Feature engineering

Load Libraries

```
library(tidyverse) # common libraries
```

Load Data

```
df<-read.csv("rent-amount.csv")
```

```
head(df)
```

1
2
3
4
5
6

6 rows | 1-1 of 11 columns

In EDA we conclude that the data set has too many outliers, which is why they require a type of processing.

I can think of 3 ways to deal with the situation.

- Remove outliers.
- Replace abnormal values by statistical measures.
- Perform a logarithmic transformation
- Calculate an upper range and based on the range replace outliers with normal values.

The first and second options are ruled out, since the data set has very few observations. And if we replace the outliers with statistical measures such as the mean or the median, we would be dirtying the data set, we would be affecting the distribution of the data.

We can combine the 3 and 4 option, we can establish an **upper range**. Based on the interval we create a criterion, we make a **sample** of random values that oscillate in those ranges. To be able to **replace** them and then perform a **logarithmic transformation** to smooth the data, improving their distribution.

```
attach(df)
```

```
upper_limit<-function(x,limit){
  mean<-mean(x) # calculate mean
  sd<-sd(x) # calculate standard division

  mean+limit*sd # calculate upper range
}
```

```
calculate_upper_limit<-function(feature){

  limits<-c(2,2.5,3,3.5,4)
  for(index in 1:5){

    print(paste("Upper Limit",limits[index]))
    print(upper_limit(feature,limits[index]))
    print("====================")
}
}
```

Upper Limits

Rent Amount

```
calculate_upper_limit(feature = rent.amount)
```

```
## [1] "Upper Limit 2"
## [1] 11549.18
## [1] "==================="
## [1] "Upper Limit 2.5"
## [1] 13337.52
## [1] "==================="
## [1] "Upper Limit 3"
## [1] 15125.85
## [1] "==================="
## [1] "Upper Limit 3.5"
## [1] 16914.19
## [1] "==================="
## [1] "Upper Limit 4"
## [1] 18702.52
## [1] "==================="
```

The best upper limit is 3 which we can round up to R $15,000. Since there are more rental houses that are around those prices as we can see in the histogram.

Fire Insurence

```
calculate_upper_limit(feature = fire.insurance)
```

```
## [1] "Upper Limit 2"
## [1] 156.8312
## [1] "==================="
## [1] "Upper Limit 2.5"
## [1] 181.4879
## [1] "==================="
## [1] "Upper Limit 3"
## [1] 206.1446
## [1] "==================="
## [1] "Upper Limit 3.5"
## [1] 230.8013
## [1] "==================="
## [1] "Upper Limit 4"
## [1] 255.458
## [1] "==================="
```

With an upper limit of 4 it is good. With an upper limit of 4 it is good. We can round it up to R $250 since there are very few cases where the price of fire insurance exceeds that amount.

Area

```
calculate_upper_limit(feature = area)
```

```
## [1] "Upper Limit 2"
## [1] 902.2629
## [1] "==================="
## [1] "Upper Limit 2.5"
## [1] 1090.043
## [1] "==================="
## [1] "Upper Limit 3"
## [1] 1277.822
## [1] "==================="
## [1] "Upper Limit 3.5"
## [1] 1465.602
## [1] "==================="
## [1] "Upper Limit 4"
## [1] 1653.382
## [1] "==================="
```

With an upper limit of 2.5 it is a good point. Since most of the departments do not exceed 1000 square meters.

```
replace_outlires<-function(data,normal_sample,upper_limit){

  return(ifelse(data>upper_limit,normal_sample,data))
}
```

Normal Values Sample for Upper Limit

```
set.seed(2018) # we define random seed, so that the numbers generated by the sample do not vary.
sample_rent<-sample(14500:15000,size=25,replace = T)
sample_fire_insurence<-sample(240:250,size=10,replace = T)
sample_area<-sample(900:1000,size=25,replace = T)
```

```
normal_values_rent<-replace_outlires(data=df$rent.amount,sample_rent,upper_limit = 15000)
normal_fire_insurence<-replace_outlires(data=df$fire.insurance,sample_fire_insurence,upper_limit = 250)
normal_area<-replace_outlires(data=area,sample_area,upper_limit = 1000)
```

Replace Outlires Values

```
df<- df %>%
  mutate(rent.amount=normal_values_rent) %>%
  mutate(fire.insurance=normal_fire_insurence) %>%
  mutate(area=normal_area)
```

With the mutate function we make modifications. We replace outliers with normal values, using a random sample.

We check the changes

```
df %>% select(rent.amount,area,fire.insurance) %>% summary()
```

```
##   rent.amount         area        fire.insurance
##  Min.   :  420   Min.   :  10.0   Min.   :  3.00
##  1st Qu.: 1800   1st Qu.:  58.0   1st Qu.: 23.00
##  Median : 3111   Median : 100.0   Median : 41.00
##  Mean   : 4383   Mean   : 145.2   Mean   : 58.11
##  3rd Qu.: 5952   3rd Qu.: 200.0   3rd Qu.: 77.00
##  Max.   :15000   Max.   :1000.0   Max.   :250.00
```

```
df<- df %>% mutate(fire.insurance=ifelse(fire.insurance==3,5,fire.insurance))
```

We replaced the minimum price value of fire insurance by five real Brasilian.

```
df %>% select(fire.insurance) %>% summary()
```

```
##  fire.insurance
##  Min.   :  4.00
##  1st Qu.: 23.00
##  Median : 41.00
##  Mean   : 58.11
##  3rd Qu.: 77.00
##  Max.   :250.00
```

logarithmic transformation

```
df<- df %>%

  mutate(area_log=log(area)) %>%
  mutate(fire.insurance_log=log(fire.insurance)) %>%
  mutate(rent.amount_log=log(rent.amount))
```

We perform logarithmic transformation to improve the distribution of continuous data.

```
histogram<-function(x,...){

    df %>%
      ggplot(aes(x=x,y=..density..)) +
      geom_histogram(color="black",fill="#FFF0C9") +
      geom_density(color="black",lwd=1) +
      geom_vline(aes(xintercept=mean(x),color="mean")) +
      geom_vline(aes(xintercept=median(x),color="median")) +
      labs(col="Stadistcs") +
      theme(legend.position = "top") +
      ...

  }
```

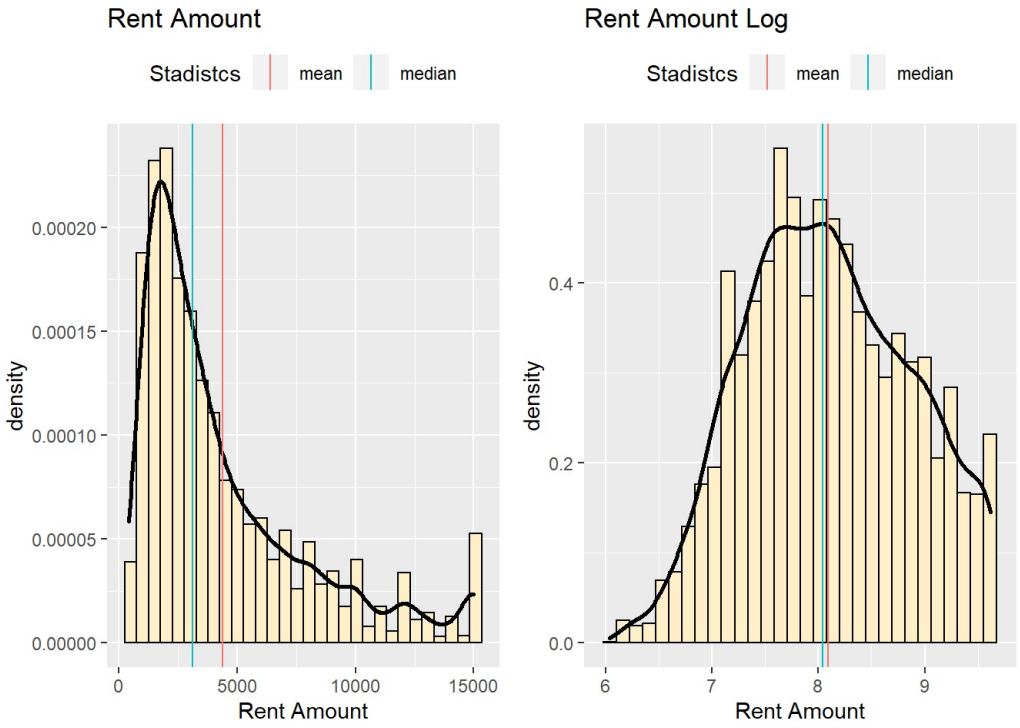
```
library(gridExtra)
```

```
rent_amount_histogram=histogram(df$rent.amount,labs(x="Rent Amount",title = "Rent Amount"))
area_hsitogram=histogram(df$area,labs(x="Area",title = "Area"))
fire_insurence_histogram=histogram(df$fire.insurance,labs(x="Fire Insurence",title = "Fire Insurence"))

rent_amount_histogram_log=histogram(df$rent.amount_log,labs(x="Rent Amount",title = "Rent Amount Log"))
area_histogram_log=histogram(df$area_log,labs(x="Area",title = "Area Log "))
fire_insurence_histogram_log=histogram(df$fire.insurance_log,labs(x="Fire Insurence",title = "Fire Insurence Log"
))
```

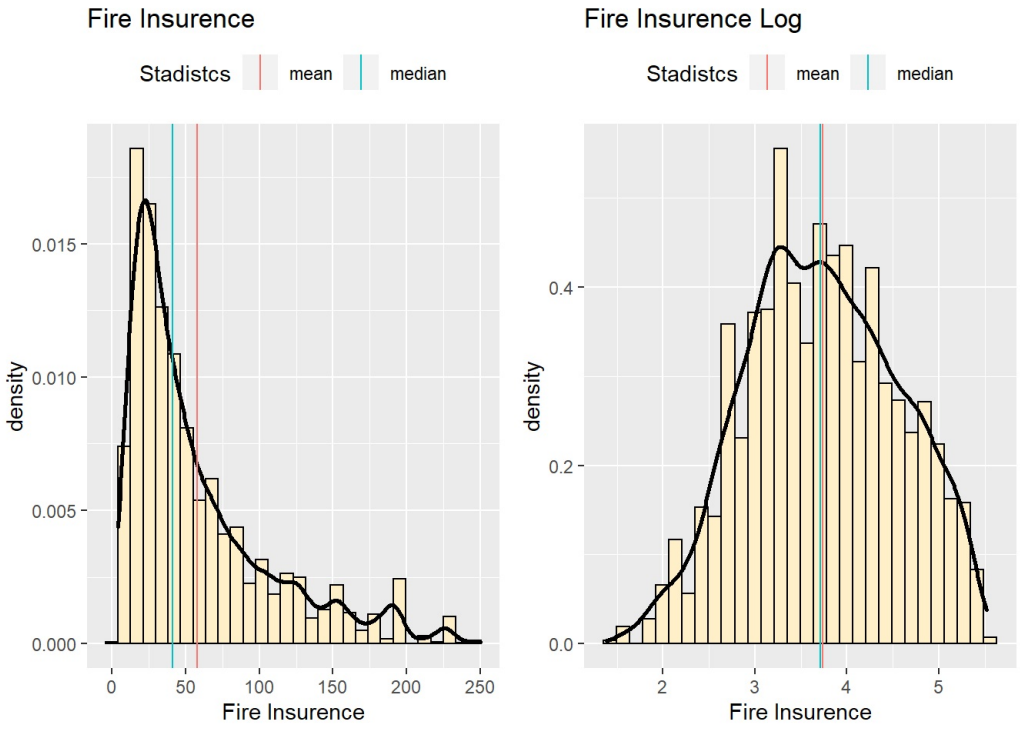
```
grid.arrange(rent_amount_histogram,
             rent_amount_histogram_log,nrow=1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



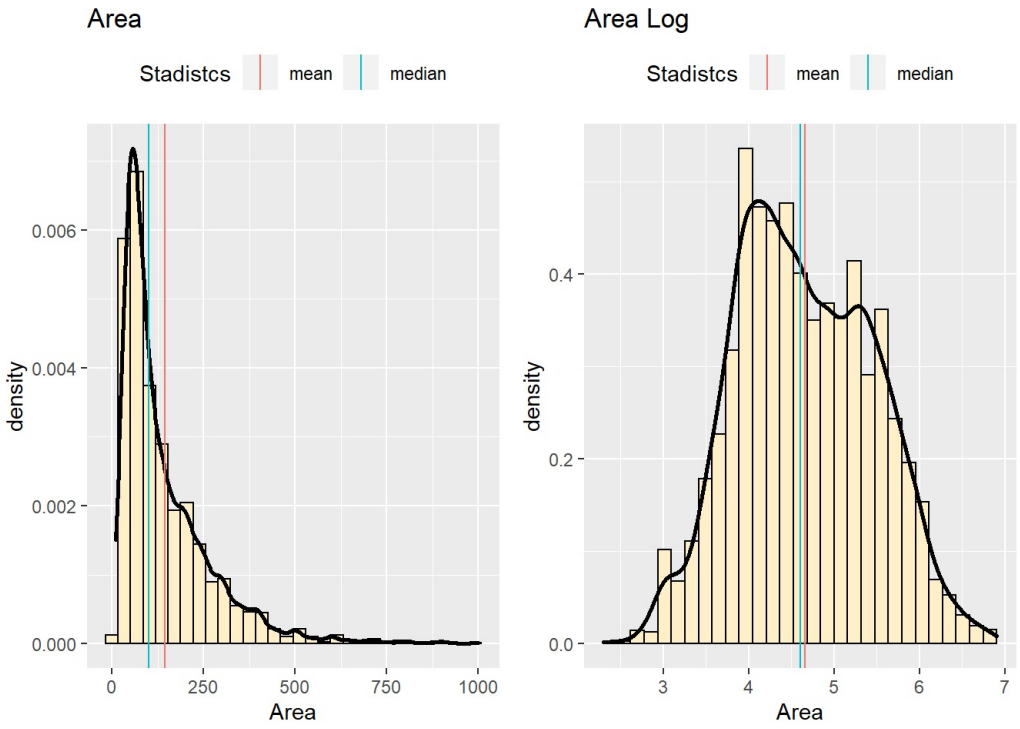

```
grid.arrange(fire_insurence_histogram,
             fire_insurence_histogram_log,nrow=1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
grid.arrange(area_hsitogram,
             area_histogram_log,nrow=1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We observed a great improvement in the distribution of the data.

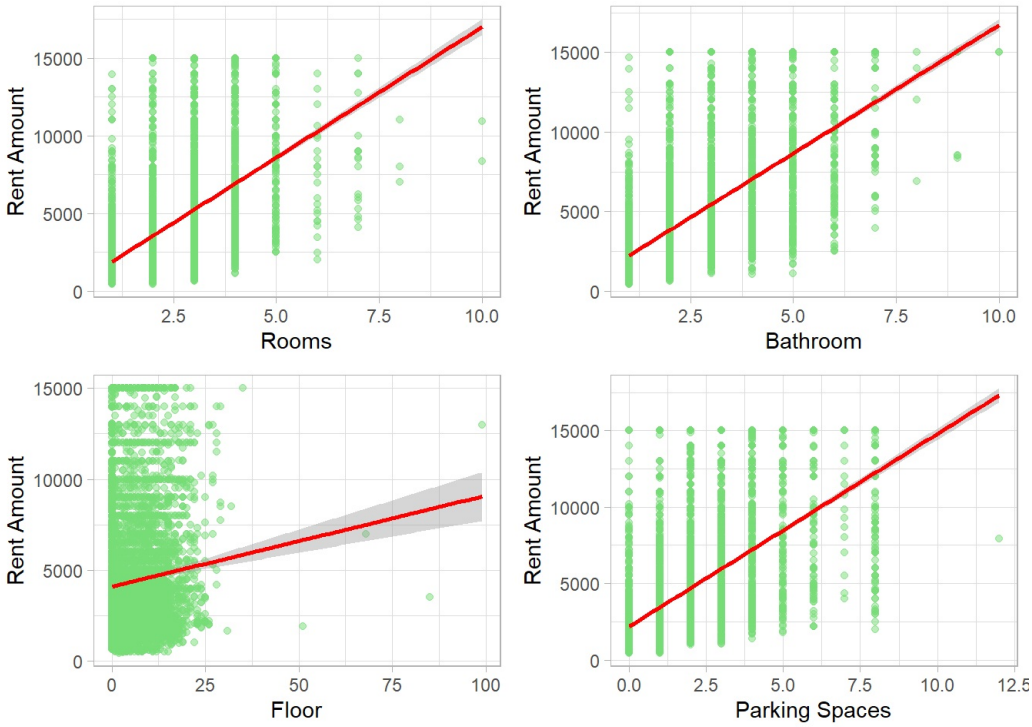
```
scattter_plot<-function(x_feature,...){

  ggplot(data=df,aes(x=x_feature,y=rent.amount)) +
  geom_point(color="#77dd77",alpha=0.5) +
  theme_light() +
  geom_smooth(method = "lm",color="red") +
  ...
}
```

```
rooms_scatter<-scattter_plot(df$rooms,labs(x="Rooms",y="Rent Amount"))
bathroom_scatter<-scattter_plot(df$bathroom,labs(x="Bathroom",y="Rent Amount"))
floor_scatter<-scattter_plot(df$floor,labs(x="Floor",y="Rent Amount"))
parking_spaces_scatter<-scattter_plot(df$parking.spaces,labs(x="Parking Spaces",y="Rent Amount"))
```

```
grid.arrange(rooms_scatter,
             bathroom_scatter,
             floor_scatter,
             parking_spaces_scatter)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



We can replace values greater than 7, since there are few departments that exceed that number of rooms. I will replace it with a value of 7, since there is little data and it will not affect the distribution of the data.

For the bathrooms we can change replace those values that are greater than 8, since there are very few bathrooms greater than the mentioned amount.

There are very few departments where the buildings of the rental houses are greater than 30.

There are very few apartments, where parking spaces exceed 8 units.

```
replace_values<-function(x,limit){

  ifelse(x>limit,limit,x)

}
```

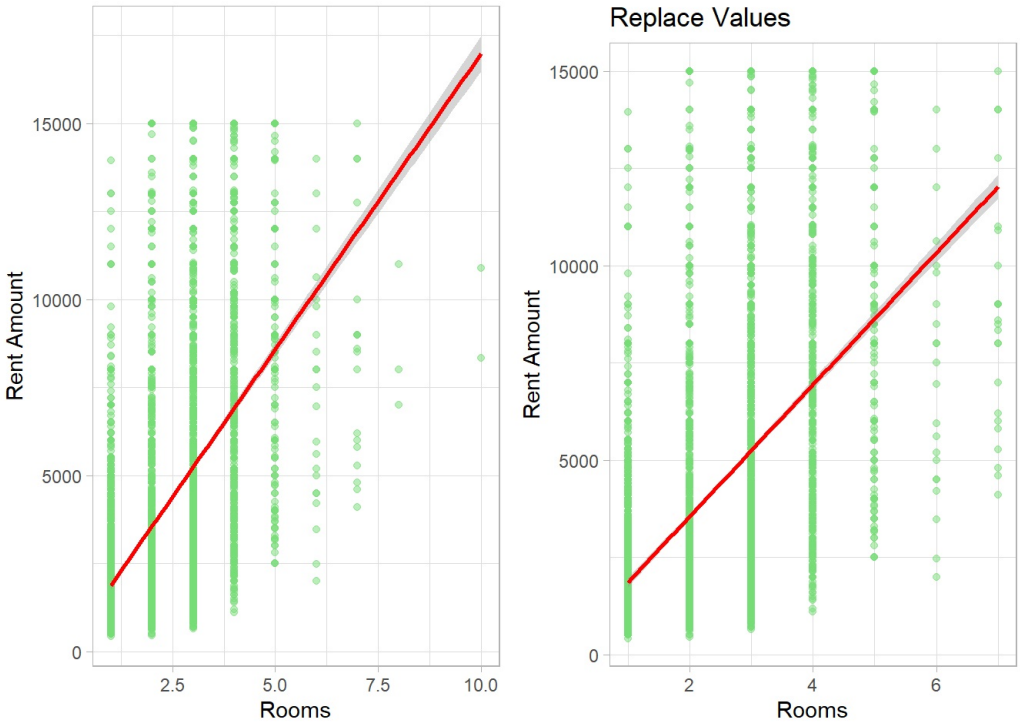
```
replace_rooms<-mapply(replace_values,rooms,7)
replace_bathrooms<-mapply(replace_values,bathroom,8)
replace_floor<-mapply(replace_values,floor,30)
replace_spaces_parking<-mapply(replace_values,parking.spaces,8)
```

```
df<- df %>%
  mutate(replace_rooms=replace_rooms) %>%
  mutate(replace_bathrooms=replace_bathrooms) %>%
  mutate(replace_floor=replace_floor)%>%
  mutate(replace_parking=replace_spaces_parking)
```

```
rooms_scatter_replace<-scattter_plot(df$replace_rooms,labs(x="Rooms",y="Rent Amount",title ="Replace Values"))
bathroom_scatter_replace<-scattter_plot(df$replace_bathrooms,labs(x="Bathroom",y="Rent Amount",title = "Replace V
alues"))
floor_scatter_replace<-scattter_plot(df$replace_floor,labs(x="Floor",y="Rent Amount",title = "Replace Values"))
parking_scatter_replace<-scattter_plot(df$replace_parking,labs(x="Floor",y="Rent Amount",title = "Replace Values"
))
```

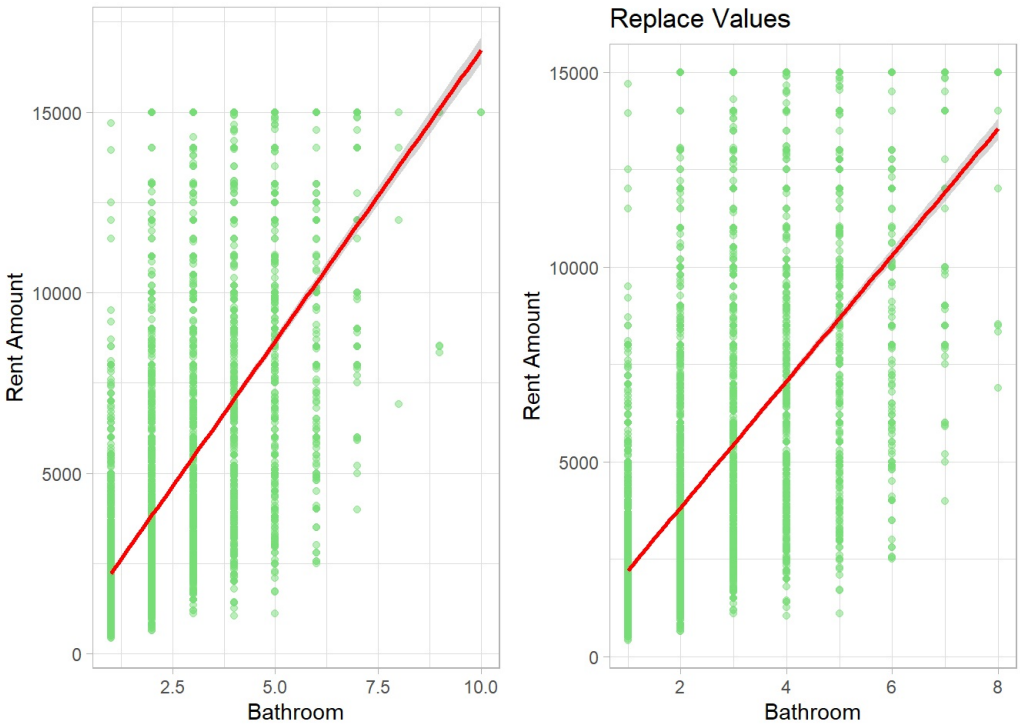
```
grid.arrange(rooms_scatter,rooms_scatter_replace,nrow=1)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



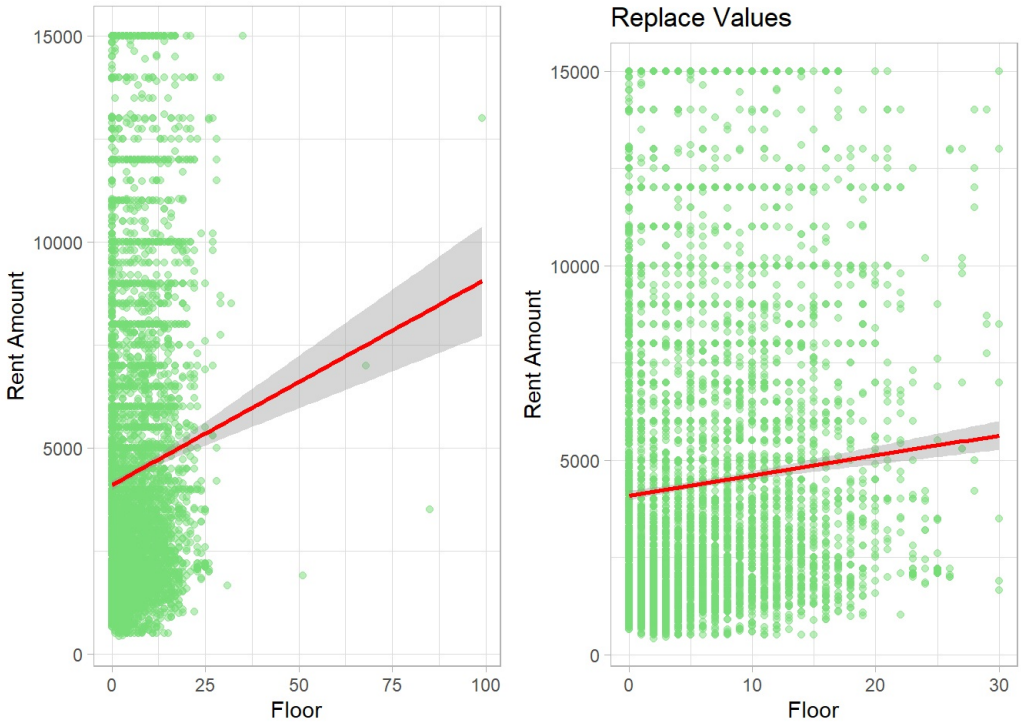

```
grid.arrange(bathroom_scatter,bathroom_scatter_replace,nrow=1)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



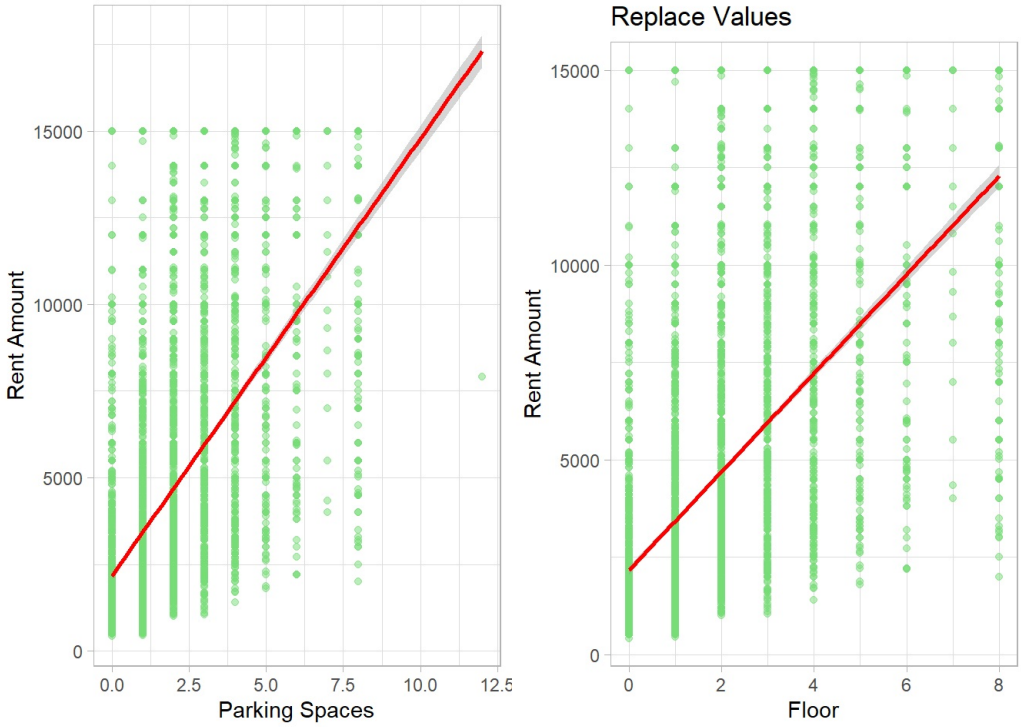

```
grid.arrange(floor_scatter,floor_scatter_replace,nrow=1)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```




```
grid.arrange(parking_spaces_scatter,parking_scatter_replace,nrow=1)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



We finish with the cleaning of our data.

We remove old variables

```
df<-df %>%

  select(-rent.amount,-area,-fire.insurance) %>%
  select(-rooms,-bathroom,-floor,-parking.spaces)
```

Rename Features

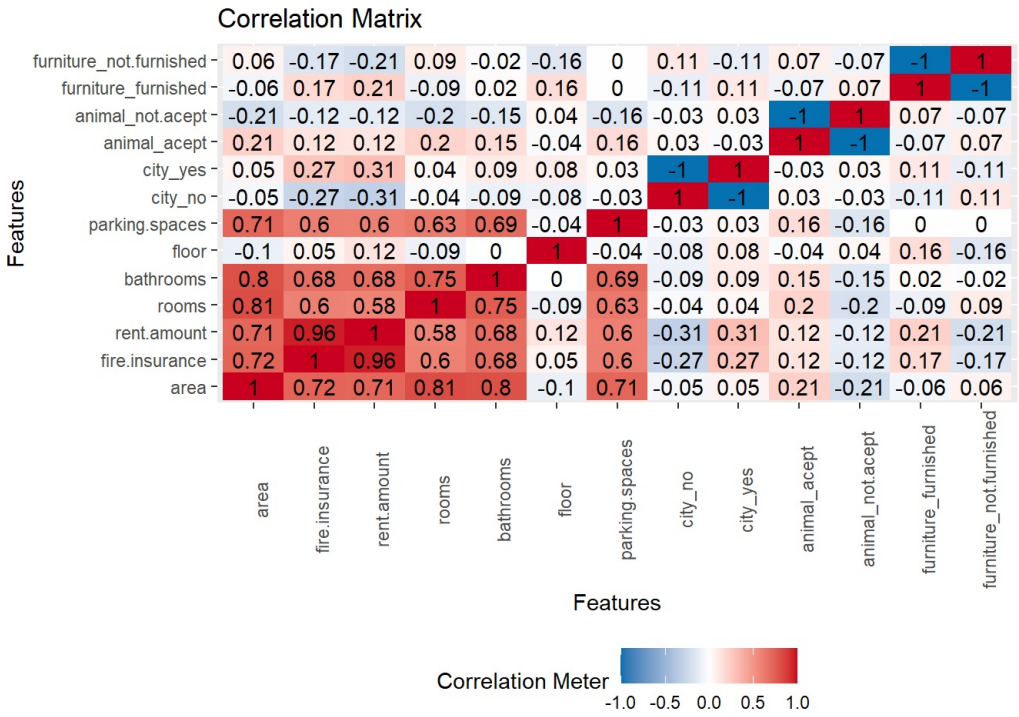
```
df_clear<-df %>%
  rename(fire.insurance=fire.insurance_log,
         area=area_log,
         rent.amount=rent.amount_log) %>%

  rename(bathrooms=replace_bathrooms,
         floor=replace_floor,
         rooms=replace_rooms)
```

```
df_clear<- df_clear %>%
  rename(parking.spaces=replace_parking)
```

```
library(DataExplorer) # correlation matrix
```

```
plot_correlation(df_clear,title = "Correlation Matrix")
```



Save Dataframe clear

```
write.csv(df_clear,"rent-amount-clear.csv",row.names = FALSE)
```