GroupProject

Jerome Agius & Matthias Bartolo

27/11/2021

Question 1

Loading the data into a data frame

```
dataInitial <- read.csv("~/GroupProject/data-rental.csv")
#loading the csv file into the dataInitial data frame
```

```
typeof(dataInitial$house_price)
typeof(dataInitial$rental_agency)
typeof(dataInitial$city)
typeof(dataInitial$bedrooms)
typeof(dataInitial$surface)
summary(dataInitial)
```

Question 2

2.1 (i)

The data features found in the .csv file include house_price, rental_agency, city, bedrooms and surface.

Observations:

1. house_price: integer, continuous variable, ratio(0 implies the absence of money), has 31 NA values.
2. rental_agency: character, categorical variable, nominal(They have no inbuilt order)
3. city: character, categorical variable, nominal(They have no inbuilt order)
4. bedrooms: integer, discrete variable(as one can only have a whole number of rooms example 3, one can't have 3.5 rooms for instance), ratio(0 implies the absence of a bedroom)
5. surface: integer, continuous variable, ratio(0 implies the absence of land)

2.1 (ii)

1. To be able to determine the relationship between the different variables one could output a scatter plot and determine the trend (best fit line).
2. One must remove any instances of duplicate data, this can be done by using the function duplicated() which will retrieve all duplicate rows or unique() which will retrieve all unique rows instead.

   example: data <- read.csv("csv file path")

   No_Duplicates <- unique(data)

   which will store all the unique rows in the variable data thus removing all instances apart from 1 of the duplicated data.
3. One can also work out the central tendency, dispersion and standard deviation where applicable to better understand how the data varies.

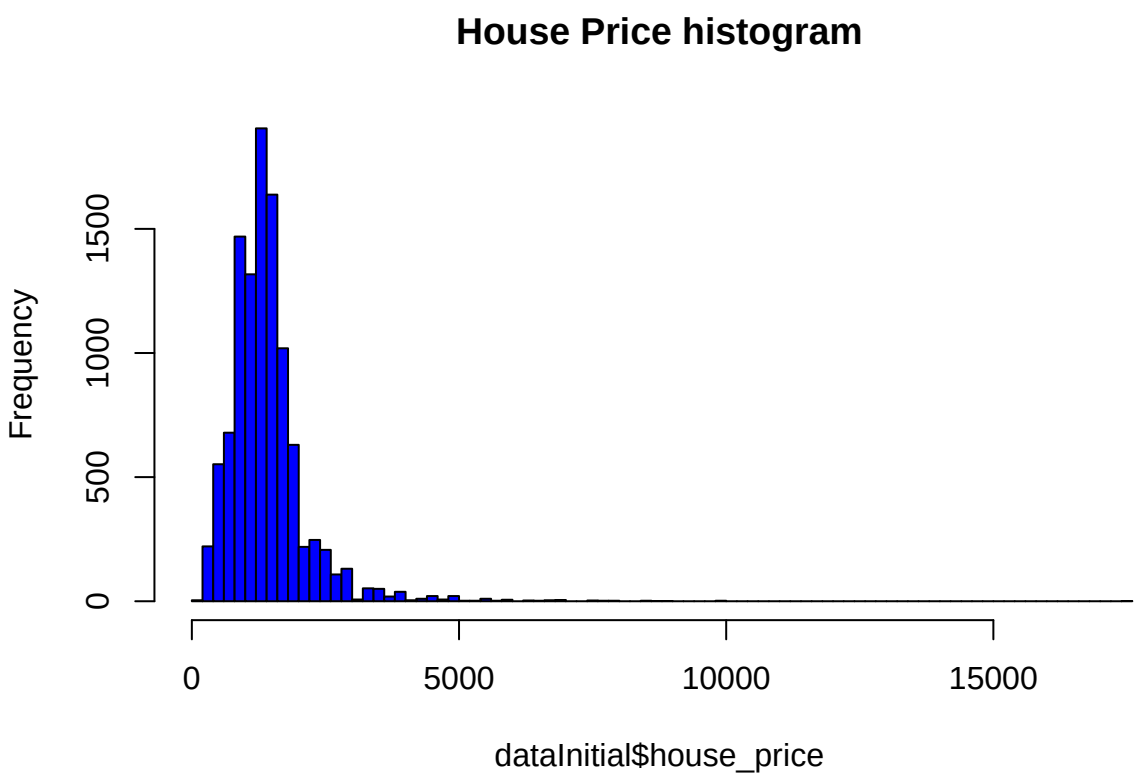
2.1 (iii)

```
library(tidyverse)
library(ggpubr)
```

```
mean(dataInitial$house_price, na.rm=TRUE)
```

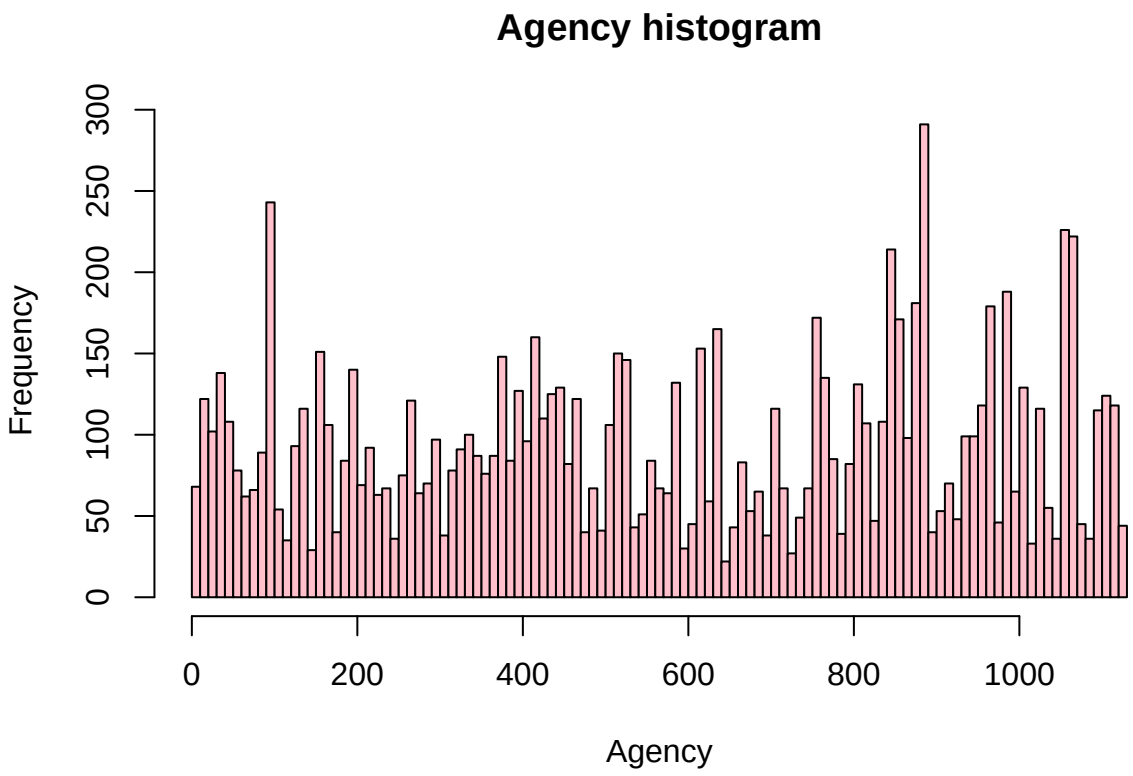
```
## [1] 1423.65
```

```
hist(dataInitial$house_price,xlim=c(0,17500), main="House Price histogram", col="blue",breaks=100)
```

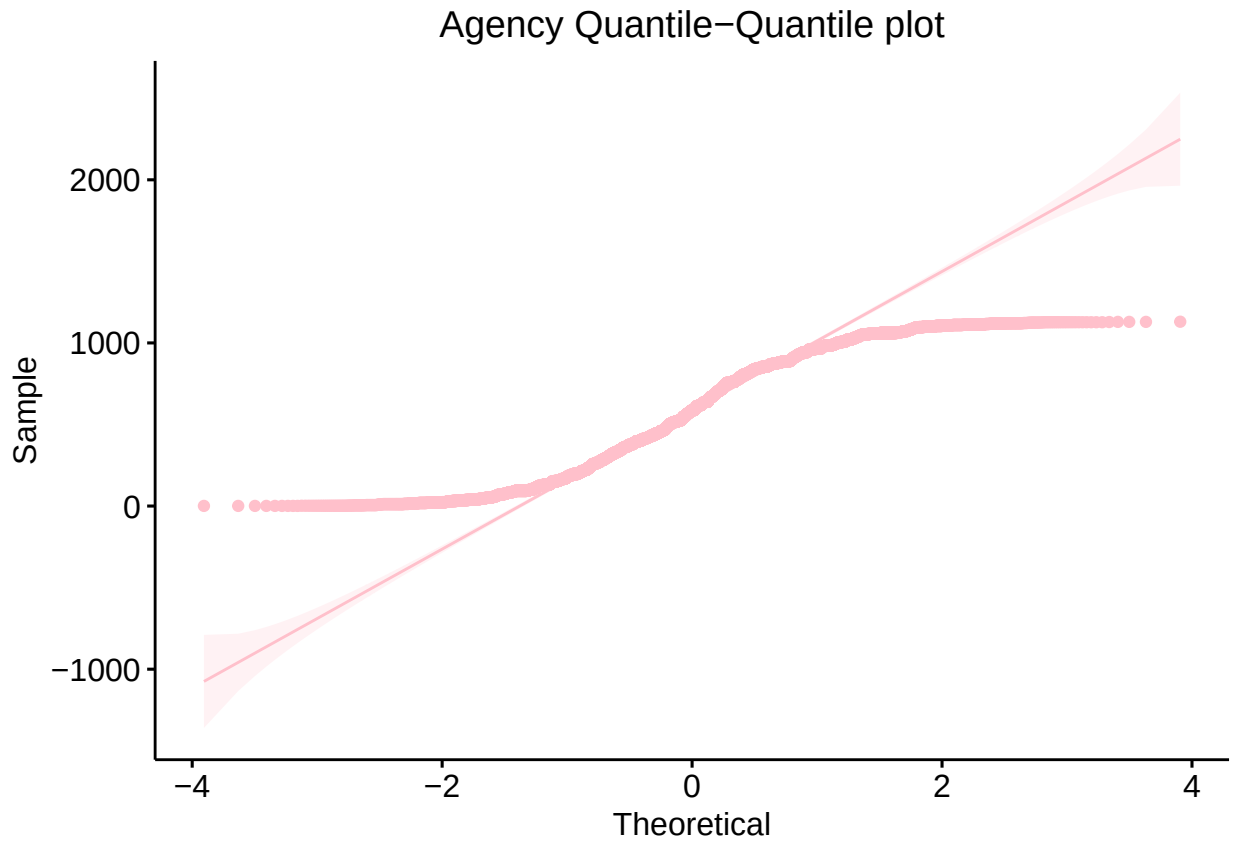


```
#According to this plot the house_price follows a normal distribution as its
#symmetrical about the mean if NA values are ignored
ggqqplot(dataInitial$house_price, title="house_price Quantile-Quantile plot", col="blue")+
  theme(plot.title = element_text(hjust = 0.5))
```

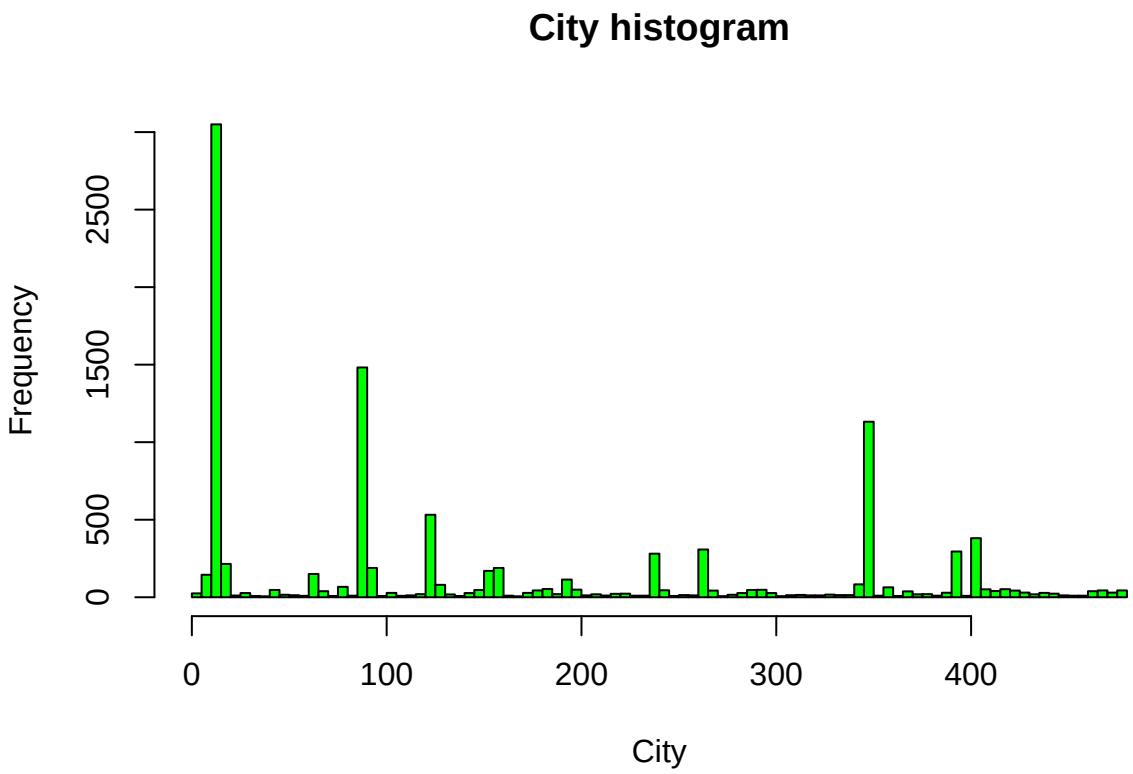

```
agency <- c(dataInitial$rental_agency)
agency.factor <- factor(agency)
Agency <- as.numeric(agency.factor)

#This histogram provides no useful information as the data doesn't
#seem to follow any distribution
hist(Agency, breaks=100, main="Agency histogram", col="pink")
```



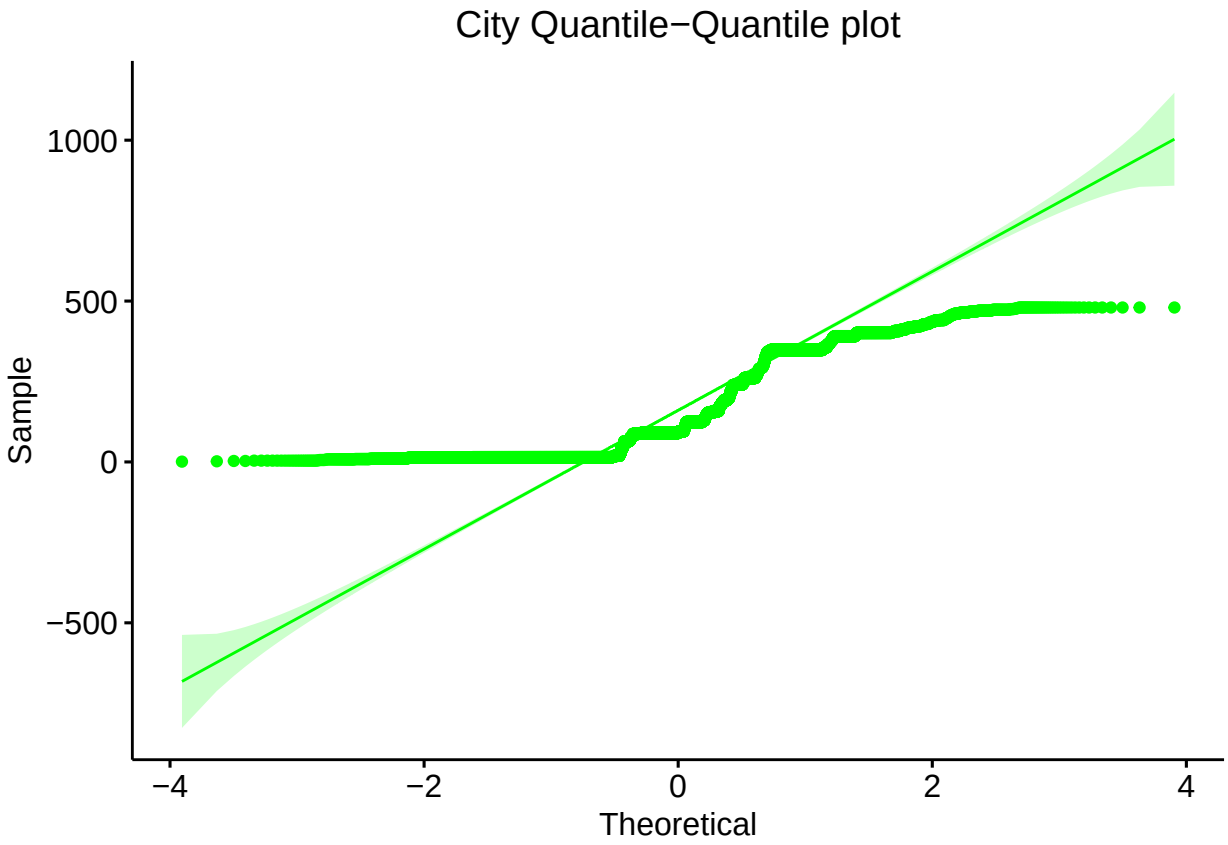

```
ggqqplot(Agency, title="Agency Quantile-Quantile plot", col="pink" )+
  theme(plot.title = element_text(hjust = 0.5))
```




```
city <- c(dataInitial$city)
city.factor <- factor(city)
City <- as.numeric(city.factor)
names(City) <- dataInitial$city

#This histogram provides no useful information as the data doesn't
#seem to follow any distribution
hist(City, breaks=100, main="City histogram", col="green")
```




```
ggqqplot(City, title="City Quantile-Quantile plot", col="green")+
  theme(plot.title = element_text(hjust = 0.5))
```

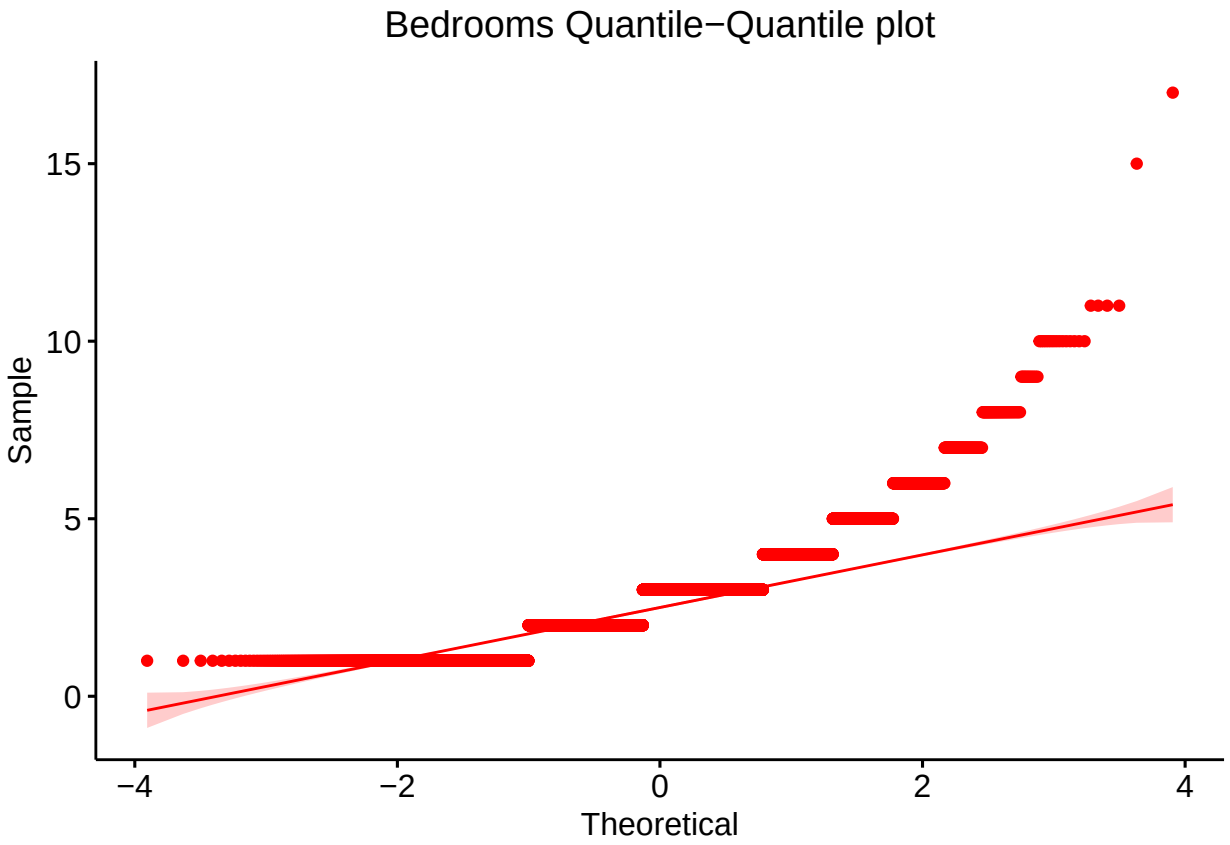



```
mean(dataInitial$bedrooms)
```

```
## [1] 2.770552
```

```
#According to this plot the bedrooms follow a poisson distribution as
#most of the data is close to the mean with few high values
hist(dataInitial$bedrooms,xlim=c(0,17),breaks=18, main="Bedrooms histogram", col="red")
```

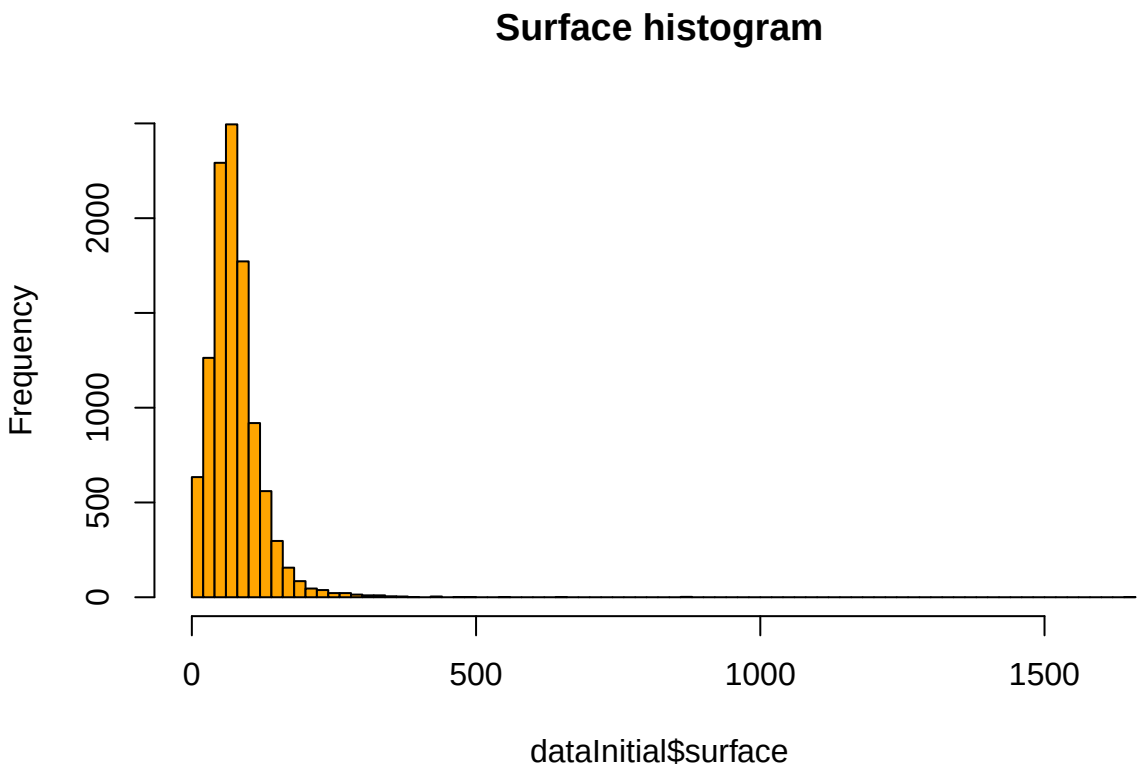

```
ggqqplot(dataInitial$bedrooms, title="Bedrooms Quantile-Quantile plot", col="red")+
  theme(plot.title = element_text(hjust = 0.5))
```



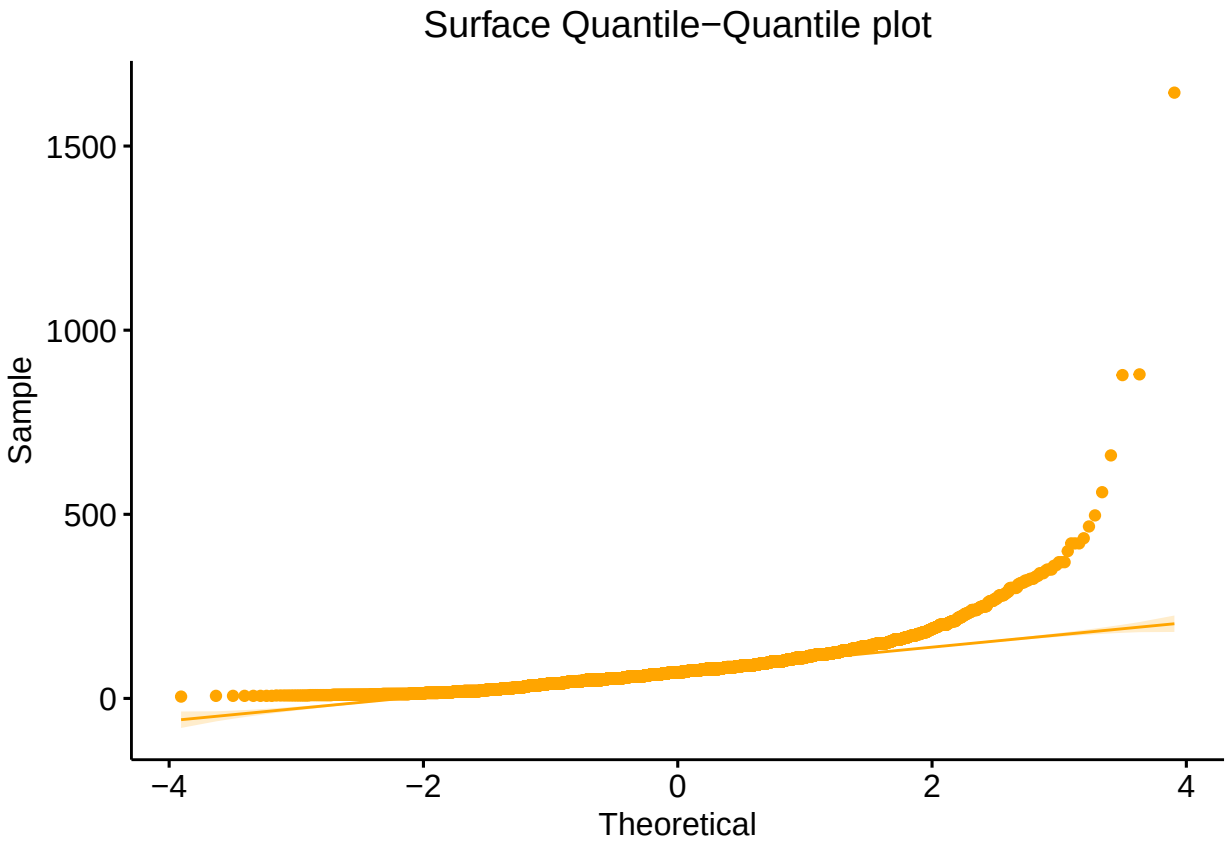

```
mean(dataInitial$surface)
```

```
## [1] 77.38945
```

```
#According to this plot the surface follows a normal distribution
#as its symmetrical about the mean
hist(dataInitial$surface,xlim=c(0,1645),breaks=100, main="Surface histogram", col="orange")
```




```
ggqqplot(dataInitial$surface, title="Surface Quantile-Quantile plot", col="orange")+
  theme(plot.title = element_text(hjust = 0.5))
```



In the above code snippet we aimed to find and understand the distributions in relation to the attributes of the data set. We accomplished this by first finding the mean where applicable and then we plotted a histogram of each attribute of the data set, via the ggqqplot function. We also compared the values in relation to a normal distribution. With the information available to us, we thus concluded that the house_price and surface area had a normal distribution, the bedrooms had a poisson distribution, whilst the rental_agency and city seemed randomly distributed. In light of this it doesn't make sense to use the same distribution for all the data, as the bedrooms column has a different distribution than the house_price and surface area columns.

2.2 (i)

```
#Retrieving the index of all the rows which have an NA
NA_data_Rows <- which(is.na(dataInitial))
print(NA_data_Rows)
```

```
##  [1]    90   504   619   670   727  1278  2340  2613  2935  4717  5043  6319
## [13]  6332  6384  6386  6742  8316  8317  8687  8792  8843  8844  9278  9317
## [25]  9669  9684  9757 10040 10095 10098 10099
```

2.2 (ii)

```
for(i in NA_data_Rows){#Looping through all the rows which have an NA
  cityRows <- vector()#creating all the necessary vectors
  cityRows <- which(dataInitial$city == dataInitial[i, 3])
  prices <- vector()
  surface <- vector()
```

```
  beds <- vector()

  if(length(cityRows) > 2){#If the number of cities present are less than 2 it
    #goes to the else statement to execute the correct regression function
    for (x in cityRows) {
      if (!is.na(dataInitial[x, 1])){#Checks of the value is NA
        prices <- c(prices, dataInitial[x,1])#Storing the relevant values
        surface <- c(surface, dataInitial[x,5])
        beds <- c(beds,dataInitial[x,4])
      }
    }
    relation <- lm(prices ~ surface)#Functions to work out the regression
    #formula and give the resultant answer
    sum <- coef(summary(relation))
    intercept <- sum[1,1]
    gradient <- sum[2,1]

    dataInitial[i,1] <- round((dataInitial[i,5]*gradient)+intercept)
    #Imputing the new house prices in the NA slots
  }
  else{
    for(x in 1:NROW(dataInitial)){
      if(!is.na(dataInitial[x,1]) || x!=i){
        prices <- c(prices, dataInitial[x,1])#Storing the relevant values
        surface <- c(surface, dataInitial[x,5])
        beds <- c(beds,dataInitial[x,4])
      }
    }
    relation <- lm(prices ~ surface)#Functions to work out the
    #regression formula and give the resultant answer
    sum <- coef(summary(relation))
    intercept <- sum[1,1]
    gradient <- sum[2,1]

    dataInitial[i,1] <- round((dataInitial[i,5]*gradient)+intercept)
    #Imputing the new house prices in the NA slots
  }
}
write.csv(dataInitial,"~/Groupproject/Clean.csv")
#Saving the data in another file
```

In the above chunk of code, we aimed to impute accurate values in the NA locations found in the house_price column. We tackled this problem by initially creating the required vectors which would hold the city rows of the relevant cities,as well as prices, surface areas, and amount of beds in separate vectors. Depending on the number of houses present in each city, we executed one of two regression functions, if more than two instances of the same city were present, we retrieved all the required values in relation of that city, which included the prices-surface area and the beds. Then we found the y-intercept and the gradient for the regression function, and through said function we managed to impute correct values. If less than two cities were found we executed the same regression function,but we took the entire data set into consideration excluding any NA values irrelevant of the city. Finally, we saved the cleaned data file and named it Clean.csv.

2.2 (iii)

First we got the rows of the null values,then we looped through all of said rows and we retrieved those which had the same city. Then we checked if there were more than 2 instance of the same city in the city row vector, if this was the case we then retrieved the prices,surface area and amount of beds which had the matching city. Using the lm, summary and coefficient function we managed to obtain the equation of the best fit line, and then we imputed the respective value in the correct location.In certain cases where there wasn't sufficient data to do imputation normally we decided to take prices,surface area and the number of beds for the whole data set irrespective of the city and decided to apply imputation throughout the whole data set as explained above.

2.3 (i)

```
Smpl <- vector()

for (i in  unique(dataInitial$city)){#looping through all the unique cities
  #and getting the respective rows which have said city
  tmp <- vector()
 for(x in which(dataInitial$city == i)){
  tmp <- c(tmp,x)
 }
  amount <- round(length(tmp)/5)#This the the amount of values which we will be taking
  #from every city group, 5 was chosen so that
  #around 2000 instances would be gathered in all
  Smpl <- c(Smpl,sample(tmp,amount))
  #Here we took sample values according to the amount variable
}

VecPrice <- vector()
VecBedrooms <- vector()
VecSurface <- vector()

for (i in Smpl){
  VecPrice <- c(VecPrice,dataInitial[i,1])#Storing the respective data in
  #accordance with the rows held in variable Smpl
  VecBedrooms <- c(VecBedrooms,dataInitial[i,4])
  VecSurface <- c(VecSurface,dataInitial[i,5])
}

#Working out the sample mean in relation to the Price,no. of bedrooms and surface area
MP <- mean(VecPrice)
MB <- mean(VecBedrooms)
MA <- mean(VecSurface)

#Sample data set means:
print(MP)#Printing out the values
```

```
## [1] 1416.742
```

```
print(MB)
```

```
## [1] 2.715931
```

```
print(MA)
```

```
## [1] 75.63148
```

```
#This was used to determine the similarities and in fact they where very similar
#Full data set means:
print(mean(dataInitial$house_price))
```

```
## [1] 1425.569
```

```
print(mean(dataInitial$bedrooms))
```

```
## [1] 2.770552
```

```
print(mean(dataInitial$surface))
```

```
## [1] 77.38945
```

In the above section we aimed to find the means of the surface area, bedrooms, and house_price from a sample. This was achieved by making use of Stratified sampling by dividing the data into groups according to their cities, depending on the number of houses for each city we took a respectable sample such that the sample would still represent the original data. Afterwards we worked out the mean for the prices, bedrooms, and surface area for the sampled data set. At the end we decided to compare the sample means with the relevant means of the entire data set to check our accuracy.

2.3 (ii)

```
APPSM <- vector()

for (i in unique(dataInitial$city)){#Looping through all the unique cities and
  #calculating the mean of the house prices and surface area
  tmp <- which(dataInitial$city==i)

  Hprice <- c(dataInitial[tmp,1])#Storing the respective data in vectors
  Surface <- c(dataInitial[tmp,5])

  MP <- mean(Hprice)#Working out the mean
  MS <- mean(Surface)

  APPSM <- append(APPSM,(MP/MS)) #Working out and storing the
  #average Price Per Square Meter
}

cities <- unique(dataInitial$city)

loc <- cities[which(APPSM==max(APPSM))]#Retrieving the city name through the
#max and match functions
print(loc)#Printing the respective location of the max value
```

```
## [1] "Beinsdorp"
```

```
print(max(APPSM))#Printing the maximum value
```

```
## [1] 44.04762
```

```
loc2 <- cities[which(APPSM==min(APPSM))]#Retrieving the city name through the
#min and match functions
print(loc2)#Printing the respective location of the min value
```

```
## [1] "Wagenborgen"
```

```
print(min(APPSM))#Printing the minimum value
```

```
## [1] 0.2954545
```

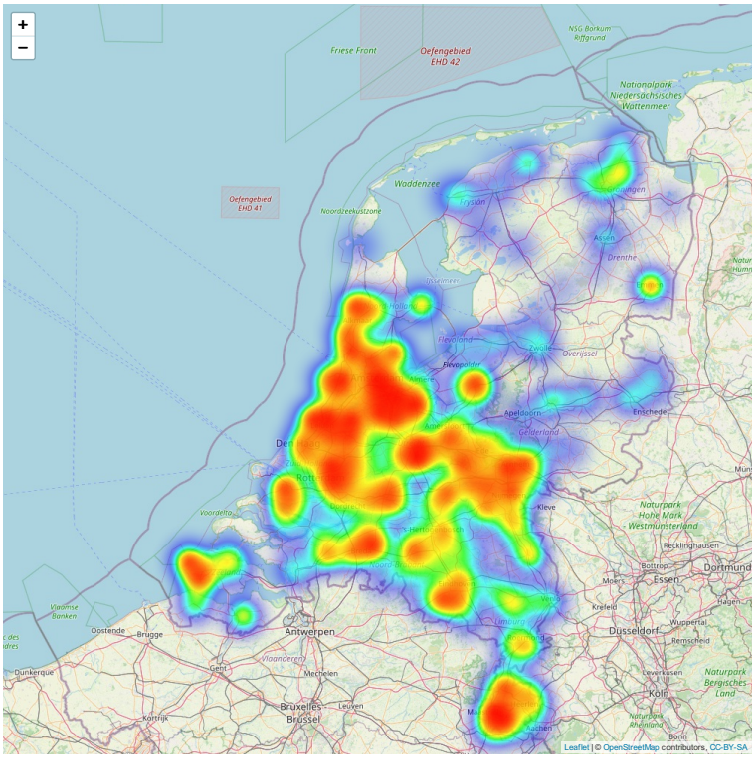
In this section we aimed to find the two cities which had the highest and lowest living cost. To do this we worked out the average price per square meter of each city in the data set and we compared said results to be able to retrieve the maximum and the minimum. According to our findings Beinsdorp is the most expensive city to live in with an average price per square meter of 44.05, whilst Wagenborgen is the cheapest city to live in with an average price per square meter of 0.3.

2.3 (iii)

```
#webshot::install_phantomjs()

#Importing libraries
library(ggplot2)
library(tidyverse)

library(maps)
library(leaflet)
library(leaflet.extras)
```

```
#getting the world.cities database and storing it in and filtering the the
#database to store only the information which has Netherlands as a city

LatLng<-read.csv("~/GroupProject/locationC.csv")

Averages <- vector()
Cities <- vector()
map<-vector()

for (i in unique(dataInitial$city)){#looping through the unique cities

  tmp <- which(dataInitial$city==i)#Storing the rows which have the same city
  Cities <- c(Cities,i)

  Hprice <- c(dataInitial[tmp,1])
  Surface <- c(dataInitial[tmp,5])

  MP <- mean(Hprice)#Working out the mean
  MS <- mean(Surface)
```

```
  Averages <- c(Averages,(MP/MS))
}

LatLng<-cbind(LatLng,"Averages"=NA)#creating new column in v called Averages

for(i in 1:NROW(Cities)){#Matching the Averages to the values
  row = which(LatLng$city == Cities[i])
  LatLng[row,10]<-Averages[i]

}

LatLng <- na.omit(LatLng)#This is used to remove any city data which
#as not relevant to the original dataset

map<-leaflet(LatLng,width="1000px",height="1000px")%>%#These are the commands used to display
  #the heat map with the required markers
  addTiles(group="city")%>%
  setView(LatLng,lng = 4.895168, lat = 52.370216, zoom = 8)%>%
  #The heat map intensity is based on the average price per
  #square meter, used the coordinates from the world.cities database
  addHeatmap(group=dataInitial$house_price, lng=~LatLng$lng, lat=~LatLng$lat,
             intensity=~LatLng$Averages, max=25, blur = 35)#%>%

  #addMarkers(lng=LatLng$lng,lat=LatLng$lat,label=paste(LatLng$city,LatLng$Averages),
             #This adds the markers
  #options = popupOptions(closeButton = TRUE),group="Name/Prices")%>%

  #addLayersControl(
    #overlayGroups= c("Name/Prices"),
    #options = layersControlOptions(collapsed = FALSE))
```


Friese Front
Oefengebied EHD 42
NSG Borkum Riffgrund
Nationalpark Niedersächsisches Wattenmeer
Waddenzee
Oefengebied EHD 41
Noordzeekustzone
Drenthe
Overijssel
Apeldoorn
Enschede
Gelderland
Kleve
Naturpark Hohe Mark Westmünsterland
Recklinghausen
Dortmund
Essen
Krefeld
Wuppertal
Düsseldorf
Remscheid
Leverkusen
Köln
Naturpark Bergisches Land
Naturpark Rheinland
Bonn
Aachen
Antwerpen
Oostende
Brugge
Gent
Mechelen
Leuven
Bruxelles Brussel
Kortrijk
Dunkerque
Vlaamse Banken
Leaflet | © OpenStreetMap contributors, CC-BY-SA

In this section we aimed to create a heatmap to show how the prices per meter squared of area fare across the country. This was achieved by first importing a data set containing the longitude and latitude coordinates for to the Netherlands, then we retrieved the required values which would dictate the intensity of the heatmap, and we mapped all this data onto a map of the Netherlands whcih was created via the leaflet library. Lastly, we also added a togglable set of markers on all relevant locations which when hovered over display the name and average house price of the city.

2.3(iv)

```
Amsterdam <- vector()
Rotterdam <- vector()

AR <- which(dataInitial$city=="Amsterdam")#Contains the index of all
#the rows which have city Amsterdam
RR <- which(dataInitial$city=="Rotterdam")

for(i in AR){
  Amsterdam <- c(Amsterdam, dataInitial[i,1])#Filling the vector with the respective prices
}

for(i in RR){
  Rotterdam <- c(Rotterdam, dataInitial[i,1])
}

SdA <- sd(Amsterdam)#Working out the standard deviation and the mean
MA <- mean(Amsterdam)
SdR <- sd(Rotterdam)
MR <- mean(Rotterdam)

print(SdA)
```

```
## [1] 908.0782
```

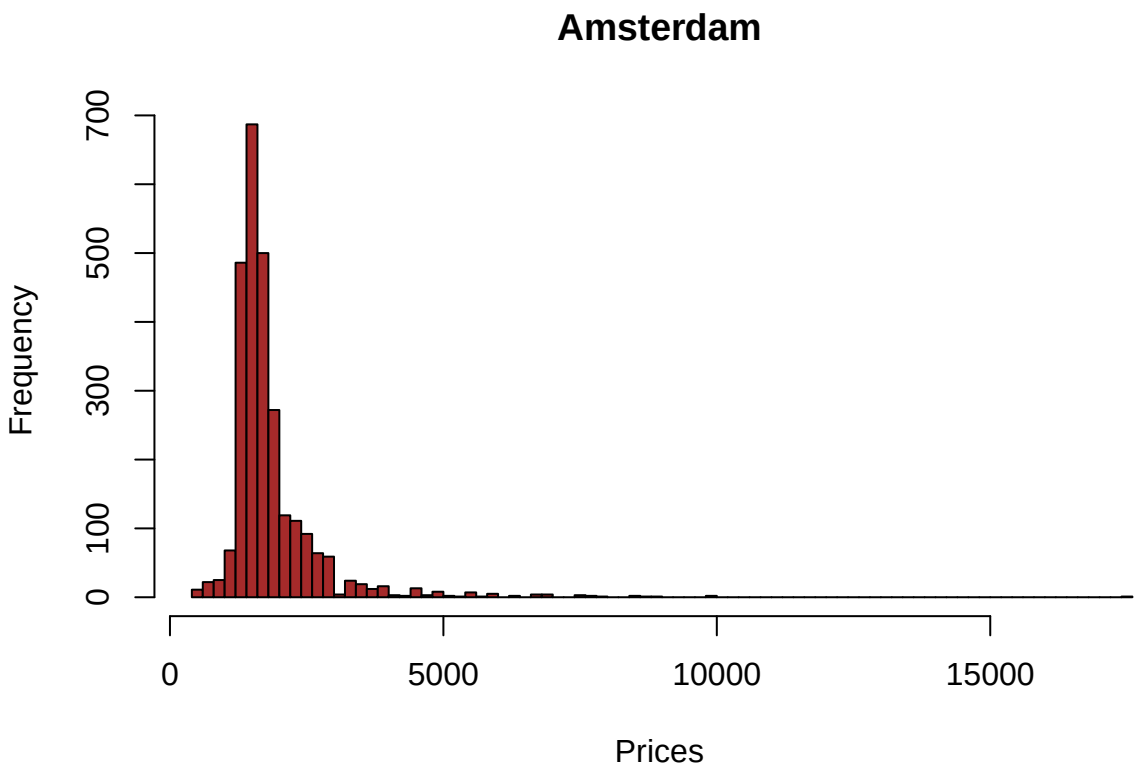
```
print(SdR)
```

```
## [1] 514.3049
```

```
print(MA) #According to this plot Amsterdam follows a normal distribution as its
```

```
## [1] 1866.272
```

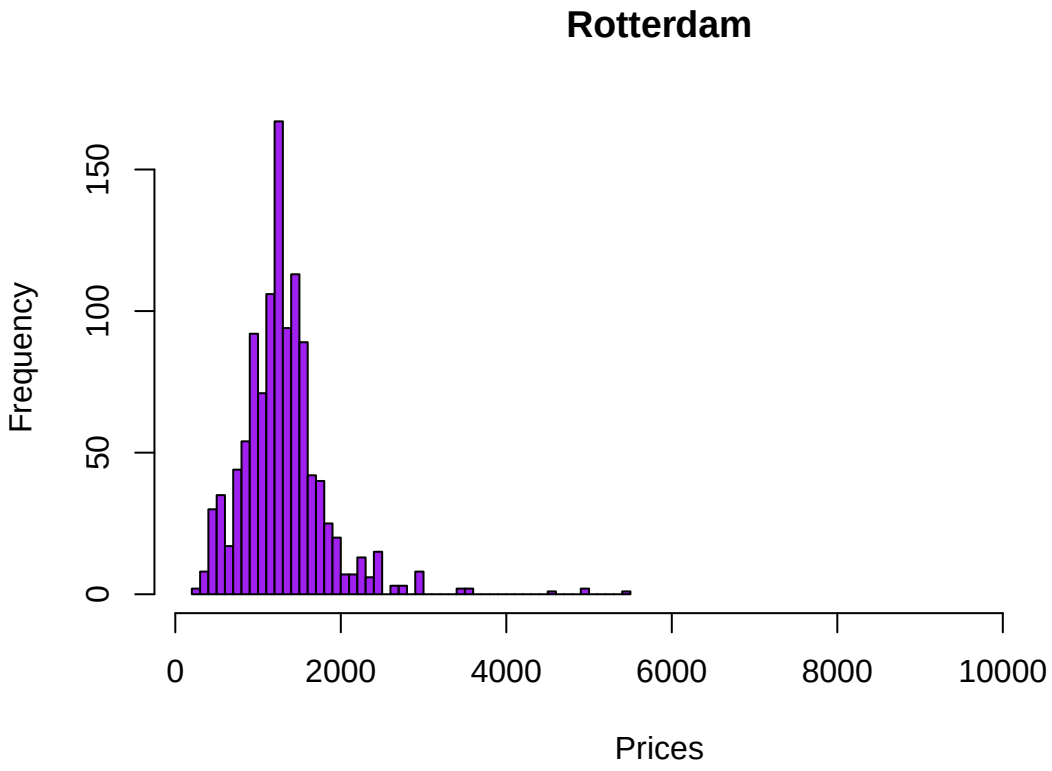
```
#symmetrical about the mean
hist(Amsterdam,breaks = 70,main="Amsterdam",xlab="Prices",col="brown",xlim=c(400,17500))
```



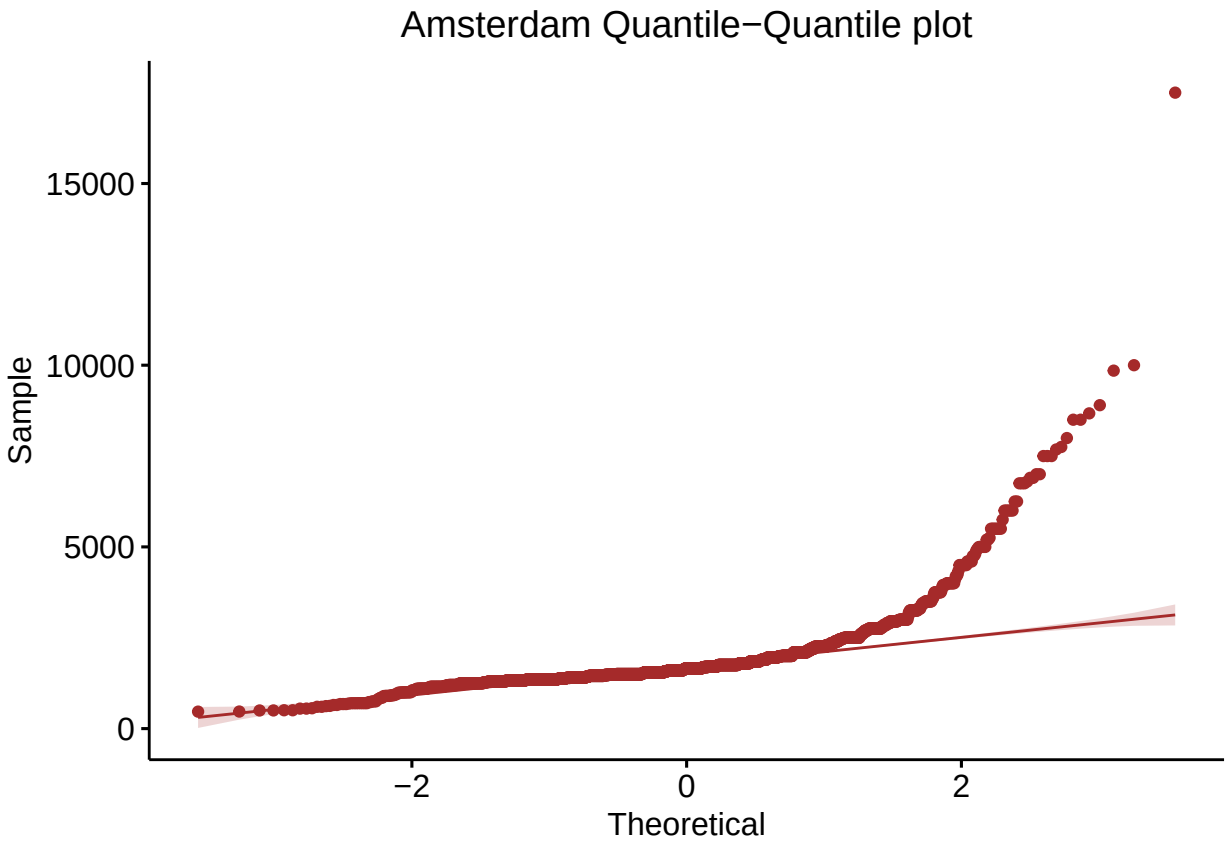

```
print(MR) #According to this plot Rotterdam follows a normal distribution as its
```

```
## [1] 1323.039
```

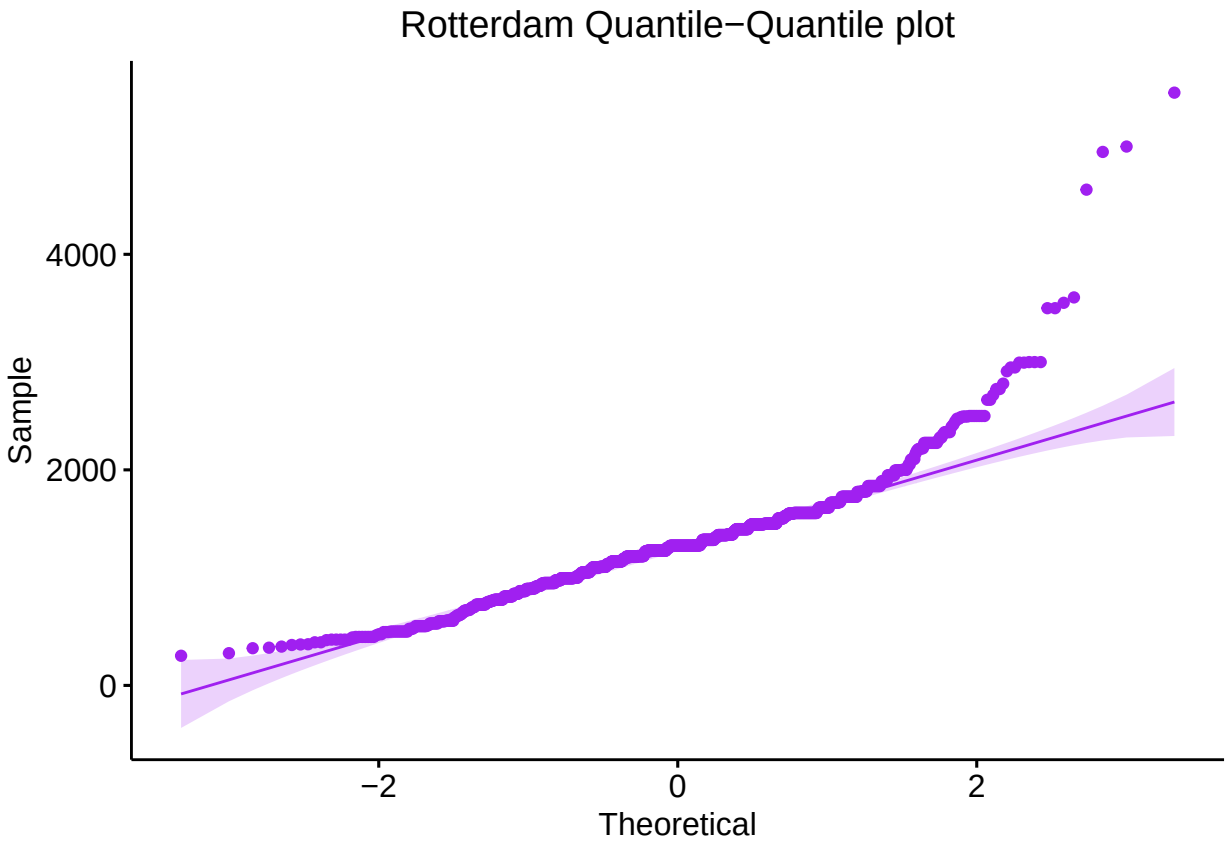
```
#symmetrical about the mean if NA values are ignored
hist(Rotterdam,breaks = 60,main="Rotterdam",xlab="Prices",col="purple",xlim=c(200,11500))
```




```
library(ggpubr)
ggqqplot(Amsterdam, title="Amsterdam Quantile-Quantile plot", col="brown")+
  theme(plot.title = element_text(hjust = 0.5))
```




```
ggqqplot(Rotterdam, title="Rotterdam Quantile-Quantile plot", col="purple")+
  theme(plot.title = element_text(hjust = 0.5))
```




```
#This plots the data give to it as well as a normal distribution line,
#if most of the points are close to this line than
#we have a normal distribution.

#As can be seen via the ggqqplots both graphs are normally
#distributes if the outliers are ignored.
```

The aim of this section was to find out the standard deviation and the distribution of the house prices in Amsterdam and Rotterdam, this was achieved by first retrieving the indexes of those rows, which had buildings located in Amsterdam and Rotterdam respectively. Then we simply worked out the standard deviation of Amsterdam and Rotterdam by using the sd function. Afterwards we plotted two histograms and worked out the means for Amsterdam and Rotterdam, to be able to conclude their distribution. We thus, concluded that for the most part both are normally distributed but both have somewhat heavy tails which can skew the data. This was also supported via the ggqqplot graphs seen above.

2.3(v)

```
AP <- vector()
AS <- vector()
RP <- vector()
RS <- vector()

AR <- which(dataInitial$city=="Amsterdam")#Contains the index of all
#the rows which have city Amsterdam
RR <- which(dataInitial$city=="Rotterdam")#Contains the index of all
#the rows which have city Rotterdam

for(i in AR){
  AP <- c(AP, dataInitial[i,1])#Filling the vector with the respective prices
  AS <- c(AS, dataInitial[i,5])#Filling the vector with the respective surface area
}

for(i in RR){
  RP <- c(RP, dataInitial[i,1])#Filling the vector with the respective prices
  RS <- c(RS, dataInitial[i,5])#Filling the vector with the respective surface area
}

print(paste0("Amsterdam correlation between prices and surface area:"
             ,round(cor(AP,AS),2)))
```

```
## [1] "Amsterdam correlation between prices and surface area:0.82"
```

```
print(paste0("Rotterdam correlation between prices and surface area:"
             ,round(cor(RP,RS),2)))
```

```
## [1] "Rotterdam correlation between prices and surface area:0.79"
```

```
print(paste0("Working out the correlation between the house price and number bedrooms:"
             ,round(cor(dataInitial$house_price, dataInitial$bedrooms),2)))
```

```
## [1] "Working out the correlation between the house price and number bedrooms:0.55"
```

```
print(paste0("Working out the correlation betweent the house price and surface area:"
             ,round(cor(dataInitial$house_price,dataInitial$surface),2)))
```

```
## [1] "Working out the correlation betweent the house price and surface area:0.67"
```

```
print(paste0("Working out the correlation betweent the number of bedrooms and surface area:"
             ,round(cor(dataInitial$bedrooms,dataInitial$surface),2)))
```

```
## [1] "Working out the correlation betweent the number of bedrooms and surface area:0.73"
```

We worked out the correlation between the prices and surface area of Amsterdam and Rotterdam and found out that both have a positive correlation but Amsterdam correlation was stronger. Amsterdam correlation was 0.82 whilst that of Rotterdam was 0.79. This implies that for both cities the surface increases as the price increases and vice-versa.

We worked out the correlation between the house price and the number of bedrooms, as well as the house price and the surface area of the building. We did this as we assumed that the number of bedrooms and the size of the house would in some way correlate to the house price. Even though this was the case it turns out that the correlation of the former was 0.54 whilst that of the latter was 0.66 which is a moderate correlation.

In relation to the correlation between the bedrooms and the surface area we found a strong correlation of 0.73 which implies that a larger house is more likely to have more bedrooms and vice-versa.

2.3(vi)

```
Asurface <- vector()
Abedroom <- vector()
Arent <- vector()

Rsurface <- vector()
Rbedroom <- vector()
Rrent <- vector()

ARows <- which(dataInitial$city=="Amsterdam") #Getting the the number of rows
#which have Amsterdam and Rotterdam as their city and storing them in a vector
RRows <- which(dataInitial$city=="Rotterdam")

for(i in ARows){ #Getting all the prices, bedrooms and surface area for Amsterdam
  Arent <- c(Arent, dataInitial[i,1])
  Abedroom <- c(Abedroom, dataInitial[i,4])
  Asurface <- c(Asurface, dataInitial[i,5])
}

for(i in RRows){
  Rrent <- c(Rrent, dataInitial[i,1])
  Rbedroom <- c(Rbedroom, dataInitial[i,4])
  Rsurface <- c(Rsurface, dataInitial[i,5])
}

relation <- lm(Asurface ~ Abedroom) #These functions work out the intercept
#and gradient for the regression function relating to the
#Surface and Bedrooms in Amsterdam
sum <- coef(summary(relation))
intercept <- sum[1,1]
gradient <- sum[2,1]

relation2 <- lm(Rsurface ~ Rbedroom) #These functions work out the intercept
#and gradient for the regression function relating to the
#Surface and Bedrooms in Rotterdam
sum2 <- coef(summary(relation2))
intercept2 <- sum2[1,1]
gradient2 <- sum2[2,1]

AmsterdamTypicalSurface <- (3*gradient)+intercept #Gives us the expected
#price of a house in Amsterdam with 3 bedrooms
RotterdamTypicalSurface <- (3*gradient2)+intercept2  #Gives us the expected
#price of a house in Rotterdam with 3 bedrooms

relation <- lm(Arent ~ Asurface) #These functions work out the intercept and
#gradient for the regression function relating to the Rent and Surface area in Amsterdam
sum <- coef(summary(relation))
intercept <- sum[1,1]
gradient <- sum[2,1]

relation2 <- lm(Rrent ~ Rsurface) #These functions work out the intercept and
#gradient for the regression function relating to the Rent
#and Surface area in Rotterdam
```

```r
sum2 <- coef(summary(relation2))
intercept2 <- sum2[1,1]
gradient2 <- sum2[2,1]

AmsterdamTypicalRent <- (125*gradient)+intercept #Working out the rent of a 125
#surface Amsterdam apartment
RotterdamTypicalRent <- (125*gradient2)+intercept2 #Working out the rent of a 125
#surface Rotterdam apartment

print(paste0("Amsterdam Surface: ", AmsterdamTypicalSurface))#Displaying the data
```

```
## [1] "Amsterdam Surface: 83.3721830945159"
```

```r
print(paste0("Rotterdam Surface: ", RotterdamTypicalSurface))
```

```
## [1] "Rotterdam Surface: 83.4424032467666"
```

```r
print(paste0("Amsterdam Rent: ", AmsterdamTypicalRent))
```

```
## [1] "Amsterdam Rent: 2811.23132527746"
```

```r
print(paste0("Rotterdam Rent: ", RotterdamTypicalRent))
```

```
## [1] "Rotterdam Rent: 1885.51004786644"
```

In this section we were requested to predict the typical meter square apartment with 3 bedrooms in Amsterdam and Rotterdam as well as the monthly rent for a 125 meters squared apartment for the same two cities. To do this we opted to make use of a regression function, for which we retrieved the relevant data, and then inputted 3 and 125 to retrieve the corresponding predictions. We manually compared the prediction to the data found in the data set and found that it was conducive.

Division of work

The work was done as a team, which means that we worked together equally on all the tasks as was required