Use Case: US House Rental Prediction

INF 2167: R for Data Science

Prepared By: Seida Ahmed

December 13, 2021

Introduction

Several factors influence the rental prices of houses and apartments. A good rental price prediction model, on the other hand, examines the various qualities of a home and its surroundings to determine the most appropriate rent price. Homeowners, real estate brokers, and investors all require precise rent prices, which are generated by a solid rent price prediction model. Clients are then shown these best results by real estate agents or online real estate portals, making rental simple regardless of apartment type, location, or features. [1]

Problem Statement

The significance of this use case is to implement various machine learning models which predicts the monthly rental of a house based on a given attributes. The result of this analysis is important to different stakeholders. First it informs landlords a reasonable price for their property. Secondly, it guides renters whether they are paying fair price for the property they rented. The study will also explore what are the factors which significantly affects the rental price based on the given attributes.

Methodology

For this study the first step will be pre-processing and cleaning the dataset. Which includes identifying variables with missing values. taking care of missing values using various techniques based on the characteristics of the data. Pre-processing focuses on feature selection which is to drop variables which does not have any impact on our analysis such as identity variables. Changing categorical variables into number for the purpose of regression etc. The second step is exploratory data analysis phases which us for this case is to create various visualization using the dataset. The third step implementing various

machine learning algorithm which includes linear regression and non-linear regression (Random Forest, Gradient Bosting) for rental price prediction as.

Description of Dataset

The data is retrieved from Kaggle it can be accessed via the following link: https://www.kaggle.com/rkb0023/exploratory-data-analysis-house-rent-prediction/data.The dataset has 265190 rows and 22 columns. Below are dependent and independent variables for our machine learning models we have implemented

Dependent Variables

Price

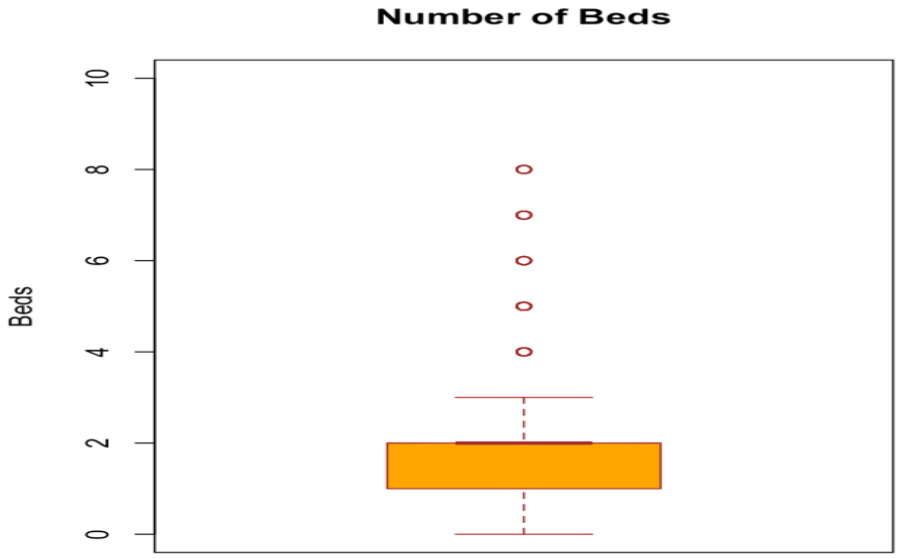
Independent Variables:

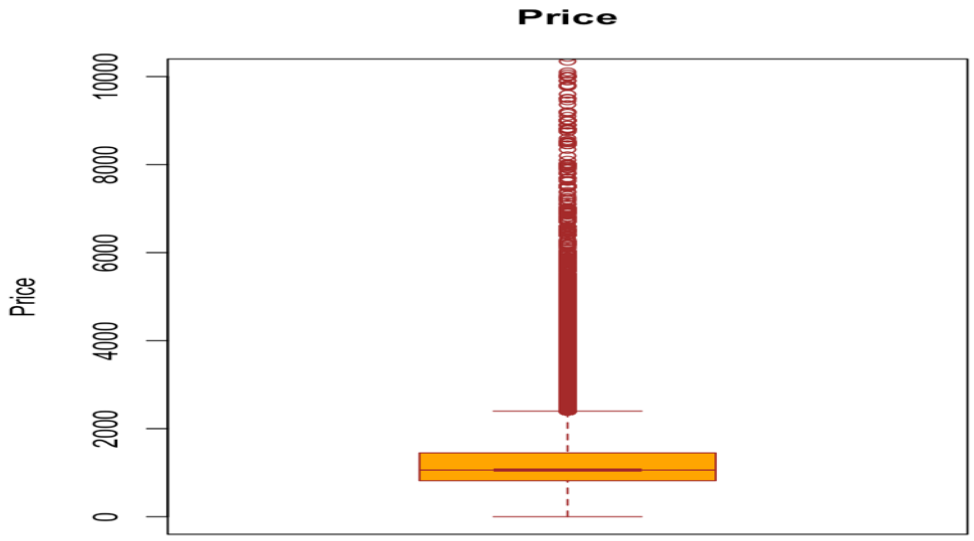
region, type, sqfeet, beds, baths, cats_allowed, dogs_allowed, smoking_allowed, wheelchair_access, electric_vehicle_charge, comes_furnished, laundry_options, parking_options, state, longitude and latitude

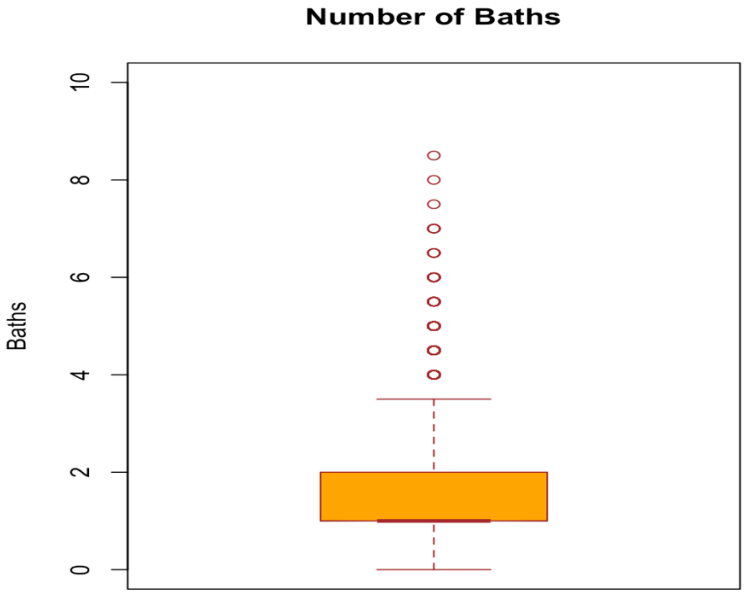
Exploratory Data Analysis

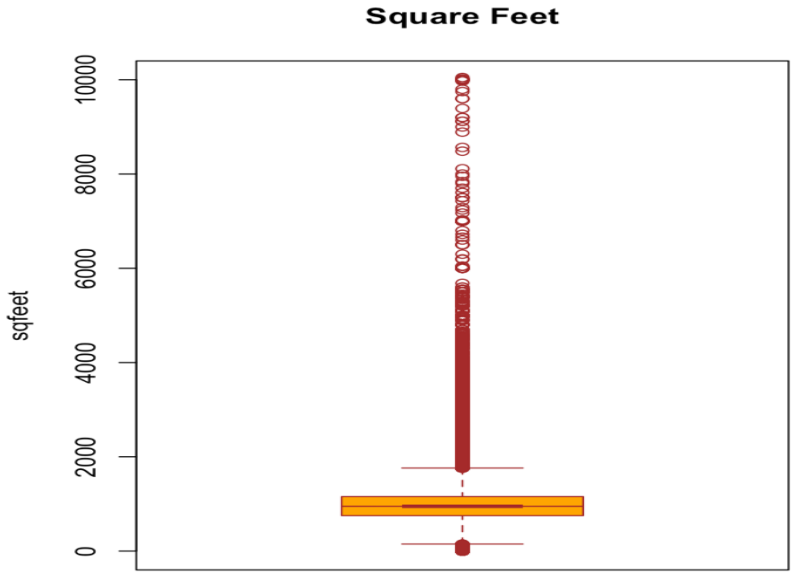
Box Plot to detect outliers

In order to detect possible outliner in numerical data I have implemented box plot for price, number of beds/baths, long and lat. From the visualization it is evident that there found pliantly of outliers which would affect our model. As a result, proper handling of outlier needs to be performed in the data prepressing/cleaning part.


Number of Beds
Beds
0
2
4
6
8
10

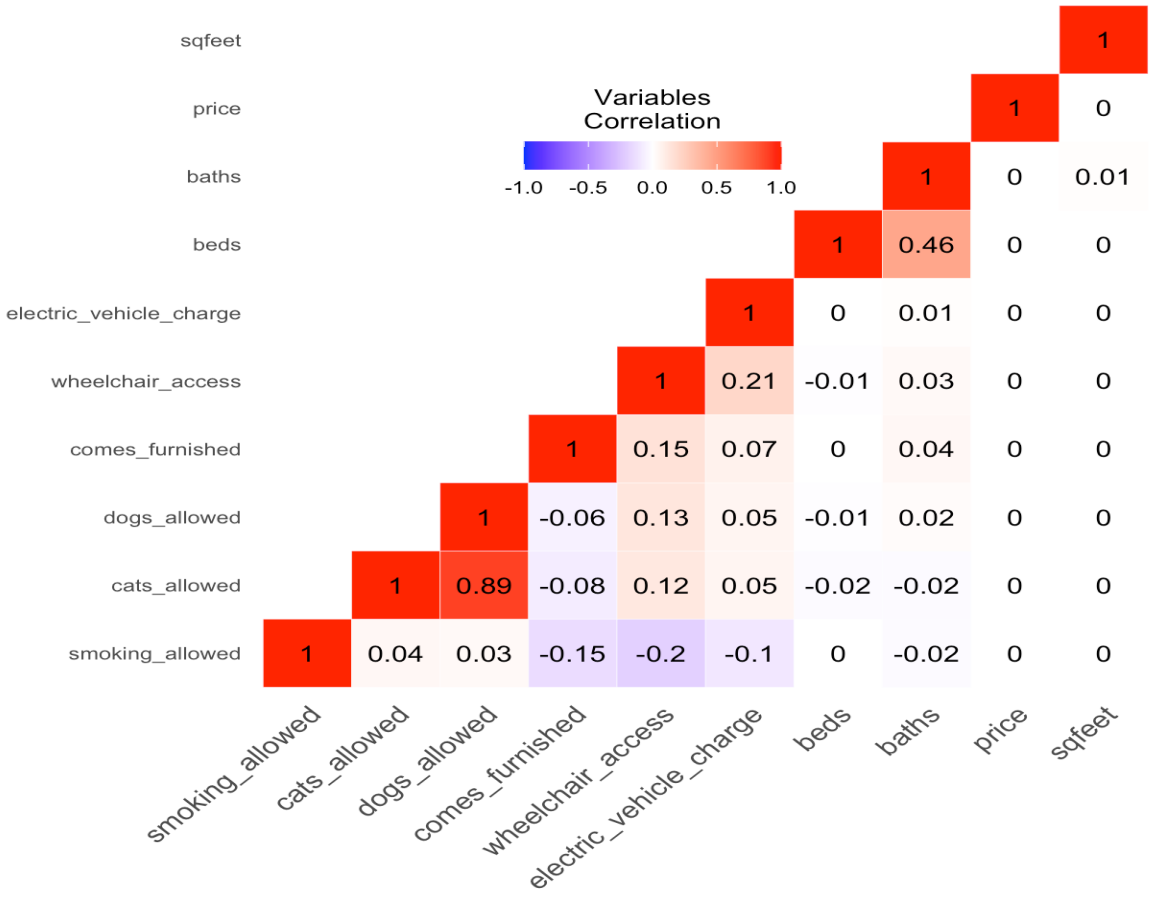

Price
Price
0
2000
4000
6000
8000
10000


Number of Baths
Baths
0
2
4
6
8
10


Square Feet
sqfeet
0
2000
4000
6000
8000
10000

Heatmap

Correlation Heatmap has been implemented to see relationship between features in the dataset. As the visualization depict there is no significant relationship between most of the variables except some of them. For example, there found an obvious strong correlation among cats_allowed and dog_allowed, and number of beds and number of bath.



Data Preprocessing and Cleaning

Getting proper data types.

Some records for instance number of baths and number of beds needs to have an integer value. However, some records in the dataset have decimal values for those variables. As a result, this has to be fixed in

order to build a robust model for our price prediction., the below R script eliminate the decimal gives proper values for the given variables.

```
data_df$baths <- as.integer(data_df$baths)
data_df$beds <-as.integer(data_df$beds)
```

Handling Missing Values

Variables containing most of the missing values in the dataset are parking_options, laundry_options, lat and long. In order to ger a robust prediction those missing values should be handled properly. First, I made a list of both categorical and numerical variables that contains missing values. And then filled the missing value of each feature with the mean/mode value of the respective variable.

```
data_df$long <-ifelse(is.na(data_df$long), mean(data_df$long, na.rm=TRUE),data_df$long)
data_df$lat <-ifelse(is.na(data_df$lat), mean(data_df$lat, na.rm=TRUE), data_df$lat)
data_df$laundry_options <-ifelse(is.na(data_df$laundry_options), mean(data_df$laundry_options,
na.rm=TRUE), data_df$laundry_options)
data_df$parking_options <-ifelse(is.na(data_df$parking_options), mean(data_df$parking_options,
na.rm=TRUE), data_df$parking_options)
```

Handling Outliers

Outliers were discovered via the boxplot for numerical variables. As a result, I removed outliers based on the interquartile range's upper and lower bounds.

```
data_df <- subset(data_df, 'price' <= 2400 & 'price' >= 1)
data_df <- subset(data_df, 'sqfeet' <= 1762 & 'sqfeet' >= 1)
data_df <- subset(data_df, 'beds' < 4& 'beds' >= 1)
data_df <- subset(data_df, 'baths' <  4 & 'baths' >= 1)
```

Variable Transformation

I log transformed some of the variables like price and sqfeet in order to get normal distribution. This tends to help Linear machine learning models such as linear regression.

```
data_df$price <-log(data_df$price)
data_df$sqfeet <-log(data_df$sqfeet)
```

Feature Selection and Encoding Categorical Variable

I dropped identity variables such as Id, URL, region-url since they don't have any significance for the model. Categorical variables need to be encoded as numerical for our MI prediction models. As a result, I encoded each categorical features used in to numerical using the below R code snippet.

```
data_df<- subset(data_df$price, select =-c(id, url, region_url))
data_df<-unclass(data_df)
```

Modeling

Linear Regression

From our EDA (Correlation Heatmap) we saw that most features don't have strong correlation. However, implementing linear regression will benefit as a reference for the other non-linear models.

R-Code

```
#10-fold CV...
cv_values <- rep(0, 10)
for(i in 1:length(cv_values)){
  print(i)
  inds <- sample(1:nrow(df_data), 0.80*nrow(df_data))
  tr_df <- df_data[inds,]
  te_df <- df_data[-inds,]
  lr <- lm(price ~., data = tr_df)
  preds <- predict(lr, newdata = te_df)
  mse <- sqrt(mean(te_df$price - preds)^2)
  cv_values[i] <- mse
}
cv_values
#CV-error:
mean(cv_values)
```

Output

```
cv_values
 2.9661212, 1.3976881 ,1.1155027, 0.8064655 ,0.7474112, 3.0046963, 0.8661778 ,4.4570636 0.1930080 ,
0.8799999
mean(cv_values)
1.643413
```

Single Tree Regression with Cross Validation

R code

```
cv_values <- rep(0, 10)
for(i in 1:length(cv_values)){
  print(i)
  inds <- sample(1:nrow(df_data), 0.80*nrow(df_data))
  tr_df <- df_data[inds,]
  te_df <- df_data[-inds,]
  tree1 <- rpart(price~., data = tr_df)
  preds <- predict(tree1, newdata = te_df)
  mse <- sqrt(mean(te_df$price - preds)^2)
  cv_values[i] <- mse
}
cv_values
#CV-error:
mean(cv_values)
```

Output

```
0.6519171 3.4094359 0.7264096 0.8278871 1.8937374 2.1141341 3.5423179 2.2289705 0.4959827
2.2409506
#CV-error:
Mean MSA
 1.813174
```

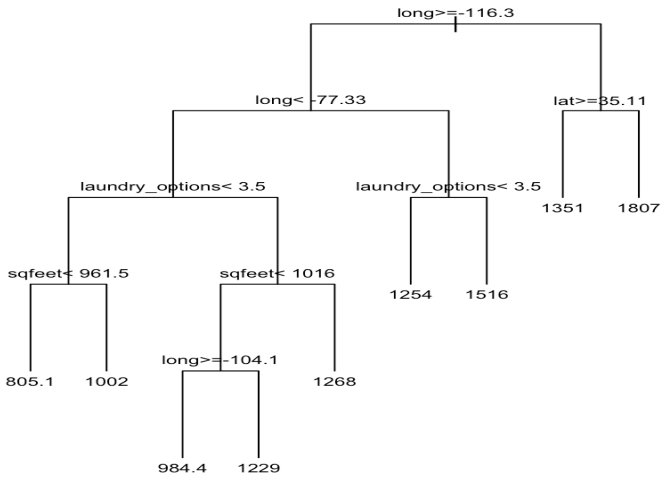


Fig: Output plot of Single Tree Regression

Single Tree Regression with parameter Pruning

R code:

```
cv_values <- rep(0, 10)
cp=list(0.01,0.02,0.1)
```

```
pr_value =10
for (j in 1:3){
 for(i in 1:length(cv_values1)){
   print(i)
   inds <- sample(1:nrow(df_data), 0.80*nrow(df_data))
   tr_df <- df_data[inds,]
   te_df <- df_data[-inds,]
   tree1 <- rpart(price~., data = tr_df)
   pfit <- prune(tree1, cp = cp[j], "CP")   preds <- predict(pfit, newdata = te_df)
   mse <- sqrt(mean(te_df$price - preds)^2)
   cv_values[i] <- mse
 }
 if (mean(cv_values) < pr_value)
    pr_value<-mean(cv_values)
    cp_best <- cp[j]
}
# best cp value
cp_best
pr_value
```

Output:

Best cp value among the given parameters is 0.1 and resulted MSA is 1.2

Gradient Booster

R-Code

```
for(i in 1:length(cv_values)){
print(i)
inds <- sample(1:nrow(df_data), 0.80*nrow(df_data))
tr_df <- df_data[inds,]
te_df <- df_data[-inds,]
boost1 <- gbm(price~., data = tr_df, distribution = "gaussian", n.trees = 5000,
          interaction.depth = 4, verbose = TRUE)
preds <- predict(boost1, newdata = te_df, n.trees = 5000)
mse <- sqrt(mean( (preds - te_df$price)^2))
mse
summary(boost1)
```

Output

mse=1.343899

According to the results of the gradient booster, long has the greatest influence, followed by sqfeet, which requires further study into how these variables, among others, have the greatest influence.

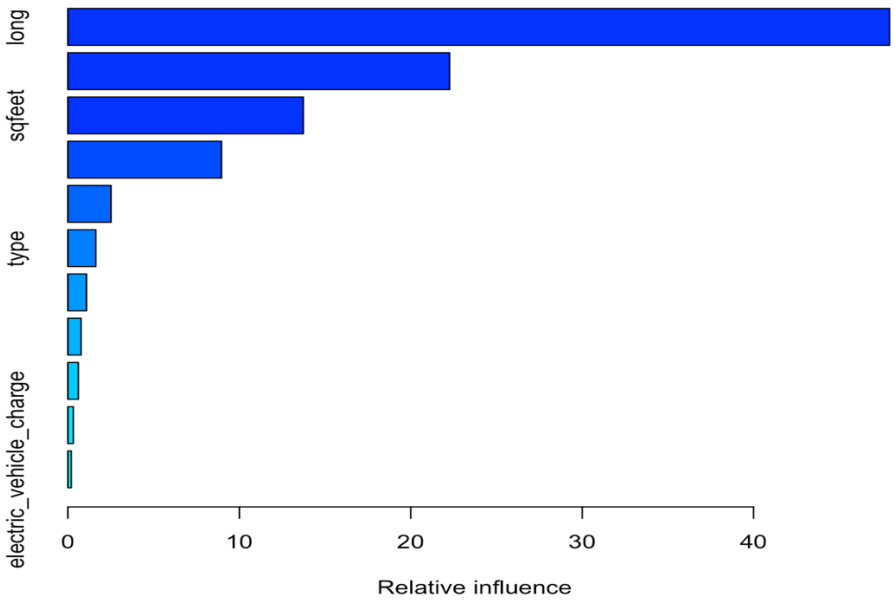


Fig: summary graph of Gradient Booster which shoes relative influence

Due to huge volume of the dataset, I could not be able to implement cross validation for the gradient booster. It is taking very long time to execute even the single iteration.

Conclusion

There were no significant differences across models based on the MSA score. The explanation for the results needs to be looked into further. The similarities in price prediction accuracy between the models (linear and nonlinear) are, in my opinion, related to the log transformation of numerical variables. Furthermore, all models yielded MSA values larger than one. It's possible that the dataset requires further preprocessing/cleaning. This is yet another duty that will necessitate further examination.

Reference

[1]Rent Price Prediction. (n.d.). Data Hunters. https://www.data-hunters.com/use_case/rental-price-prediction/

[2] R.A. (n.d.). *Exploratory Data Analysis-House Rent Prediction*. Kaggle. https://www.kaggle.com/rkb0023/exploratory-data-analysis-house-rent-prediction/notebook