Hotel Operations and Customer Behaviour through Data Analysis

Project #27 (SAS-READING-2)

Mentor:
Yan Yu
Mentees:
Arwen Mao
Ellen He
Julia Wrona
Rachel Tania

Learning Data Analysis in R

I. Visualization

II. Modeling: Regression

III. Model Selection: LASSO, Random Forest

IV. Missing Data, Interpolation and Imputation

1. Project Intro & Data Merging
2. Stories We Found
3. Data Preprocessing
4. LASSO Model: Hotel Price
5. Random Forest: Length of Stay

1. **Project Intro & Data Merging**
2. **Stories We Found**
3. **Data Preprocessing**
4. **LASSO Model: Hotel Price**
5. **Random Forest: Length of Stay**

Project Introduction: 3 Hotel Datasets

hotel_door dataset

```
> head(hotel_door)
```

	guest_id	day_of_door	room_id	floor	room_on_floor	open_success	user_type	timestamp	week	ts_hour	day_of_week
1	1001	0	1403	14	3	TRUE	guest	2018-06-13 17:43:23	1	17	Wed
2	1002	0	1413	14	13	TRUE	guest	2018-06-16 16:21:09	1	16	Sat
3	1003	0	1706	17	6	TRUE	guest	2018-06-14 17:23:38	1	17	Thu
4	1003	1	1706	17	6	TRUE	guest	2018-06-15 12:21:26	1	12	Fri
5	1003	1	1706	17	6	TRUE	guest	2018-06-15 18:18:42	1	18	Fri
6	1003	2	1706	17	6	TRUE	guest	2018-06-16 22:27:01	1	22	Sat

hotel_front_desk dataset

```
> head(hotel_front_desk)
```

	guest_id	in_timestamp	out_timestamp	length_of_stay	room_id	floor	room_on_floor	days_booked_ago	price	week	in_day_of_week	out_day_of_week	in_ts_hour	out_ts_hour
1	1001	2018-06-13 17:34:15	2018-06-14 08:05:11	1	1403	14	3	13	497.99	1	Wed	Thu	17	8
2	1002	2018-06-16 16:12:02	2018-06-17 09:18:35	1	1413	14	13	18	492.76	1	Sat	Sun	16	9
3	1003	2018-06-14 17:14:29	2018-06-20 10:15:39	6	1706	17	6	21	2289.49	1	Thu	Wed	17	10
4	1004	2018-06-16 16:56:51	2018-06-19 08:07:38	3	711	7	11	22	740.18	1	Sat	Tue	16	8
5	1005	2018-06-14 14:28:00	2018-06-17 08:43:21	3	230	2	30	13	383.12	1	Thu	Sun	14	8
6	1006	2018-06-16 16:39:58	2018-06-18 08:41:07	2	1629	16	29	24	1003.84	1	Sat	Mon	16	8

hotel_elevator dataset

```
> head(hotel_elevator)
```

	user_id	room_id	day_of_trip	car	from	to	timestamp	week	day_of_week	ts_hour
1	1001	1403	1	1	1	14	2018-06-13 17:42:47	1	Wed	17
2	1001	1403	2	1	14	1	2018-06-14 07:57:07	1	Thu	7
3	1002	1413	1	1	1	14	2018-06-16 16:20:43	1	Sat	16
4	1002	1413	2	1	14	1	2018-06-17 09:08:48	1	Sun	9
5	1003	1706	1	1	1	17	2018-06-14 17:23:06	1	Thu	17
6	1003	1706	2	3	17	1	2018-06-15 10:24:06	1	Fri	10

Merging Data

Objective: Merge hotel_elevator, hotel_door, and hotel_frontdesk for unified analysis.

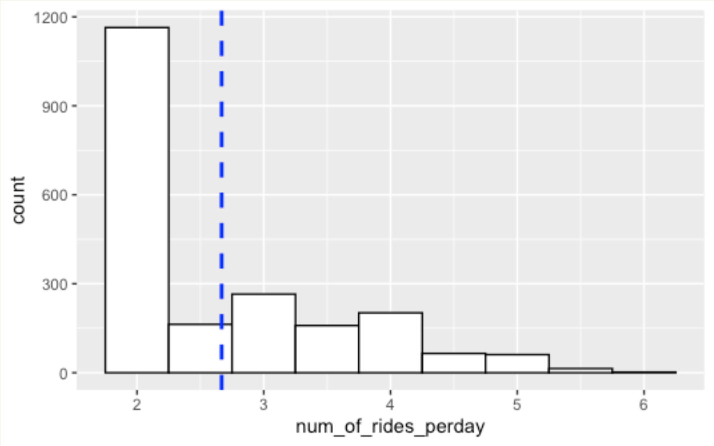
1. Used hotel_frontdesk as the base dataset (guest_id as the unique key).

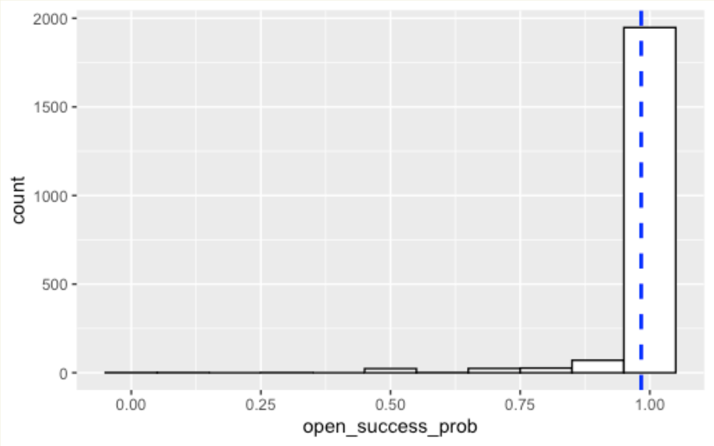
1. Derived variables:

From hotel_elevator: **num_of_rides_perday** (average elevator rides per day).

From hotel_door: **open_success_prob** (success rate of door access).

1. Aggregated multi-row data into concise formats

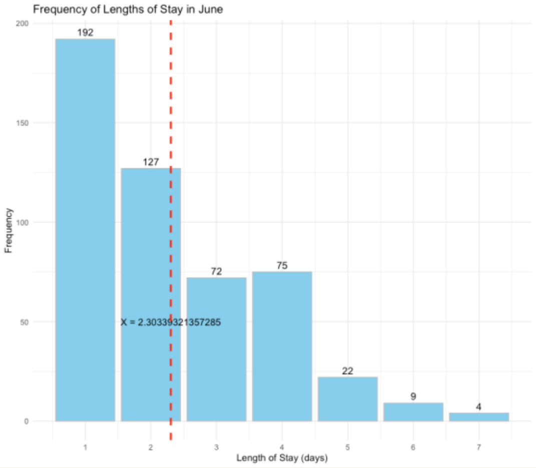




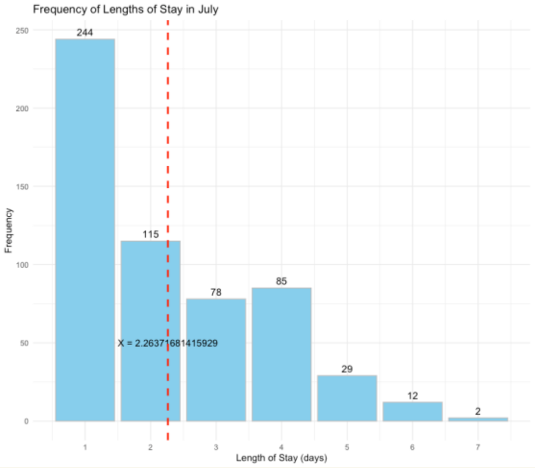
1. Project Intro & Data Merging
2. **Stories We Found**
3. Data Preprocessing
4. LASSO Model: Hotel Price
5. Random Forest: Length of Stay

Peak Summer Months and their Stay Lengths

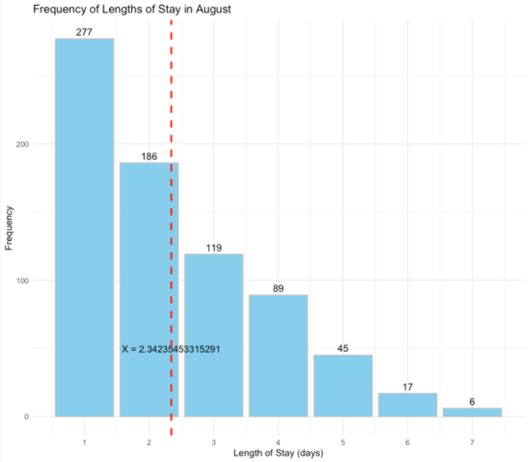
- Significant drop in bookings in September
- August is the peak summer month
- August signifies the last month of summer vacation for most, so customers tend to make the most of it before a return to work and school
- Longest average stay length: August
- Shortest average stay length: September



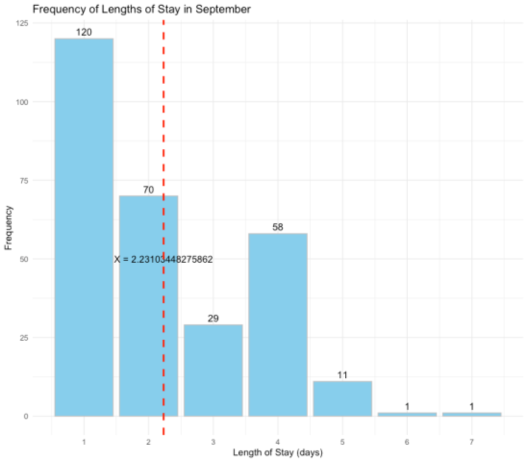
# of hotel bookings in June: 501



# of hotel bookings in July: 565



# of hotel bookings in August 739:

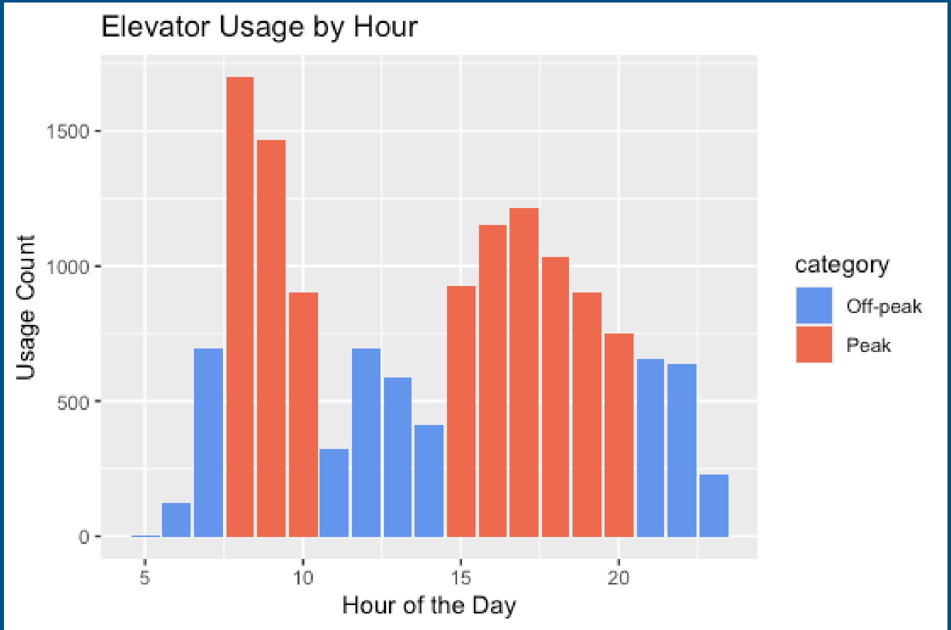


# of hotel bookings in September: 290

Elevator Peak vs. Off-peak Hours

After reviewing the *hotel_elevator* dataset, we wanted to observe what the off-peak and peak hours are for the hotel based on elevator usage.

From the graph, we can conclude that elevator usage in the hotel is highest during two distinct peak periods: **8–10 AM** and **3–8 PM**



How did we do this?

R-Code

1. Group data by timestamp and count number of elevator usages
2. Separate hours into two groups based on median usage count of 695:
   a. Peak
   b. Off-peak
3. Plot usage counts by hour with ggplot

```
# peak hours of elevator usage

library(dplyr)
library(ggplot2)
# Group data by 'ts_hour' and count the number of elevator usages
hourly_usage <- elevator_data %>%
  group_by(ts_hour) %>%
  summarise(usage_count = n()) %>%
  arrange(ts_hour)

# Categorize hours into 'Peak' and 'Off-peak' based on quantiles
hourly_usage <- hourly_usage %>%
  mutate(category = ifelse(usage_count > median(usage_count), "Peak", "Off-peak"))

# Display the resulting dataset
print(hourly_usage)

# Plot the usage counts by hour
ggplot(hourly_usage, aes(x = ts_hour, y = usage_count, fill = category)) +
  geom_bar(stat = "identity") +
  labs(title = "Elevator Usage by Hour",
       x = "Hour of the Day",
       y = "Usage Count") +
  scale_fill_manual(values = c("Off-peak" = "cornflowerblue", "Peak" = "coral2")) +
  theme_gray()
```

1. Project Intro & Data Merging
2. Stories We Found
3. **Data Preprocessing**
4. LASSO Model: Hotel Price
5. Random Forest: Length of Stay

Missing Data

- Consider multicollinearity, dropped unused variables (e.g., room_id, redundant timestamps)

-Factored categorical variables (e.g., in_day_of_week, out_day_of_week)

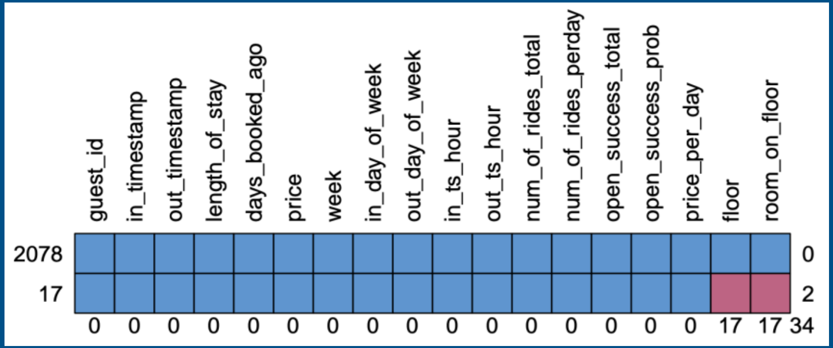
- Handled missing values in floor and room_on_floor with median imputation

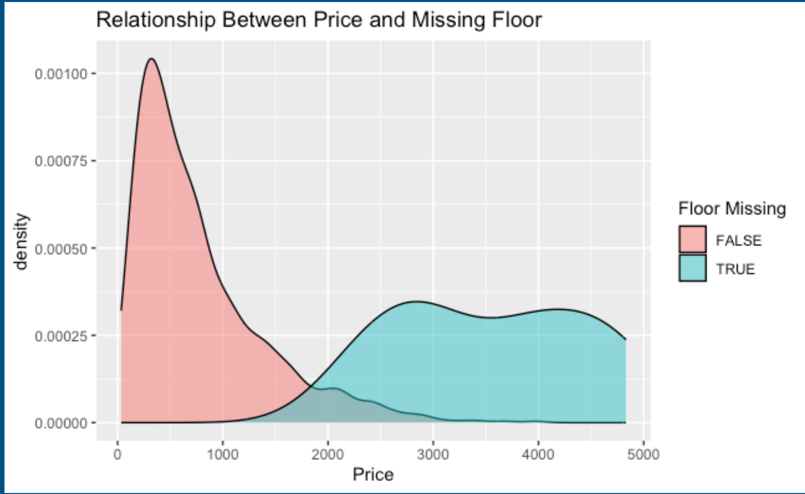
- Explore: MAR or NMAR? Test the relationship with observed variables (eg: price)

```
md.pattern(hotel_frontdesk)
hotel_frontdesk$missing_floor <- is.na(hotel_frontdesk$floor)
hotel_frontdesk$missing_room_on_floor <- is.na(hotel_frontdesk$room_on_floor)

ggplot(hotel_frontdesk, aes(x = price, fill = missing_floor)) +
  geom_density(alpha = 0.5) +
  labs(title = "Relationship Between Price and Missing Floor",
       x = "Price",
       fill = "Floor Missing")
t.test(price ~ missing_floor, data = hotel_frontdesk)
```

-Reason for missing value: missing data related the floor and room_id due the need to protect privacy.





t = -12.763, df = 16.125, p-value = 7.602e-10
Alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to 0

1. Project Intro & Data Merging
2. Stories We Found
3. Data Preprocessing
4. **LASSO Model: Hotel Price**
5. Random Forest: Length of Stay

Imputing Data

Using mice, we imputed the missing data with *pmm* method (predictive mean matching)

```
# Perform imputation using mice
imputed_data <- mice(hotel_data, m = 5, method = 'pmm')

# Replace the original data with the completed dataset
data <- complete(imputed_data)
```

Splitting Data

Randomly split **80%** into training data and **20%** into test data

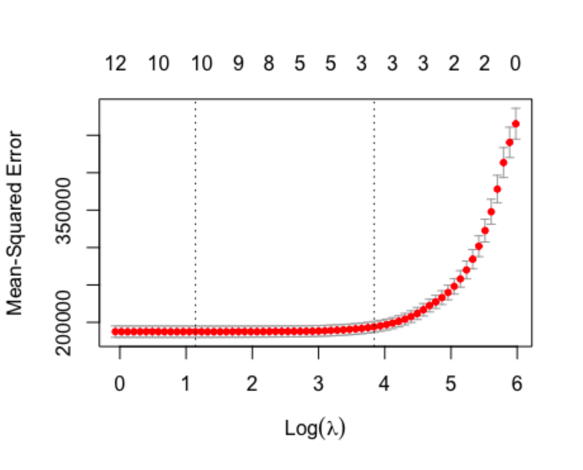
```
# Training sets + splitting data
hotel_data_split<-initial_split(data,prop=0.8)
hotel_data_train<-training(hotel_data_split)
hotel_data_test<-testing(hotel_data_split)
# Assign train and test data
train_x<-data.matrix(hotel_data_train[,c(2,4,5,6,7,9,10,11,12,13,15,17)])
train_y<-hotel_data_train$price_per_day
test_x<-data.matrix(hotel_data_test[,c(2,4,5,6,7,9,10,11,12,13,15,17)])
```

LASSO Model

Need to find the optimal λ: the smallest is **3.13017**

```r
# LASSO model
LASSO_cv_model<-cv.glmnet(train_x,hotel_data_train$price,
                          alpha=1)
#Find the best lambda value
smallest_lambda<-LASSO_cv_model$lambda.min
smallest_lambda

plot(LASSO_cv_model)
```



The lower right picture showcases the coefficients of the optimal model.

```r
Optimal_model<-glmnet(train_x,hotel_data_train$price,
                      alpha=1,lambda=smallest_lambda)

# The coefficient of the optimal model.
coef(Optimal_model)
```

```
> coef(Optimal_model)
13 x 1 sparse Matrix of class "dgCMatrix"
                              s0
(Intercept)         62296.755329
in_timestamp           -3.535552
length_of_stay        270.177890
floor                  63.772420
room_on_floor          -1.246679
days_booked_ago        -1.795136
week                   -3.412858
in_day_of_week          .
out_day_of_week        -2.312401
in_ts_hour              5.283702
out_ts_hour             .
num_of_rides_perday    24.390434
open_success_prob     -62.730414
```

Model Evaluation

- Median(y_predicted)
    - **784.2862**
- R-Squared
    - **57.55%** of the variance is explained by the predictors in the LASSO model
- Root Mean Squared Error
    - On average, the model's predictions deviate by **$382.60** from the true values

```
# Use test set to predict price value
y_predicted<-predict(Optimal_model,s=smallest_lambda,
                     newx = test_x)
print(median(y_predicted))

# See the Model Evaluation Metrics
ssr_lasso<-sum((hotel_data_test$price-y_predicted)**2)
sst_lasso<-sum((hotel_data_test$
                  price-mean(hotel_data_test$price))**2)
rsq_lasso<-1-ssr_lasso/sst_lasso
Len <-length(hotel_data_test$price)
rmse_lasso<-sqrt(ssr_lasso/Len)
```

```
> print(rsq_lasso)
[1] 0.5755102
> print(rmse_lasso)
[1] 382.6031
```

Conclusion

- Relative to the size of the dataset, our model is acceptable
- Other models, such as the Random Forest model, may provide a better fit

1. Project Intro & Data Merging
2. Stories We Found
3. Data Preprocessing
4. LASSO Model: Hotel Price
5. **Random Forest: Length of Stay**

Median Imputing

Replaces missing values with the median of the observed values in each variable.

Why we chose this approach:

- Robust to outliers, reliable
- Central tendency

```r
# Check the number of missing values before imputation
cat("Missing values before imputation:\n")
cat("Floor:", sum(is.na(hotel_frontdesk$floor)), "\n")
cat("Room on Floor:", sum(is.na(hotel_frontdesk$room_on_floor)), "\n")

# Perform median imputation
hotel_frontdesk$floor[is.na(hotel_frontdesk$floor)] <- median
    (hotel_frontdesk$floor, na.rm = TRUE)
hotel_frontdesk$room_on_floor[is.na(hotel_frontdesk$room_on_floor)]
    <- median(hotel_frontdesk$room_on_floor, na.rm = TRUE)

# Check the number of missing values after imputation
cat("\nMissing values after imputation:\n")
cat("Floor:", sum(is.na(hotel_frontdesk$floor)), "\n")
cat("Room on Floor:", sum(is.na(hotel_frontdesk$room_on_floor)), "\n")
```

Splitting Data

- 80% training and 20% test sets
- as matrices

```r
hotel_frontdesk_split<-initial_split(hotel_frontdesk,prop=0.8)
hotel_frontdesk_train<-training(hotel_frontdesk_split)
hotel_frontdesk_test<-testing(hotel_frontdesk_split)
```

```r
# Assign train and test data
train_stay_x<-data.matrix(hotel_frontdesk_train[,c(2,5,6,7,9,10,12,13,15,17,18)])
train_stay_y<-hotel_frontdesk_train$length_of_stay
test_stay_x<-data.matrix(hotel_frontdesk_test[,c(2,5,6,7,9,10,12,13,15,17,18)])
test_stay_y<-hotel_frontdesk_test$length_of_stay
```

Random Forest

Configured with

2,500 trees

`mtry = ceiling(2 * sqrt(10))`

node size = 5.

Key predictors:

`Price_per_day`

`Num_of_rides_perday`

`floor`

```r
## Implement the Random forest to predict length of stay
hotel_length_rf_model <- randomForest(x=train_stay_x,
                                       y=train_stay_y,do.trace =50,
                                   ntree=2500,mtry=ceiling(2*sqrt(10)),
                                       nodesize = 5,replace = TRUE)
```

```r
# Report the importance of the model
importance(hotel_length_rf_model)
```

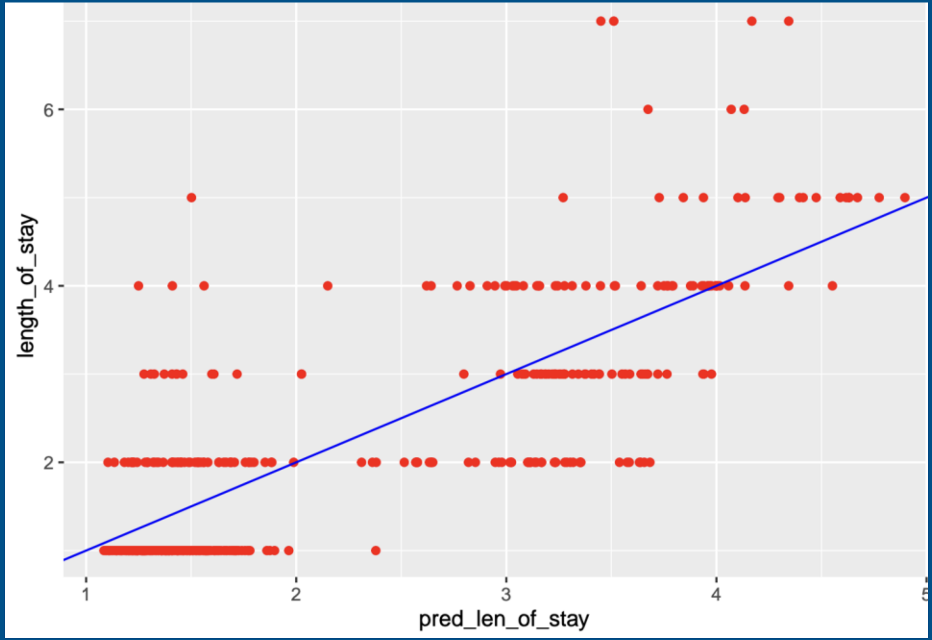
```
##                      IncNodePurity
## in_timestamp             174.91716
## floor                    128.60682
## room_on_floor            146.86190
## days_booked_ago          124.11399
## week                      60.23663
## in_day_of_week            85.81213
## in_ts_hour                90.64897
## out_ts_hour               64.40817
## num_of_rides_perday     1968.64195
## open_success_prob        121.38898
## price_per_day            248.39926
```

Evaluation

R-squared: 64.2%

RMSE: 0.74

Moderate predictive performance



Thank You!

Questions?