUE, all.y = FALSE)
```

```
#frontdesk_door_elevator
frontdesk_door_elevator <- merge(frontdesk_door_merged,hotel_elevator, by.x = "guest_id",
             by.y = "user_id", all.x = TRUE, all.y = FALSE)

#### we have combined datasets using joins- first we have joined the frontdesk and elevator dataset usi
```

Number of false attempts made to open the hotel door:

```
plot1 <-ggplot(hotel_door, aes(x = open_success)) +
  geom_bar()
plot1
```



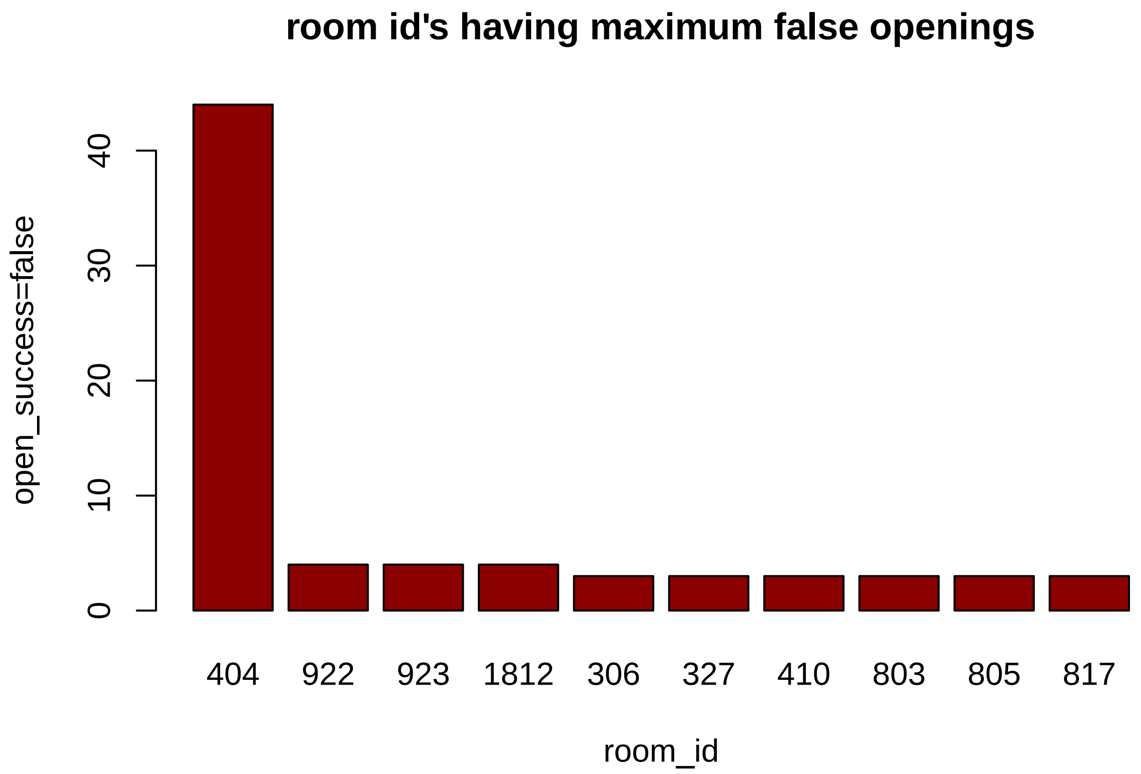
```
df <- tibble(x= hotel_door$open_success)
dplyr::count(df, x, sort = TRUE)
```

```
## # A tibble: 2 x 2
##   x         n
##   <lgl> <int>
## 1 TRUE   7029
## 2 FALSE   785
```

We can observe that approximately 785 out 7990 attempts made to open the door have been unsuccessful.

Rooms that have maximum number of false login attempts

```
sub1=filter(hotel_door, open_success=='FALSE')
df2 <- tibble(x= sub1$open_success,y=sub1$room_id)
new_df<-dplyr::count(df2, x, y, sort = TRUE)
sub2<-head(new_df, 10)

barplot(sub2$n,
main = "room id's having maximum false openings",
xlab = "room_id",
ylab = "open_success=false",
names.arg = sub2$y,
col = "darkred"
)
```



Therefore, we can conclude that in room 404 has had the maximum number of false login attempts.

Guests that had done major number of false attempts to open room 404 is shown below

```
sub3=filter(hotel_door, open_success=='FALSE')

df2 <- tibble(x= sub3$guest_id,y=sub3$room_id)

new_df<-dplyr::count(df2, x, y, sort = TRUE)
guests_with_max_false_logins<-head(new_df, 10)
guests_with_max_false_logins
```

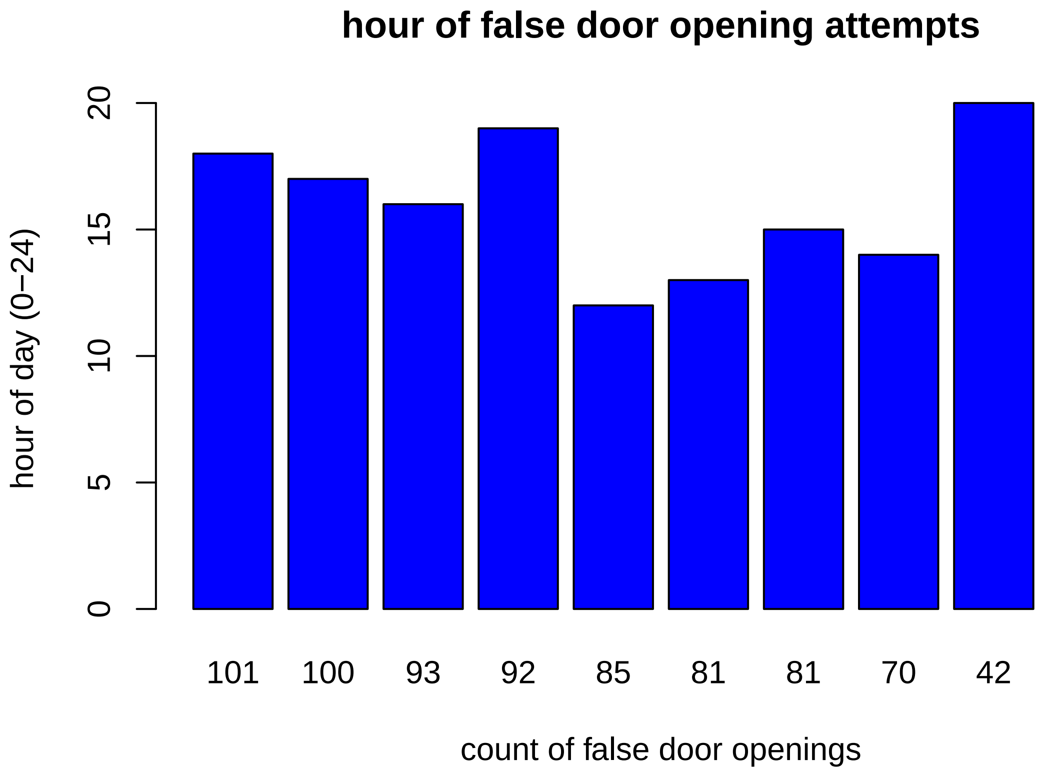
```
## # A tibble: 10 x 3
##        x     y     n
##    <int> <int> <int>
##  1  2022   404    30
##  2 10164   404    12
##  3  4090  1015     2
##  4  5056   923     2
##  5  5162  1507     2
##  6  7107   922     2
##  7  9070  1812     2
##  8  1015  1503     1
##  9  1018  1226     1
## 10  1024   101     1
```

```
t<-table(hotel_frontdesk$floor, hotel_frontdesk$room_on_floor)

#There are 18 floors -> ground floor has no rooms (must be the reception)
#There are 29 rooms on each floor
```

Therefore, guests with guest_id 2022,10164 had the most number of false login attempts into room no 404, indicative of probably making attempts to login using the wrong keycard or trying to break in.

Timings at which most false attempts at opening

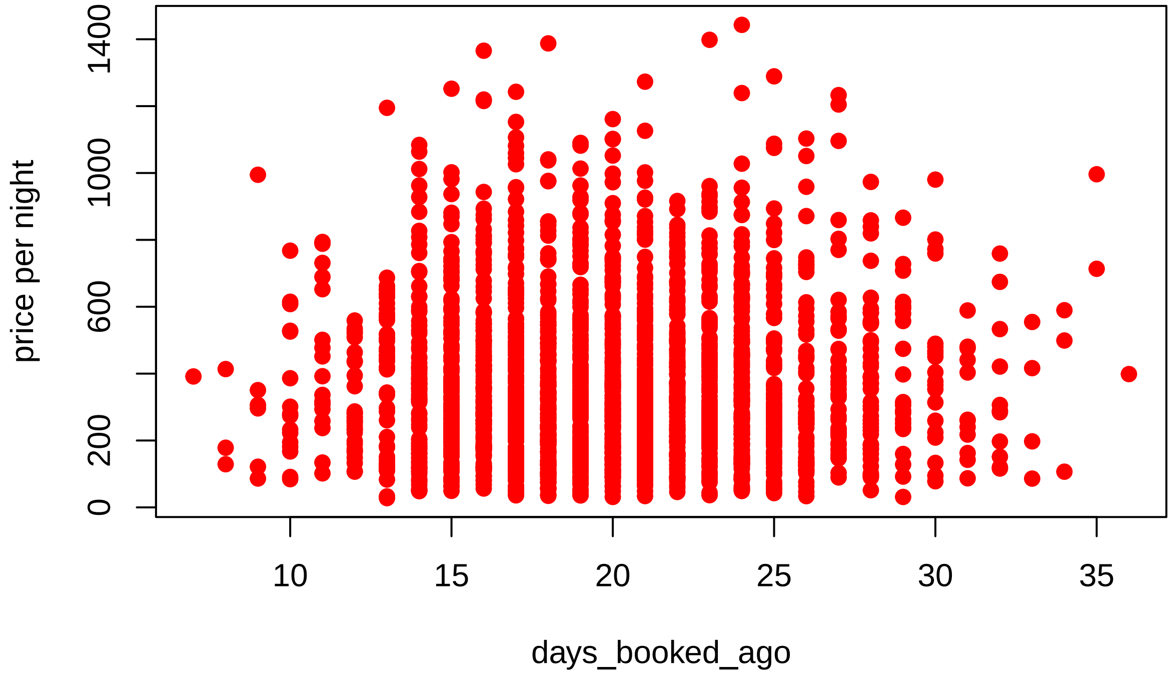
```
sub4=filter(hotel_door, open_success=='FALSE')
df2 <- tibble(x= sub4$ts_hour)

new_df<-dplyr::count(df2, x, sort = TRUE)
timings<-head(new_df, 10)

barplot(timings$x,
main = "hour of false door opening attempts",
xlab = "count of false door openings",
ylab = "hour of day (0-24)",
names.arg = timings$n,
col = "blue"
)
```



This means that most of the false attempts have occurred at around 6 pm,5pm,4pm in the evening, followed by 7pm at night and 12-1pm in the afternoon. The reason behind this could be that these are usually the prime times when guests are out of their rooms either for evening snacks/events or to have their lunch.

Relationship between room price/night and day before which the room was booked

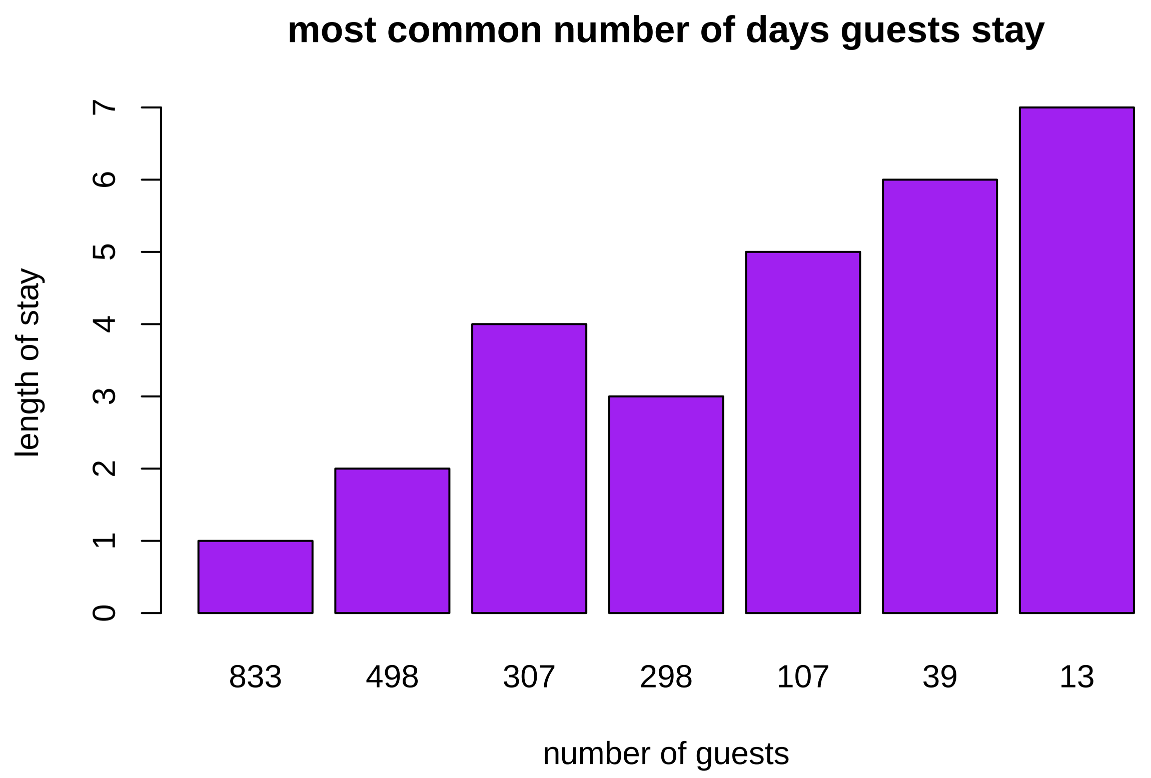
```r
##pattern between price per night and booking day time gap
price_per_night<-hotel_frontdesk$price/ hotel_frontdesk$length_of_stay
plot(x=hotel_frontdesk$days_booked_ago, y=price_per_night, pch = 19 ,
     col = "red", xlab = "days_booked_ago", ylab = "price per night")
```



## INSIGHTS: No particular pattern can be observed between the days booked ago and price i.e. irrespective of whether guests booked 10 days ago or 35 days ago, price remains quite independent of that. However, those guests who arrive on the weekends (Saturdays and Sundays) tend to pay more than those arriving on weekdays irrespective of days booked ago.

Commonly preferred trip duration by guests

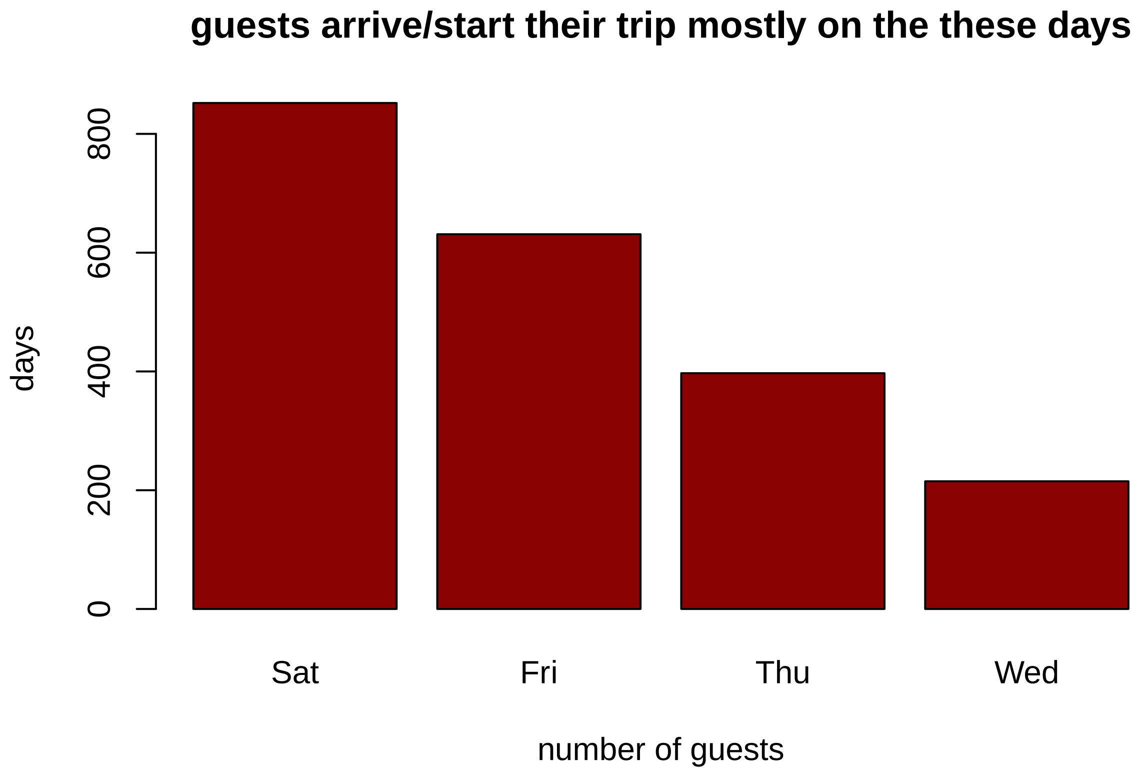
```
#most common number of days guests stay (length of stay)

df2 <- tibble(x= hotel_frontdesk$length_of_stay)
new_df<-dplyr::count(df2, x, sort = TRUE)

barplot(new_df$x,
main = "most common number of days guests stay",
xlab = "number of guests",
ylab = "length of stay",
names.arg = new_df$n,
col = "purple"
)
```



Therefore, we can conclude that almost 833 guests (approximately 40%) prefer to take a trip for 1 day followed by 2,4,3,5,6 and 7 days. Recommendation: This could indicate that most guests live nearby or that there aren't many places to visit in the location where the hotel is. If more information about where the guests come from was given, we could categorize the type of guests as local population or foreigners and who are frequent guests of the hotel.

Common day at which most guests arrive at the hotel

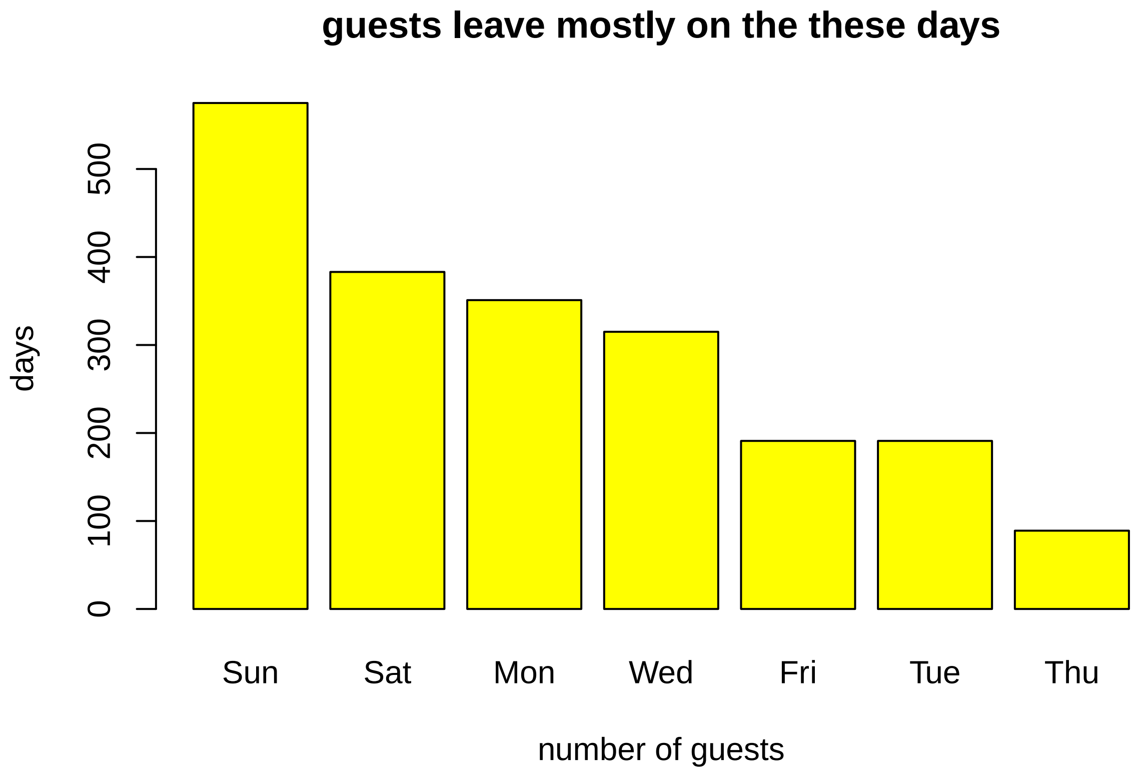
```
df2 <- tibble(x= hotel_frontdesk$in_day_of_week)
new_df<-dplyr::count(df2, x, sort = TRUE)

barplot(new_df$n,
main = "guests arrive/start their trip mostly on the these days",
xlab = "number of guests",
ylab = "days",
names.arg = new_df$x,
col = "darkred"
)
```



This means that most guests seem to arrive at the hotel on Saturdays and Fridays, probably because it's closer to the weekend and they can get to spend the entire weekend as part of their vacation.

Days at which most guests end their trip

```
df2 <- tibble(x= hotel_frontdesk$out_day_of_week)
new_df<-dplyr::count(df2, x, sort = TRUE)


barplot(new_df$n,
main = "guests leave mostly on the these days",
xlab = "number of guests",
ylab = "days",
names.arg = new_df$x,
col = "yellow"
)
```



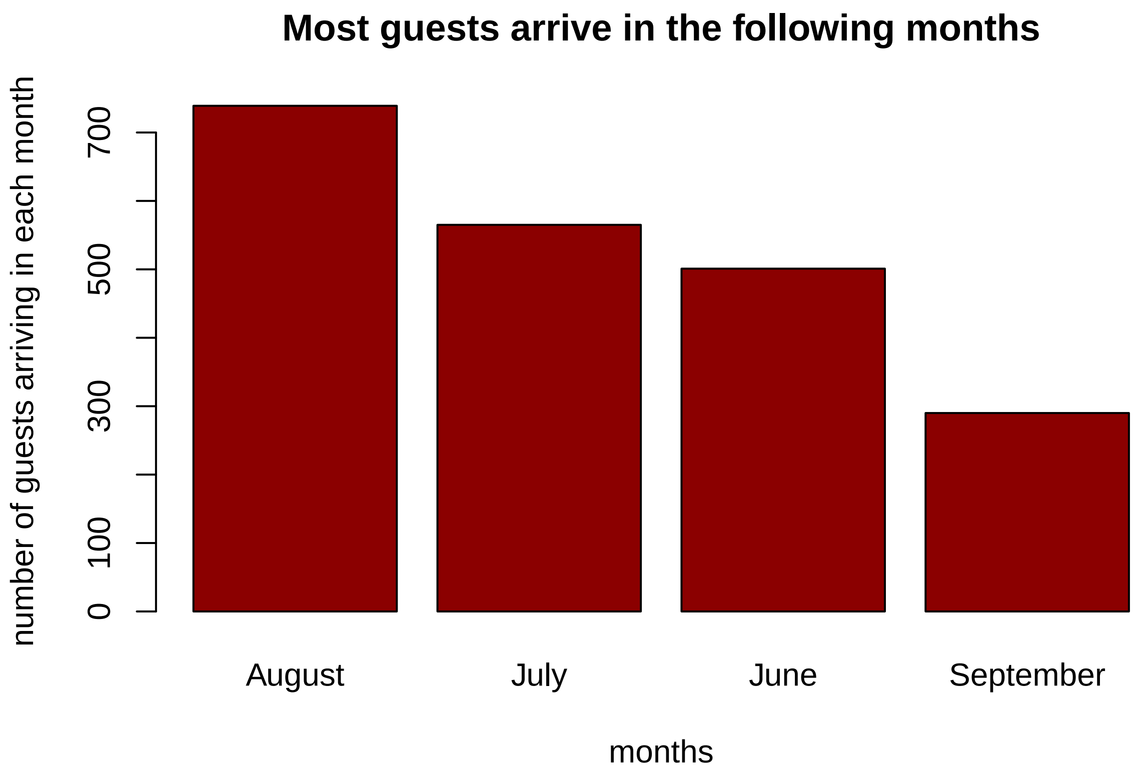
This implies that most guests arrive on saturday and leave on sunday (1 day trips are most common), followed by 2 day trips (Arrive on Friday and leave on Sunday). This therefore, confirms our previous findings are correct.

Months where the hotel has high demand by guests

```
date <- format(as.POSIXct(hotel_frontdesk$in_timestamp, format='%Y-%m-%d %H:%M:%S'),format='%Y-%m-%d')
date <- as.Date(date)
m<-months(date)
myDate = as.POSIXct(date)
numeric_month<-format(myDate,"%m")
df2 <- tibble(x= m)



new_df<-dplyr::count(df2, x, sort = TRUE)
new_df
```

```
## # A tibble: 4 x 2
##   x             n
##   <chr>     <int>
## 1 August      739
## 2 July        565
## 3 June        501
## 4 September   290
```

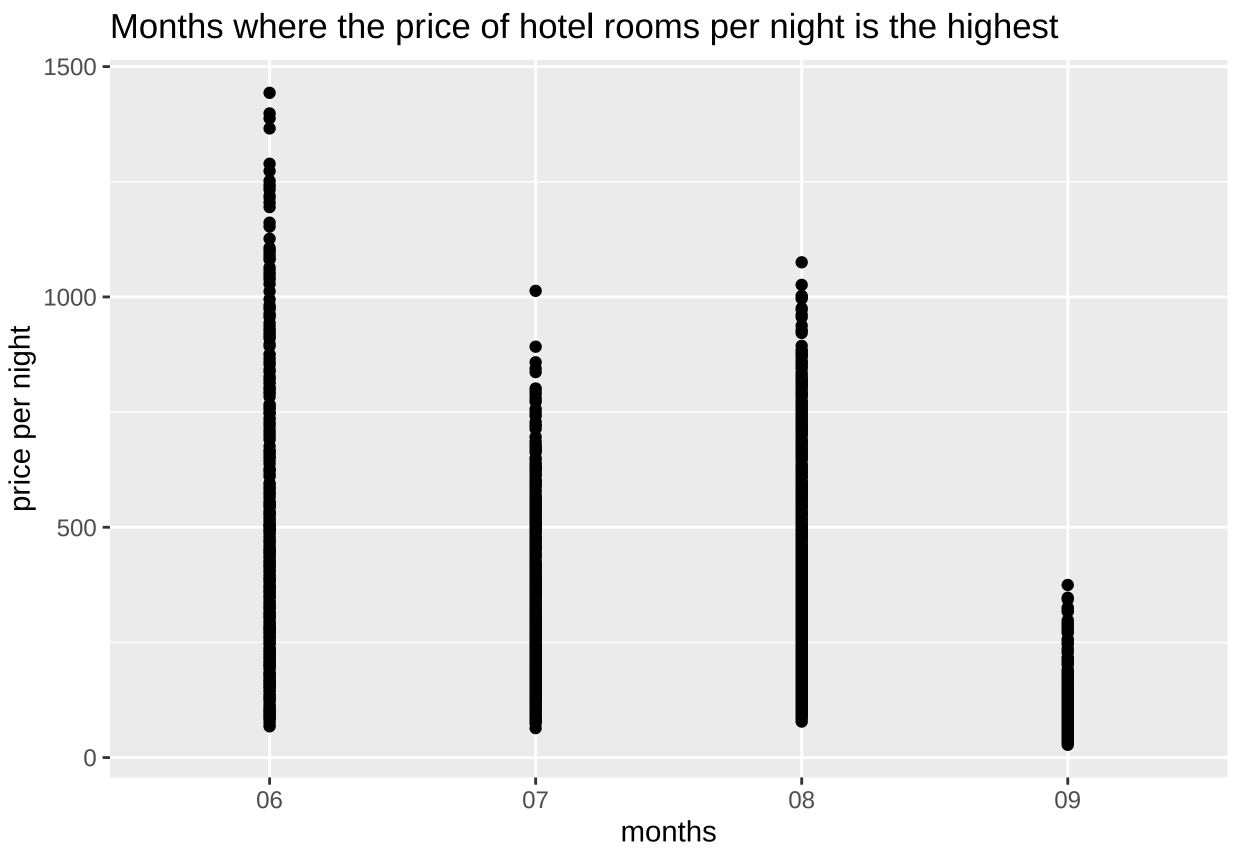
```
barplot(new_df$n,
main = "Most guests arrive in the following months",
xlab = "months",
ylab = "number of guests arriving in each month",
names.arg = new_df$x,
col = "darkred"
)
```



Therefore, we can observe that most guests prefer to travel in the August (733 guests) and September is the least preferred month. Recommendation: We have the guest data for the months of June, July, Aug and Sept. If we had the data for other months of the year, we could have a more accurate picture of which months the guests like to visit the hotel the most and make recommendations to the hotel accordingly to increase prices that time for increasing profits.

Relationship between price per night and booking month

```
#pattern between price per night and booking month
defaultW <- getOption("warn")
options(warn = -1)
df2 <- tibble(x= numeric_month, y= price_per_night)
new_df<-dplyr::count(df2, x, y, sort = TRUE)
p<-ggplot(data=new_df,mapping= aes(x=new_df$x,y=new_df$y))+ geom_point()
p+ggtitle("Months where the price of hotel rooms per night is the highest")+ xlab("months")+ylab("price
```

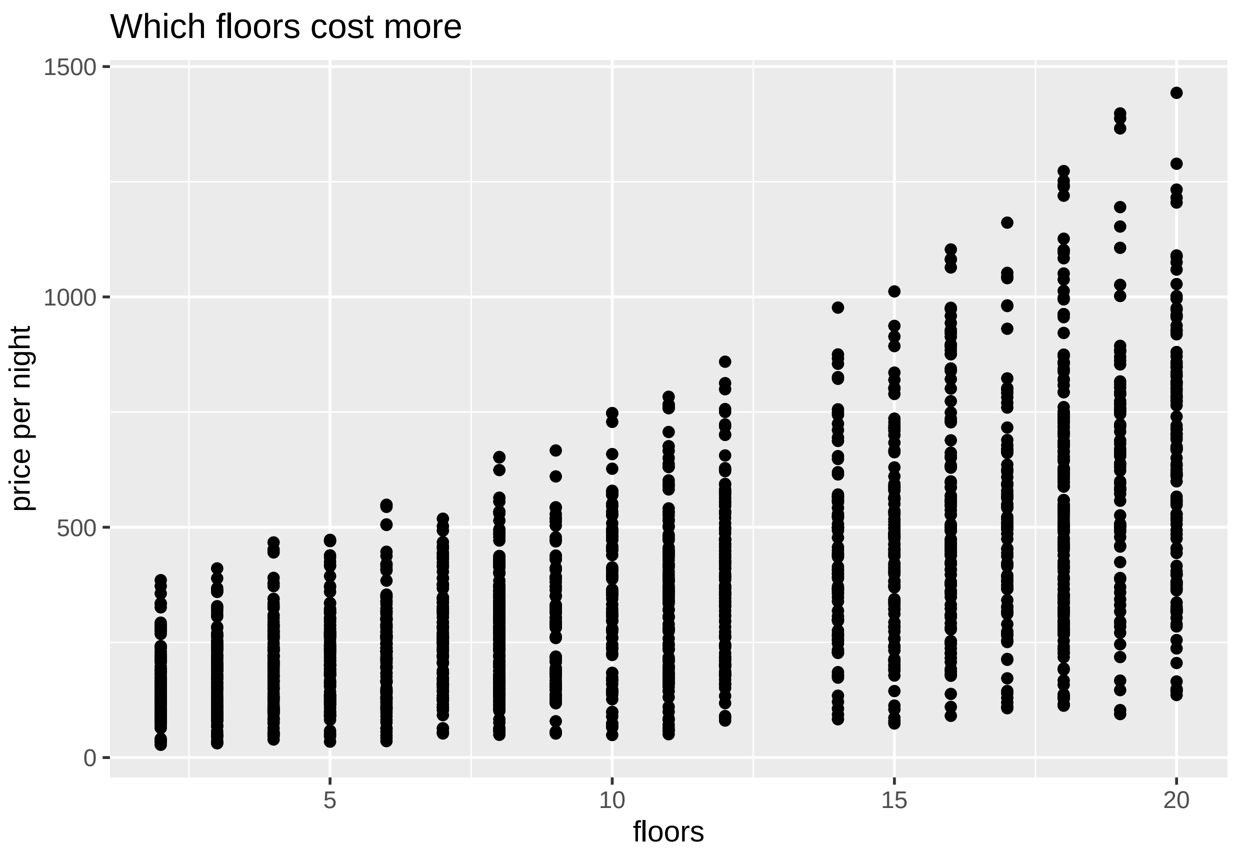


```
options(warn = defaultW)
```

Price per night seems to be highest in the month of June. Recommendation: However, since most people travel in August and July, it would make more sense to increase the price in the months of August and July where the demand/ number of guests who would book the hotel for stay would be more to improve overall profits.

Relationship between price and floor number

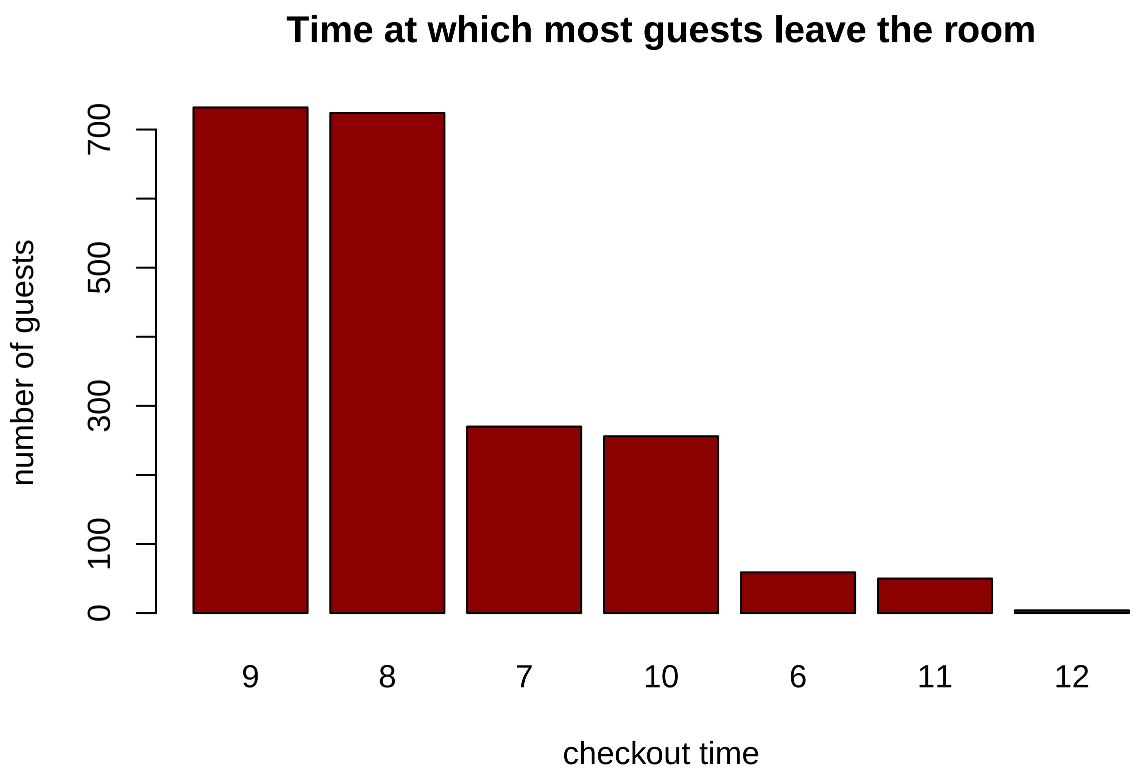
```
defaultW <- getOption("warn")
options(warn = -1)
df2 <- tibble(x= hotel_frontdesk$floor, y= price_per_night)
new_df<-dplyr::count(df2, x, y, sort = TRUE)
p<-ggplot(data=new_df,mapping= aes(x=new_df$x,y=new_df$y))+ geom_point()
p+ggtitle("Which floors cost more")+ xlab("floors")+ylab("price per night")
```



```r
options(warn = defaultW)
```

We can clearly observe a linear relationship between the floors and price per night i.e. the minimum starting rate for higher floors is higher than that of lower floors and the maximum range of prices for higher floors also seems to be higher.

```r
df2 <- tibble(x= hotel_frontdesk$out_ts_hour)
new_df<-dplyr::count(df2, x, sort = TRUE)


barplot(new_df$n,
main = "Time at which most guests leave the room",
xlab = "checkout time",
ylab = "number of guests",
names.arg = new_df$x,
col = "darkred"
)
```



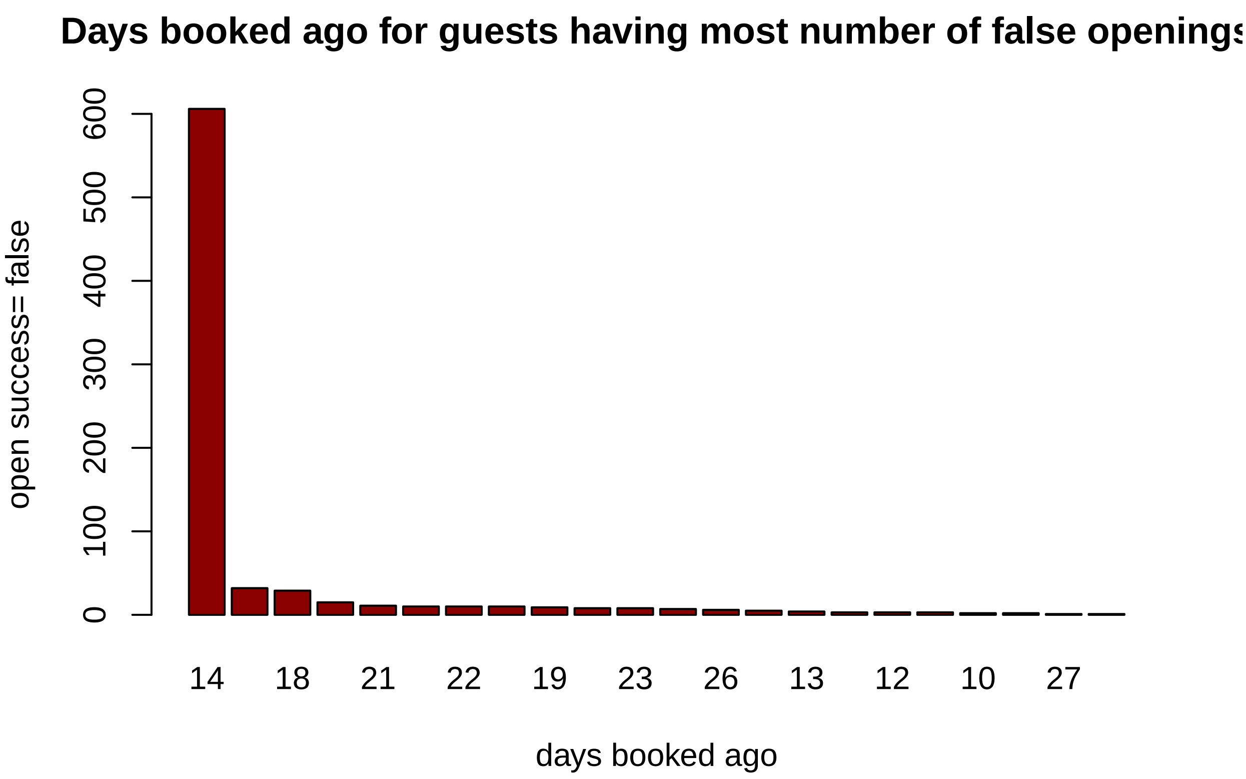
We can infer that most (728) guests leave their rooms at around 9am or 8am. This could mean that they usually get up for breakfast or to go out at around 9am or they usually leave their rooms on the last day of trip at 9am. Also, all guests leave their room by 12 pm irrespective of the day of their trip.

Recommendation:User type -> if specified as guest or staff -> if we can differentiate we could have more insights on who is responsible/ the person behind the false opening attempts.

QUESTION 1: Data has been combined from different tables for below analysis

Pattern between days booked ago and guests who had most false attempts

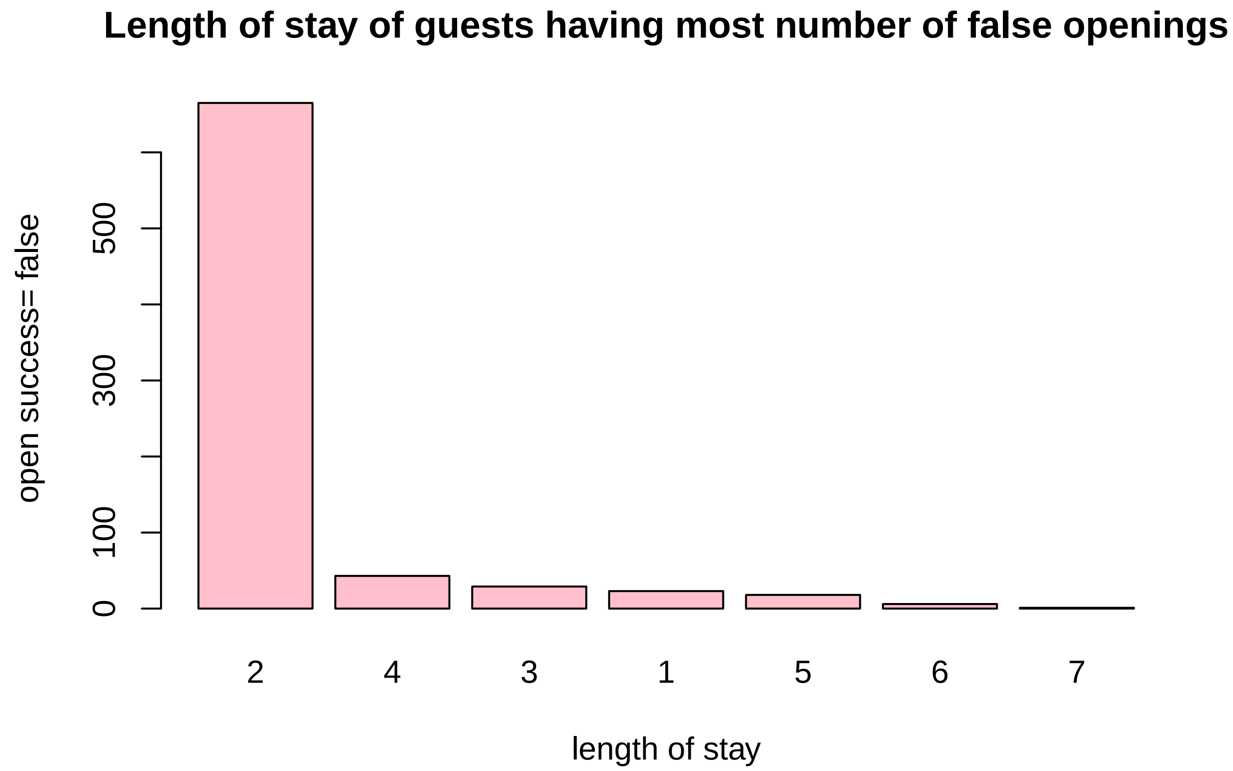
```
#how datasets have been combined -Q1
#hotel_frontdesk and hotel_door tables have been combined here for use
#days booked ago vs open_success= false
sub7= filter(frontdesk_door_merged, open_success=="FALSE")
df2 <- tibble(x= sub7$days_booked_ago, y=sub7$open_success)
new_df<-dplyr::count(df2, x,y, sort = TRUE)


barplot(new_df$n,
main = "Days booked ago for guests having most number of false openings",
xlab = "days booked ago",
ylab = "open success= false",
names.arg = new_df$x,
col = "darkred"
)
```



From the above barplot it is clear, that most of the false openings of the hotel doors have occurred in the case of guests who booked their rooms about 14 days before arrival at the hotel. This could indicate a pattern wherein people who came to steal booked their rooms before 14 days. (assumption)

Length of stay vs guests who had most false attempts

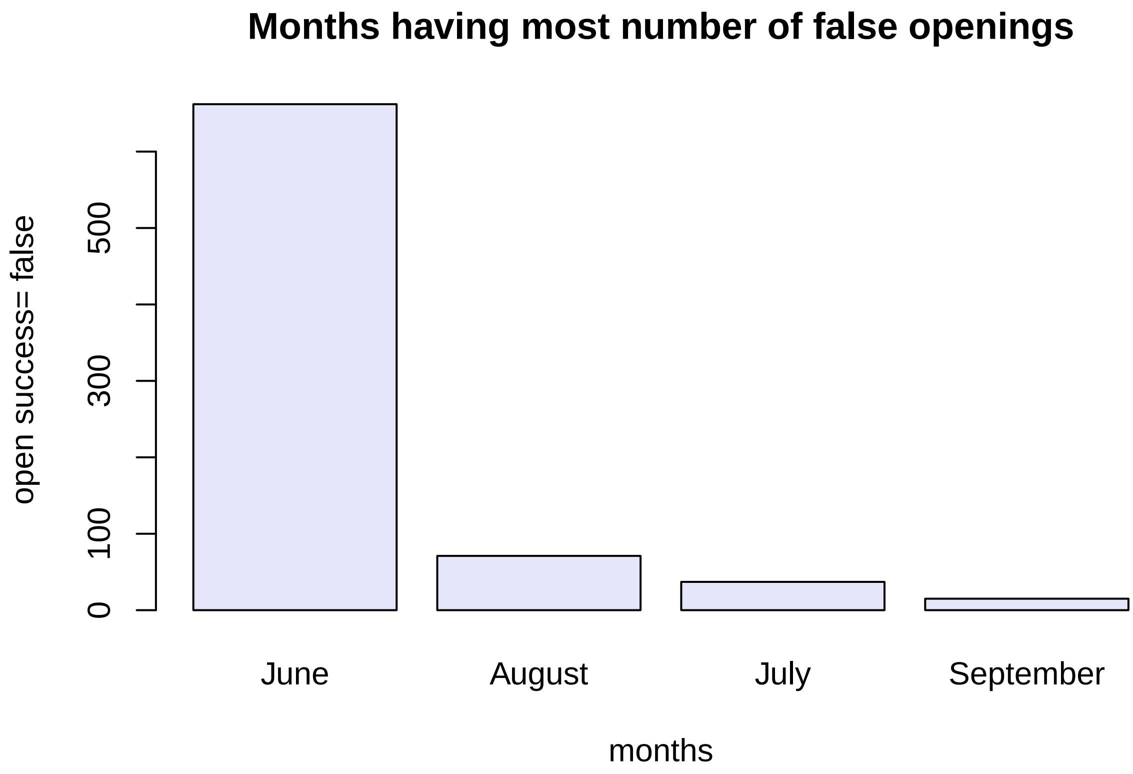
```
#Q1
#length of stay vs open_success = false
sub7= filter(frontdesk_door_merged, open_success=="FALSE")
df2 <- tibble(x= sub7$length_of_stay, y=sub7$open_success)
new_df<-dplyr::count(df2, x,y, sort = TRUE)


barplot(new_df$n,
main = "Length of stay of guests having most number of false openings",
xlab = "length of stay",
ylab = "open success= false",
names.arg = new_df$x,
col = "pink"
)
```



We can note that most guests who have failed to open the door correctly on the first attempt have a duration of stay of 2 days.This could be indicative of a pattern wherein most guests who plan on stealing/thiefs plan 2 day trips. This also indicates that guests who stay for a duration more than 2 days usually do not have any that many incidents of false openings in comparison. Also a clear linear decrease in the number of false openings can be observed as the length of stay of guests increases. This could also mean that the longer they stay, less likely they are to they are to fail to open the door (i.e. used to it).

Months where most false login attempts occurred

```
#month vs open_success= false

date <- format(as.POSIXct(sub7$in_timestamp, format='%Y-%m-%d %H:%M:%S'),format='%Y-%m-%d')
date <- as.Date(date)
m<-months(date)
myDate = as.POSIXct(date)
numeric_month<-format(myDate,"%m")
df2 <- tibble(x= m)

sub7= filter(frontdesk_door_merged, open_success=="FALSE")
df2 <- tibble(x= m , y=sub7$open_success)
new_df<-dplyr::count(df2, x,y, sort = TRUE)

barplot(new_df$n,
main = "Months having most number of false openings",
xlab = "months",
ylab = "open success= false",
names.arg = new_df$x,
col = "lavender"
)
```



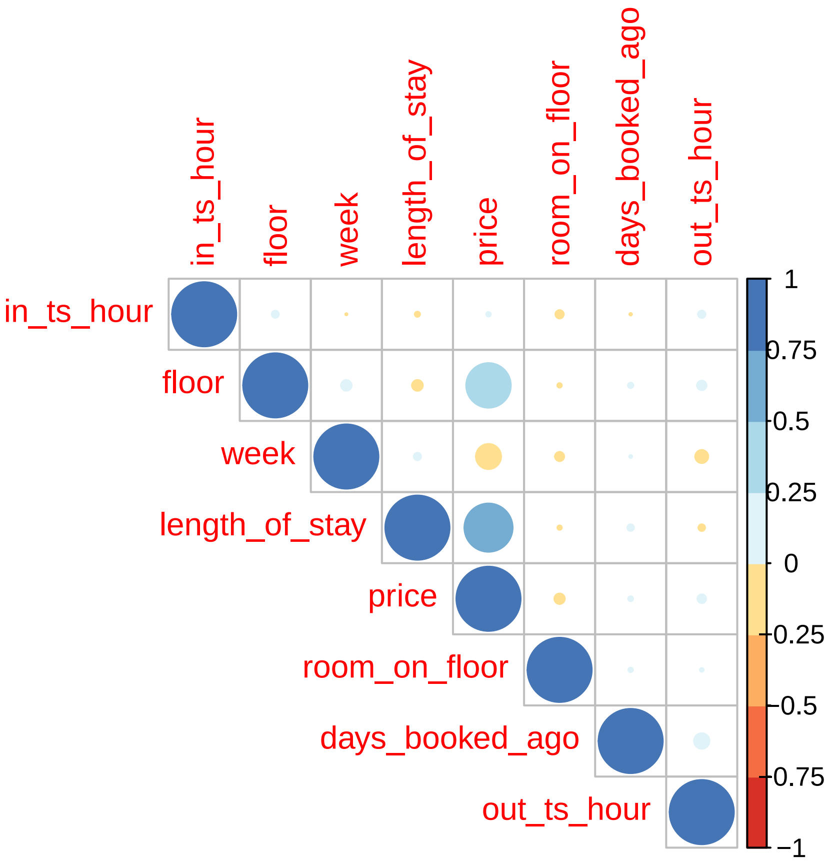
We can infer that false login attempts/possibility of theft was highest around 662 in June.

Correlation between different variables in hotel_frontdesk

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(RColorBrewer)
library(pheatmap)

M <-cor(hotel_frontdesk_roomid_dropped[,-c(1,2,3,5,11,12)])
corrplot(M, type="upper", order="hclust",
         col=brewer.pal(n=8, name="RdYlBu"))
```



Imputing hotel_frontdesk (for experimentation to see how it affects our analysis)

```
hotel_frontdesk <- read.csv("hotel_frontdesk.csv")

#IMPUTATION OF ROOM IDS
hotel_frontdesk_collinearity_removed <- subset(hotel_frontdesk, select = -c(6,7) )

tempData <- mice(hotel_frontdesk_collinearity_removed,m=5,maxit=50,meth='pmm',seed=500)
```

```
## Warning: Number of logged events: 5
```

```
summary(tempData)
#above imputes room id using pmm

tempData$imp$room_id
tempData$meth
completedData <- complete(tempData,1)

sapply(completedData, function(x) sum(is.na (x)))

DF1 <- cbind(completedData, hotel_frontdesk[!names(hotel_frontdesk) %in% names(completedData)])
```

```
DF2 <- DF1[, c(1,2,3,4,5,13,14,6,7,8,9,10,11,12)]

#DF2 IS THE HOTEL_FRONT DESK ROOM_ID IMPUTED DATAFRAME BUT FLOOR AND ROOM_ON_FLOOR ARE MISSING
```

```
# Floor and room_on_floor depend on room_id, so they have been imputed based on the room_id
#same code snippet has been used for all 17 missing room_id's (not shown here due to keep the report sh

bar <- subset(DF2, guest_id ==1131)
bar$floor =15
bar$room_on_floor =21
DF2[match(bar$guest_id, DF2$guest_id), ] <- bar
```

```
##DF2 is final imputed matrix of hotel_frontdesk
hotel_frontdesk_imputed<- DF2
 #final imputed hotel front desk table
```

QUESTION 3:

Lasso regression to predict hotel prices using imputed data

```
#Describe a model to predict hotel prices
#define response variable
y <- hotel_frontdesk_imputed$price
f <- as.formula(y ~ floor+ I(log(length_of_stay)^0.5)+week+floor:length_of_stay)
#define matrix of predictor variables
x <- model.matrix(f, hotel_frontdesk_imputed)
#finding best lambda using k fold cv
cv_model <- cv.glmnet(x, y, alpha = 1)
best_lambda <- cv_model$lambda.min
best_lambda
#plot(cv_model)
#from this we can infer with about 95% confidence that the best lambda uses only about 2-3 variables
#plot(cv_model$glmnet.fit, "lambda", label=TRUE)
best_model <- glmnet(x, y, alpha = 1, lambda = best_lambda)
coef(best_model)
y_predicted <- predict(best_model, s = best_lambda, newx = x)
sst <- sum((y - mean(y))^2)
sse <- sum((y_predicted - y)^2)
rsq <- 1 - sse/sst
rsq

new = matrix(c(17,6,1,17,6), nrow=1, ncol=5)
predict(best_model, s = best_lambda, newx = new)
```

```
coef(best_model)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                                      s0
## (Intercept)                   262.31406
## (Intercept)                         .
## floor                       13.98573
## I(log(length_of_stay)^0.5) 186.85206
## week                       -26.39822
## floor:length_of_stay        17.89681
```

```
rsq
```

```
## [1] 0.6314391
```

Linear regression to predict hotel prices

```
#Describe a model to predict hotel prices
#MODEL USING ONE CONTINUOUS, CATEGORICAL, POLYNOMIAL AND INTERACTION TERM.
hotel_frontdesk_original <- read.csv("hotel_frontdesk.csv")
hotel_frontdesk_original= na.omit(hotel_frontdesk_original)
hotel_price_pred_model<- lm(price~ floor+ I(log(length_of_stay)^0.5)+in_day_of_week+floor:length_of_sta
#summary(hotel_price_pred_model)
summary(hotel_price_pred_model)$r.squared
```

```
## [1] 0.6022417
```

```
summary(hotel_price_pred_model)$df
```

```
## [1]    7 2071    7
```

```
log_odds = predict(hotel_price_pred_model)
predicted = exp(log_odds)/(1+exp(log_odds))
hotel_frontdesk_original$predicted = predicted


#MODEL WITH HIGHEST R SQUARED
hotel_price_pred_model<- lm(price~ floor+ I(log(length_of_stay)^0.5)+week+floor:length_of_stay, data= h
summary(hotel_price_pred_model)$r.squared
```

```
## [1] 0.6314705
```

```
summary(hotel_price_pred_model)$df
```

```
## [1]    5 2073    5
```

Price is highly correlated/dependent on the length of stay (since, more no of days, more is the cost), floor (higher floors have more cost as observed) and week (some weeks have higher prices for rooms than others)

QUESTION 4:

Lasso regression to predict length_of_stay using imputed data

```r
#Describe a model to predict length of stay
#define response variable
y <- hotel_frontdesk_imputed$length_of_stay
#define matrix of predictor variables
x <- data.matrix(hotel_frontdesk_imputed[, c('price','week','out_day_of_week','days_booked_ago')])
#finding best lambda using k fold cv
cv_model <- cv.glmnet(x, y, alpha = 1)
best_lambda <- cv_model$lambda.min
best_lambda
#plot(cv_model)
#from this we can infer with about 95% confidence that the best lambda uses only about 2-3 variables
#plot(cv_model$glmnet.fit, "lambda", label=TRUE)
best_model <- glmnet(x, y, alpha = 1, lambda = best_lambda)
coef(best_model)
y_predicted <- predict(best_model, s = best_lambda, newx = x)
sst <- sum((y - mean(y))^2)
sse <- sum((y_predicted - y)^2)
rsq <- 1 - sse/sst
rsq
```

```r
coef(best_model)
```

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                            s0
## (Intercept)     -0.0033737922
## price            0.0009459772
## week             0.0240262938
## out_day_of_week  0.3408355567
## days_booked_ago  0.0026766030
```

```r
rsq
```

```
## [1] 0.5122783
```

Linear regression to predict length of stay

```r
#Describe a model to predict length of stay
hotel_staylength_pred_model<- glm(length_of_stay ~  floor+ I(log(price)^0.5)
                               + out_day_of_week + floor:price, data= hotel_frontdesk)
my_sum =summary(hotel_staylength_pred_model)
1 - (my_sum$deviance / my_sum$null.deviance)
```

```
## [1] 0.7696785
```

```r
#df
my_sum$df.null- my_sum$df.residual
```

```
## [1] 9
```

```
log_odds = predict(hotel_staylength_pred_model)
predicted = exp(log_odds)/(1+exp(log_odds))
hotel_frontdesk_original$predicted = predicted
```

Length of stay seems to be highly correlated on out_day_of_week and price more than in_day_of_week.

Since we have the check in date and timestamp of each guest in hotel_frontdesk and the date and timestamp at which the hotel door was opened, we can match both to impute the day_of_door in hotel_door column.Now only those 176 records having room_id's missing (17 unique guests) are dropped since it does not make sense to impute room_id's.