SIM Project 1. Preprocessing

Adrià Casanova Víctor Garcia Zhengyong Ji

November, 19th 2023

Contents

In this work, we will study the data set called "Ames Housing dataset", collected by Dean De Cock for the purpose to analyze the correlation about house prices and different features that describe the house condition, and then to build a regression model that will allows us to predict the sale price.

The data set has two parts, the training part and testing part, with 1460 and 1459 observations each other, and 81 variables (including the id variable).

```r
# Delete any existing object
if(!is.null(dev.list())) dev.off()
rm(list = ls())

library(car)
library(mice)
library(dplyr)
library(missMDA)
library(FactoMineR)
library(chemometrics)
library(DataExplorer)
library(corrplot)
library(DataExplorer)

train = read.csv("train.csv")
test = read.csv("test.csv")

#Create EDA report before any data preparation
#create_report(train, output_format = "pdf_document", output_file = "train.pdf")
#create_report(test, output_format = "pdf_document", output_file = "test.pdf")
```

Data preparation and data cleaning

0. Data preparation and data cleaning

After loading the datasets we defined the types of the variables (categorical, numerical or dates). Some of them required further transformation, based on some assumptions, that are detailed below.

```
Categorical_val = c("MSSubClass","MSZoning","Street","Alley","LotShape","LandContour","Utilities","LotC

Numerical_val = c("LotFrontage","LotArea","MasVnrArea","BsmtFinSF1","BsmtFinSF2","BsmtUnfSF","TotalBsmt

Date_val = c("YearBuilt","YearRemodAdd","GarageYrBlt","MoSold","YrSold")

# Identify variables susceptible to be transformed into categorical
sapply(select(train, Numerical_val), table)
sapply(select(train, Categorical_val), table)
sapply(select(train, Date_val), table)
```

1) Non applicable NaN's: There were 3 variables with an important number of missing (aprox 90%) because the measure was not applicable. This happened, firstly, in PoolArea because the pool area can not be computed for houses without a pool. It was also the case of LowQualFinSF because it is only refered to surfaces finished with low quality, and with BsmtFinSF2, that is only applicable for basement of type 2. Our solution was to define those three variables as binary variables.

```
# As we can see there are an important number of Nan
# PoolArea: 99% missings
length(which(train$PoolArea > 0))/dim(train)[1]*100
```

```
## [1] 0.4794521
```

```
length(which(test$PoolArea > 0))/dim(test)[1]*100
```

```
## [1] 0.4112406
```

```
# LowQualFinSF: 98% missings
length(which(train$LowQualFinSF > 0))/dim(train)[1]*100
```

```
## [1] 1.780822
```

```
length(which(test$LowQualFinSF > 0))/dim(test)[1]*100
```

```
## [1] 0.9595613
```

```
#BsmtFinSF2: 89% missings
length(which(train$BsmtFinSF2 > 0))/dim(train)[1]*100
```

```
## [1] 11.43836
```

```
length(which(test$BsmtFinSF2 > 0))/dim(test)[1]*100
```

```
## [1] 12.33722
```

```
# Under the assumption 1, we transform the variables to binary
test <- test %>%
  mutate(PoolArea = ifelse(PoolArea > 0, "Yes", "No"))
test$PoolArea = as.factor(test$PoolArea)
train <- train %>%
  mutate(PoolArea = ifelse(PoolArea > 0, "Yes", "No"))
```

```r
train$PoolArea = as.factor(train$PoolArea)

test <- test %>%
  mutate(LowQualFinSF = ifelse(LowQualFinSF > 0, "Yes", "No"))
test$LowQualFinSF = as.factor(test$LowQualFinSF)
train <- train %>%
  mutate(LowQualFinSF = ifelse(LowQualFinSF > 0, "Yes", "No"))
train$LowQualFinSF = as.factor(train$LowQualFinSF)

test <- test %>%
  mutate(BsmtFinSF2 = ifelse(BsmtFinSF2 > 0, "Yes", "No"))
test$BsmtFinSF2 = as.factor(test$BsmtFinSF2)
train <- train %>%
  mutate(BsmtFinSF2 = ifelse(BsmtFinSF2 > 0, "Yes", "No"))
train$BsmtFinSF2 = as.factor(train$BsmtFinSF2)
```

2) LotFrontage, which represents the distance from the property to the street, has a high percentage of missing values, 18% in "train" and 16% in "test". A quick look at the summary in both datasets shows there is not any house with a value of 0 for this variable. However, in the real world there exist houses whose entrance is right next to the street, with no separation from it. Hence, we deduce that missing values correspond to a distance of 0 and we impute LotFrontage like so.

```r
#Analysis of the percentage of missings
percent_miss <- function(data) {
  return (length(which(is.na(data)))/length(data)*100)
}
percent_miss(train$LotFrontage)
```

```
## [1] 17.73973
```

```r
percent_miss(test$LotFrontage)
```

```
## [1] 15.5586
```

```r
# Transformation Na'n to 0
lltrain <- which(is.na(train$LotFrontage))
lltest <- which(is.na(test$LotFrontage))
train$LotFrontage[lltrain] <- 0
test$LotFrontage[lltest] <- 0
```

3) Only few values possible: Variables BsmtHalfBath KitchenAbvGr have only 3 and 4 values possible, so we transform them into categorical

```r
# BsmtHalfBath is numerical but it can only be 0, 1 or 2
length(which(train$BsmtHalfBath > 0))/dim(train)[1]*100
```

```
## [1] 5.616438
```

```r
length(which(test$BsmtHalfBath > 0))/dim(test)[1]*100
```

```
## [1] 6.374229
```

```r
#KitchenAbvGr can only be 0, 1, 2 or 3
length(which(train$KitchenAbvGr != 1))/dim(train)[1]*100
```

```
## [1] 4.657534
```

```r
length(which(test$KitchenAbvGr != 1))/dim(test)[1]*100
```

```
## [1] 4.523646
```

```r
#Transformation into categorical
train$BsmtHalfBath <- as.factor(train$BsmtHalfBath)
test$BsmtHalfBath <- as.factor(test$BsmtHalfBath)

train$KitchenAbvGr <- as.factor(train$KitchenAbvGr)
test$KitchenAbvGr <- as.factor(test$KitchenAbvGr)
levels(test$KitchenAbvGr) = c(levels(test$KitchenAbvGr),"3")
```

4) Variables with too many categories: OverallQual, Neighborhood and MSSubClass have too many levels to study their interactions in the models we will create later. Hence, we aggregate their categories following logical criterias. Even though, these will create a bias in the model, it will allow us to study their effect on the target. That being said, OverallQual will have 5 ordered levels.

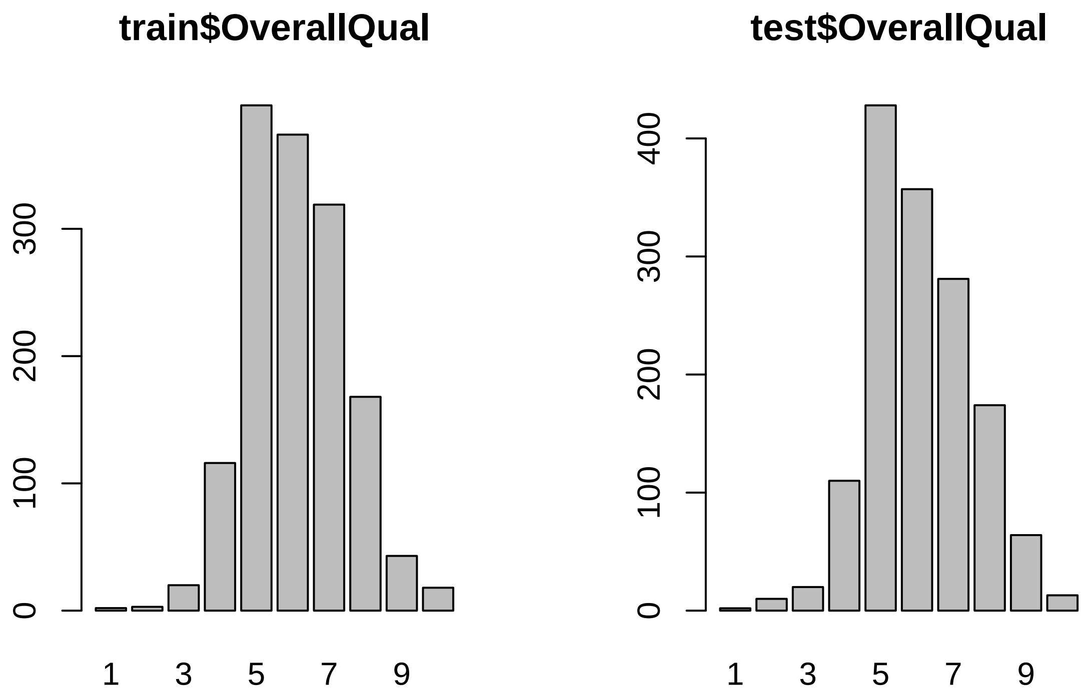
```r
t.train <- table(train$OverallQual); t.train
```

```
##
##   1   2   3   4   5   6   7   8   9  10
##   2   3  20 116 397 374 319 168  43  18
```

```r
t.test <- table(test$OverallQual); t.test
```

```
##
##   1   2   3   4   5   6   7   8   9  10
##   2  10  20 110 428 357 281 174  64  13
```

```r
par(mfrow=c(1,2))
barplot(t.train, main = "train$OverallQual")
barplot(t.test, main = "test$OverallQual")
```




```
par(mfrow=c(1,1))

train$OverallQual <- replace(train$OverallQual, train$OverallQual %in% 1:2, "VBad")
train$OverallQual <- replace(train$OverallQual, train$OverallQual %in% 3:4, "Bad")
train$OverallQual <- replace(train$OverallQual, train$OverallQual %in% 5:6, "Moderate")
train$OverallQual <- replace(train$OverallQual, train$OverallQual %in% 7:8, "Good")
train$OverallQual <- replace(train$OverallQual, train$OverallQual %in% 9:10, "VGood")

test$OverallQual <- replace(test$OverallQual, test$OverallQual %in% 1:2, "VBad")
test$OverallQual <- replace(test$OverallQual, test$OverallQual %in% 3:4, "Bad")
test$OverallQual <- replace(test$OverallQual, test$OverallQual %in% 5:6, "Moderate")
test$OverallQual <- replace(test$OverallQual, test$OverallQual %in% 7:8, "Good")
test$OverallQual <- replace(test$OverallQual, test$OverallQual %in% 9:10, "VGood")

train$OverallQual <- factor(train$OverallQual, levels = c("VBad", "Bad", "Moderate", "Good", "VGood"))
test$OverallQual <- factor(test$OverallQual, levels = c("VBad", "Bad","Moderate", "Good","VGood"))

t.train2 <- table(train$OverallQual); t.train2
```

```
##
##     VBad      Bad Moderate     Good    VGood
##        5      136      771      487       61
```

```
t.test2 <- table(test$OverallQual); t.test2
```

```
##
##     VBad      Bad Moderate     Good    VGood
```

```
##        12       130       785       455        77
```

```r
barplot(t.train2, main = "train$OverallQual")
```


```r
barplot(t.test2, main = "test$OverallQual")
```

Neighborhood will have 3 ordered levels ("Poor", "Moderate" or "Rich") following the real-estate order found in https://www.neighborhoodscout.com/ia/ames/real-estate.

```
t.train <- table(train$Neighborhood); t.train
```

```
##
## Blmngtn Blueste  BrDale BrkSide ClearCr CollgCr Crawfor Edwards Gilbert  IDOTRR
##      17       2      16      58      28     150      51     100      79      37
## MeadowV Mitchel   NAmes NoRidge NPkVill NridgHt  NWAmes OldTown  Sawyer SawyerW
##      17      49     225      41       9      77      73     113      74      59
## Somerst StoneBr   SWISU  Timber Veenker
##      86      25      25      38      11
```

```
t.test <- table(test$Neighborhood); t.test
```

```
##
## Blmngtn Blueste  BrDale BrkSide ClearCr CollgCr Crawfor Edwards Gilbert  IDOTRR
##      11       8      14      50      16     117      52      94      86      56
## MeadowV Mitchel   NAmes NoRidge NPkVill NridgHt  NWAmes OldTown  Sawyer SawyerW
##      20      65     218      30      14      89      58     126      77      66
## Somerst StoneBr   SWISU  Timber Veenker
##      96      26      23      34      13
```

```
Rich = c("NoRidge", "NridgHt", "StoneBr", "Timber", "Veenker", "Somerst", "ClearCr", "Crawfor")
Moderate = c("SWISU", "CollgCr", "Blueste", "Blmngtn", "Gilbert", "Mitchel", "NWAmes", "NPkVill")
Poor = c("Edwards", "BrDale", "BrkSide", "IDOTRR", "MeadowV", "NAmes", "OldTown", "Sawyer", "SawyerW")

train$Neighborhood <- replace(train$Neighborhood, train$Neighborhood %in% Poor, "Poor")
```

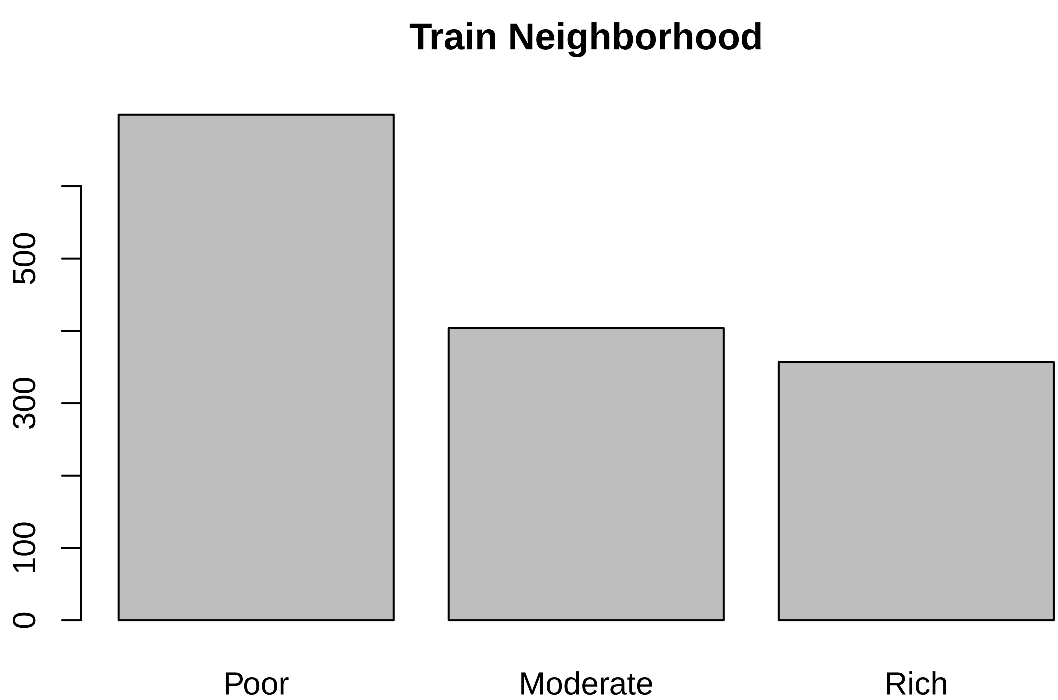
```r
train$Neighborhood <- replace(train$Neighborhood, train$Neighborhood %in% Moderate, "Moderate")
train$Neighborhood <- replace(train$Neighborhood, train$Neighborhood %in% Rich, "Rich")

test$Neighborhood <- replace(test$Neighborhood, test$Neighborhood %in% Poor, "Poor")
test$Neighborhood <- replace(test$Neighborhood, test$Neighborhood %in% Moderate, "Moderate")
test$Neighborhood <- replace(test$Neighborhood, test$Neighborhood %in% Rich, "Rich")

train$Neighborhood <- factor(train$Neighborhood, levels = c("Poor", "Moderate", "Rich"))
test$Neighborhood <- factor(test$Neighborhood, levels = c("Poor", "Moderate", "Rich"))

t.train2 <- table(train$Neighborhood); t.train2
```

```
##
##     Poor Moderate     Rich
##      699      404      357
```

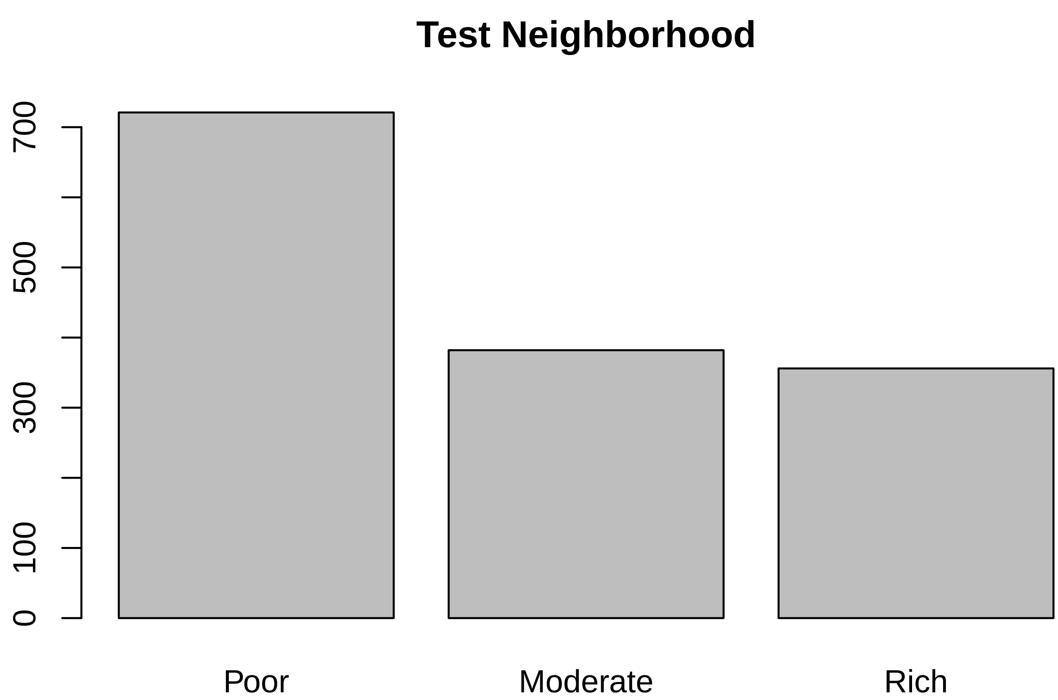
```r
t.test2 <- table(test$Neighborhood); t.test2
```

```
##
##     Poor Moderate     Rich
##      721      382      356
```

```r
barplot(t.train2, main = "Train Neighborhood")
```



```r
barplot(t.test2, main = "Test Neighborhood")
```



4) Non applicable 0's: There are three variables that represent the area of different types of porches (EnclosedPorch, X3SsnPorch and ScreenPorch). In all of them, there is an important percentatge of 0's (about 90%). As a consequence, we consider that it is more efficient to treat those variables as binary to have a more balanced variable and because the univariate analysis of those variables, like outlier detection, of those variables would be very complicated, as their IQR was 0.

```r
# Calculation of the % of non 0's
length(which(train$EnclosedPorch > 0))/dim(train)[1]*100
```

```
## [1] 14.24658
```

```r
length(which(test$EnclosedPorch > 0))/dim(test)[1]*100
```

```
## [1] 17.20356
```

```r
length(which(train$X3SsnPorch > 0))/dim(train)[1]*100
```

```
## [1] 1.643836
```

```r
length(which(test$X3SsnPorch > 0))/dim(test)[1]*100
```

```
## [1] 0.8910212
```

```r
length(which(train$ScreenPorch > 0))/dim(train)[1]*100
```

```
## [1] 7.945205
```

```
length(which(test$ScreenPorch > 0))/dim(test)[1]*100
```

```
## [1] 9.595613
```

```
#Transformation of the variables into binary
test <- test %>%
  mutate(EnclosedPorch = ifelse(EnclosedPorch > 0, "Yes", "No"))
test$EnclosedPorch = as.factor(test$EnclosedPorch)
train <- train %>%
  mutate(EnclosedPorch = ifelse(EnclosedPorch > 0, "Yes", "No"))
train$EnclosedPorch = as.factor(train$EnclosedPorch)

test <- test %>%
  mutate(X3SsnPorch = ifelse(X3SsnPorch > 0, "Yes", "No"))
test$X3SsnPorch = as.factor(test$X3SsnPorch)
train <- train %>%
  mutate(X3SsnPorch = ifelse(X3SsnPorch > 0, "Yes", "No"))
train$X3SsnPorch = as.factor(train$X3SsnPorch)

test <- test %>%
  mutate(ScreenPorch = ifelse(ScreenPorch > 0, "Yes", "No"))
test$ScreenPorch = as.factor(test$ScreenPorch)
train <- train %>%
  mutate(ScreenPorch = ifelse(ScreenPorch > 0, "Yes", "No"))
train$ScreenPorch = as.factor(train$ScreenPorch)
```

5) Redundant variable: MiscVal, that measures the price of a miscellaneous feature (like having an elevator) has a lot of 0's (96%) as it is only applicable for some properties. Moreover, the information of the properties that have a miscellaneous feature can be also optained in "MiscFeature" variable. Consequently, we decided to remove this variable from the analysis.

```
# Analysis of non 0's
length(which(train$MiscVal > 0))/dim(train)[1]*100
```

```
## [1] 3.561644
```

```
length(which(test$MiscVal > 0))/dim(test)[1]*100
```

```
## [1] 3.495545
```

```
miscVal_train <- train$MiscVal
miscVal_test <- test$MiscVal
train$MiscVal <- NULL
test$MiscVal <- NULL
```

6) Creation of a new level for categorical: Because we do not know if all the Nan's in categorical variables are at random we decided that we will not impute any categorical. Consequently, we created a new level for all the missings.

```
# Declaration of a categorical as factor variables with a new level, "Nan"
levels(train$Alley) <- c(levels(train$Alley), "NAlley")
train$Alley[which(is.na(train$Alley))] <- "NAlley"
levels(test$Alley) <- c(levels(test$Alley), "NAlley")
test$Alley[which(is.na(test$Alley))] <- "NAlley"

levels(train$BsmtQual) <- c(levels(train$BsmtQual), "NBsmt")
train$BsmtQual[which(is.na(train$BsmtQual))] <- "NBsmt"
```

```r
levels(test$BsmtQual) <- c(levels(test$BsmtQual), "NBsmt")
test$BsmtQual[which(is.na(test$BsmtQual))] <- "NBsmt"

levels(train$BsmtCond) <- c(levels(train$BsmtCond), "NBsmt")
train$BsmtCond[which(is.na(train$BsmtCond))] <- "NBsmt"
levels(test$BsmtCond) <- c(levels(test$BsmtCond), "NBsmt")
test$BsmtCond[which(is.na(test$BsmtCond))] <- "NBsmt"

levels(train$BsmtExposure) <- c(levels(train$BsmtExposure), "NBsmt")
train$BsmtExposure[which(is.na(train$BsmtExposure))] <- "NBsmt"
levels(test$BsmtExposure) <- c(levels(test$BsmtExposure), "NBsmt")
test$BsmtExposure[which(is.na(test$BsmtExposure))] <- "NBsmt"

levels(train$BsmtFinType1) <- c(levels(train$BsmtFinType1), "NBsmt")
train$BsmtFinType1[which(is.na(train$BsmtFinType1))] <- "NBsmt"
levels(test$BsmtFinType1) <- c(levels(test$BsmtFinType1), "NBsmt")
test$BsmtFinType1[which(is.na(test$BsmtFinType1))] <- "NBsmt"

levels(train$BsmtFinType2) <- c(levels(train$BsmtFinType2), "NBsmt")
train$BsmtFinType2[which(is.na(train$BsmtFinType2))] <- "NBsmt"
levels(test$BsmtFinType2) <- c(levels(test$BsmtFinType2), "NBsmt")
test$BsmtFinType2[which(is.na(test$BsmtFinType2))] <- "NBsmt"

levels(train$FireplaceQu) <- c(levels(train$FireplaceQu), "NFp")
train$FireplaceQu[which(is.na(train$FireplaceQu))] <- "NFp"
levels(test$FireplaceQu) <- c(levels(test$FireplaceQu), "NFp")
test$FireplaceQu[which(is.na(test$FireplaceQu))] <- "NFp"

levels(train$GarageType) <- c(levels(train$GarageType), "NGar")
train$GarageType[which(is.na(train$GarageType))] <- "NGar"
levels(test$GarageType) <- c(levels(test$GarageType), "NGar")
test$GarageType[which(is.na(test$GarageType))] <- "NGar"

levels(train$GarageFinish) <- c(levels(train$GarageFinish), "NGar")
train$GarageFinish[which(is.na(train$GarageFinish))] <- "NGar"
levels(test$GarageFinish) <- c(levels(test$GarageFinish), "NGar")
test$GarageFinish[which(is.na(test$GarageFinish))] <- "NGar"

levels(train$GarageQual) <- c(levels(train$GarageQual), "NGar")
train$GarageQual[which(is.na(train$GarageQual))] <- "NGar"
levels(test$GarageQual) <- c(levels(test$GarageQual), "NGar")
test$GarageQual[which(is.na(test$GarageQual))] <- "NGar"

levels(train$GarageCond) <- c(levels(train$GarageCond), "NGar")
train$GarageCond[which(is.na(train$GarageCond))] <- "NGar"
levels(test$GarageCond) <- c(levels(test$GarageCond), "NGar")
test$GarageCond[which(is.na(test$GarageCond))] <- "NGar"

levels(train$PoolQC) <- c(levels(train$PoolQC), "NPool")
train$PoolQC[which(is.na(train$PoolQC))] <- "NPool"
levels(test) <- c(levels(test$PoolQC), "NPool")
test$PoolQC[which(is.na(test$PoolQC))] <- "NPool"
```

```r
levels(train$Fence) <- c(levels(train$Fence), "NFen")
train$Fence[which(is.na(train$Fence))] <- "NFen"
levels(test$Fence) <- c(levels(test$Fence), "NFen")
test$Fence[which(is.na(test$Fence))] <- "NFen"

levels(train$MiscFeature) <- c(levels(train$MiscFeature), "N")
train$MiscFeature[which(is.na(train$MiscFeature))] <- "N"
levels(test$MiscFeature) <- c(levels(test$MiscFeature), "N")
test$MiscFeature[which(is.na(test$MiscFeature))] <- "N"
```

7) Missing in KitchenQual: there is a single missing value in test$KitchenQual, so we impute it with the mode of the variable, TA.

```r
test$KitchenQual <- replace(test$KitchenQual, is.na(test$KitchenQual), "TA")
```

8) Transformations into categorical: In some variables, like Month, we decided to transform them into categorical as only some values are possible

```r
# Transformation of other variables into categorical
test <- test %>%
  mutate_if(is.character, as.factor)
train <- train %>%
  mutate_if(is.character, as.factor)

test$MSSubClass = as.factor(test$MSSubClass)
test$OverallQual = as.factor(test$OverallQual)
test$OverallCond = as.factor(test$OverallCond)

train$MSSubClass = as.factor(train$MSSubClass)
train$OverallQual = as.factor(train$OverallQual)
train$OverallCond = as.factor(train$OverallCond)

test$MoSold = month.name[test$MoSold]
test$MoSold = as.factor(test$MoSold)
train$MoSold = month.name[train$MoSold]
train$MoSold = as.factor(train$MoSold)
```

9) Correction of errors: we found that “Exterior2nd” has a record of “Brk Cmn”, which does not match with the data description “BrkComm”. So we rename it (in order to match with “Exterior1st”)

```r
names(test)[names(test) == "Brk Cmn"] <- "BrkComm"
```

Lastly, we define the new indexes of all types of variables after transformation.

```r
# Find numerical, categorical and date variables after the imputation
id_num_val = which(sapply(test, is.numeric)==TRUE)

# We won't analyze the id variable
id_num_val = as.numeric(id_num_val)[-1]; id_num_val
```

```
##  [1]  4  5 20 21 27 35 38 39 44 45 47 48 50 51 52 55 57 60 62 63 67 68 77
```

```r
id_cat_val = which(sapply(test, is.factor)==TRUE)
id_cat_val = as.numeric(id_cat_val); id_cat_val
```

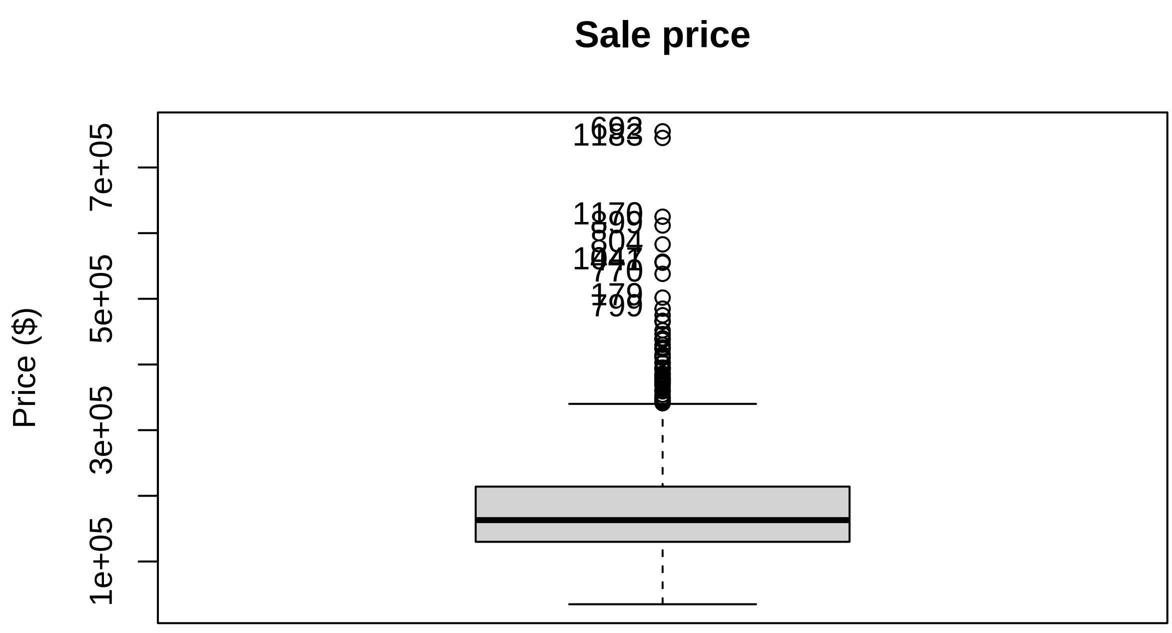
```
##  [1]  2  3  6  7  8  9 10 11 12 13 14 15 16 17 18 19 22 23 24 25 26 28 29 30 31
## [26] 32 33 34 36 37 40 41 42 43 46 49 53 54 56 58 59 61 64 65 66 69 70 71 72 73
## [51] 74 75 76 78 79
```

```r
id_date_val = c(20,21,60,77,78)

# In our datasets, categorical variables are:
Categorical_val = c("MSSubClass","MSZoning","Street","Alley","LotShape","LandContour","Utilities","LotC

# The numerical variables, except the target are
Numerical_val = c("LotFrontage","LotArea","YearBuilt","YearRemodAdd","MasVnrArea","BsmtFinSF1","BsmtUnf

train_num = select(train, Numerical_val)
train_cat = select(train, Categorical_val)
```

1. Univariate outliers detection

First we analysed the target variable, where we found 12 severe outliers as this variable. Because the target variable can not be imputed we decided to remove those observations. You can see all the outliers in the following plot

```r
sevout <- quantile(train$SalePrice,0.75,na.rm=TRUE)+3*(quantile(train$SalePrice,
        0.75,na.rm=TRUE)-quantile(train$SalePrice,0.25,na.rm=TRUE))
target_outlier <- which(train$SalePrice > sevout)

Boxplot(train$SalePrice, main = "Sale price", ylab = "Price ($)")
```




```
##  [1]  692 1183 1170  899  804 1047  441  770  179  799
```

```r
severe_outliers <- function(data) {
  ss <- summary(data)
  # Upper/lower severe thresholds
  utso <- as.numeric(ss[5]+3*(ss[5]-ss[2]))
  ltso <- as.numeric(ss[2]-3*(ss[5]-ss[2]))

  return (which((data>utso)|(data<ltso)))
}
```

Secondly, for all remaining numerical variables (26), we detected outliers and, for severe outliers, we set them to NA to impute them. This process was done automatically with a loop.

```r
# Function to detect outliers
severe_outliers <- function(data) {
  ss <- summary(data)
  # Upper/lower severe thresholds
  utso <- as.numeric(ss[5]+3*(ss[5]-ss[2]))
  ltso <- as.numeric(ss[2]-3*(ss[5]-ss[2]))

  return (which((data>utso)|(data<ltso)))
}

# Set them to NA'n and visualize them
par(mfrow=c(1,2))

for (var in id_num_val) {
  train[severe_outliers(train[,var]),var] <- NA
  Boxplot(train[,var], ylab = names(test)[var], main = "Train")

  test[severe_outliers(test[,var]),var] <- NA
  Boxplot(test[,var], ylab = names(test)[var], main = "Test")
}
```

```r
par(mfrow=c(1,1))

# Remove the outliers for the target variable
train = train[-severe_outliers(train$SalePrice),]
```

2. PCA imputation

Before the detection of outliers there was arround 1% of missing in some numerical variables (see the profiling at the annexes for more datail). After this detection, the variables that contained most missings were "GarageYrBlt" (6% in train and 5% in test), "MasVnrArea" (2% in train and 3% in test), and "OpenPorchSF"(1% in both).

To impute, we assumed that all numerical variables had NA's that were at random and used a PCA to impute both "test" and "train" datasets. As the quartile distributions for all imputed variables are similar, as we can see in the box-plot, we conclude that the imputation was successful for all variables and created a new dataframe with the imputed values. However, for train, we found that for OpenPorchSF feature, there is a negative record. As this is the square feet for open porch area, and it cannot be negative. We suspect that it could be 0, and transformed it.

```r
# Impute
res.PCA = imputePCA (train[,id_num_val])
str (res.PCA)
```

```r
str(res.PCA$completeObs)

res.PCA.test = imputePCA (test[,id_num_val])   # impute numeric variables
str (res.PCA.test)
str(res.PCA.test$completeObs)

# Create a new dataframe
train_impute = data.frame(res.PCA$completeObs)
train_impute$SalePrice <- train$SaleP

test_impute = data.frame(res.PCA.test$completeObs)
```

```r
# Check if the imputation was successful or not: TRAIN
before_imputation = summary(train[,id_num_val])
after_imputation = summary(train_impute)

label = c('Before imputation', 'After imputation')

for (x in c(1,2,5,6,8,9,11,15,16,18,20,21,22)) {
d = data.frame(A = train[,id_num_val][x], B = train_impute[,x])
b = boxplot(d, names=label, main = names(train[,id_num_val][x]));b
}
```

```r
# Transform all negative values of "OpenPorchSF" to 0's
train_impute[which(train_impute$OpenPorchSF < 0),"OpenPorchSF"] = 0
```

```r
# Check if the imputation was successful or not: TEST
before_imputation_test = summary(test[,id_num_val])
after_imputation_test = summary(test_impute)


label = c('Before imputation', 'After imputation')

for (x in c(1,2,5,6,8,9,11,15,16,18,20,21,22)) {
d = data.frame(A = test[,id_num_val][x], B = test_impute[,x])
b = boxplot(d, names=label, main = names(test[,id_num_val][x]));
b
}
```

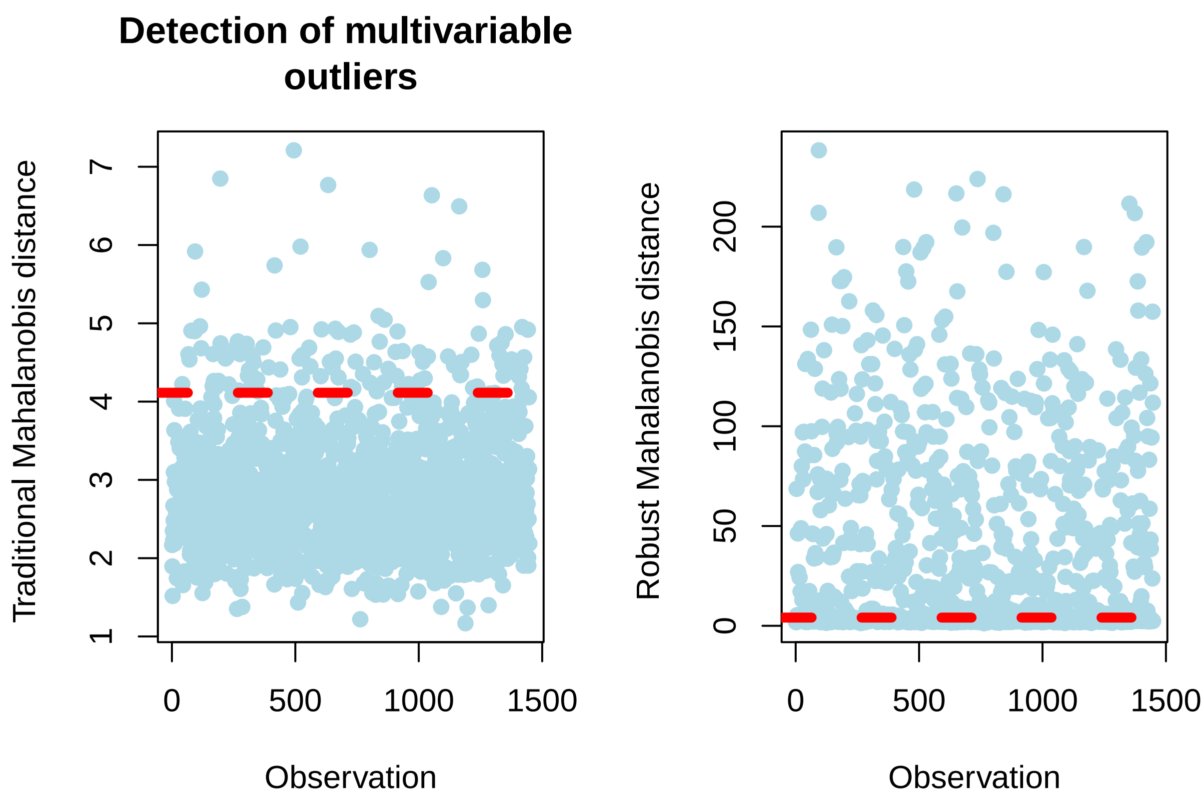
3. Multivariate outliers detection

After the imputation, we decided to perform a Moutlier analysis to detect multivariate outliers. As using all numerical variables returns a singular matrix we decided to make the analysis with only the following variables: "LotFrontage", "LotArea", "YearRemodAdd", "BsmtFinSF1", "BsmtUnfSF", "GrLivArea", "Fireplaces", "GarageYrBlt", "GarageArea".

The analysis showed that there are 112 multivariate outliers in the train dataset and 115 in the test datset.

```r
set.seed(123) #ensure that we always get the same result in Moutlier
# Best combination of variables
id_num_val_not_corr = c(1, 2, 4, 6, 7, 11, 17,
                        18, 20)

# Analysis for train
res.mout <- Moutlier(train_impute[,id_num_val_not_corr], quantile = 0.95, plot= FALSE)
```

```r
par(mfrow=c(1,2))
plot(res.mout$md, col="lightblue", pch = 19, main = 'Detection of multivariable
outliers', xlab= 'Observation',
     ylab ='Traditional Mahalanobis distance ')
abline(h = res.mout$cutoff, col = "red", lwd = 5, lty = 2)

plot(res.mout$rd, col="lightblue", pch = 19, xlab= 'Observation',
     ylab ='Robust Mahalanobis distance ')
abline(h = res.mout$cutoff, col = "red", lwd = 5, lty = 2)
```

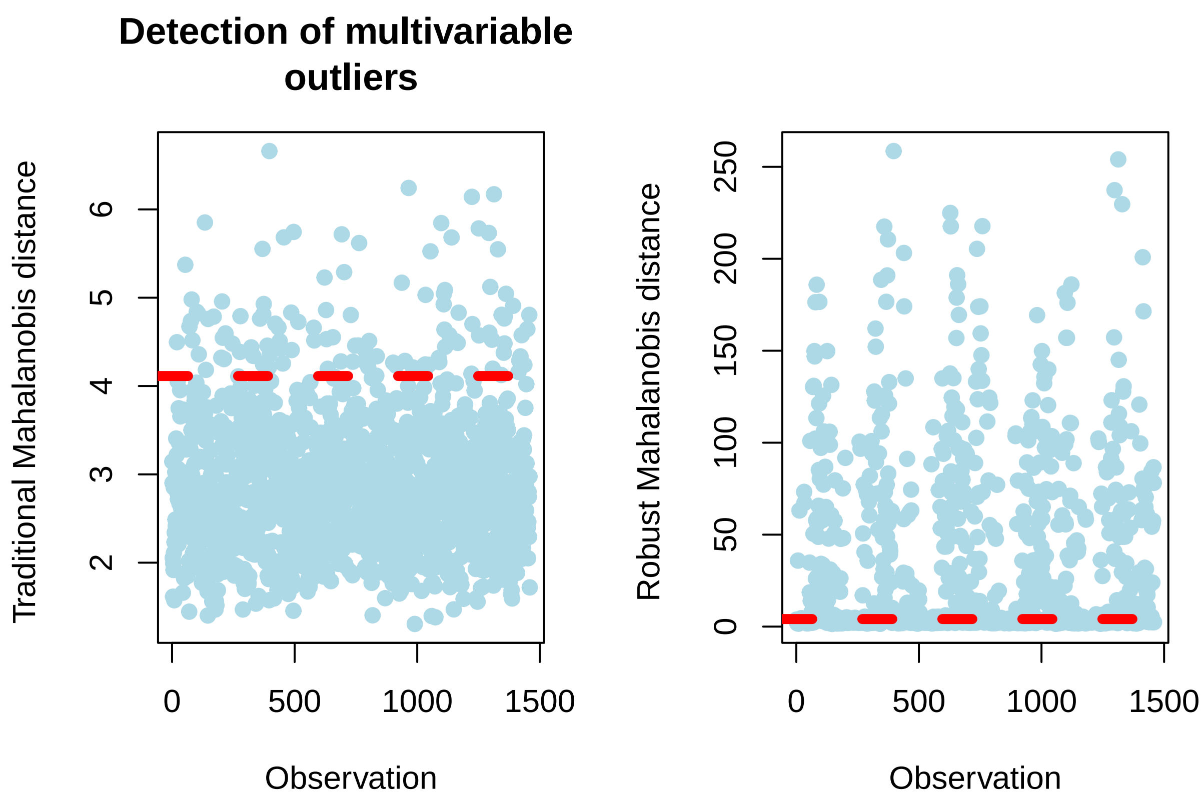



```r
par(mfrow=c(1,1))

outliers = which(res.mout$md>res.mout$cutoff & res.mout$rd > res.mout$cutoff)
length(outliers)
```

```
## [1] 112
```

```r
set.seed(123) #ensure that we always get the same result in Moutlier
# Analysis for test
res.mout.test <- Moutlier(test_impute[,id_num_val_not_corr], quantile = 0.95, plot= FALSE)
par(mfrow=c(1,2))

plot(res.mout.test$md, col="lightblue", pch = 19, main = 'Detection of multivariable
outliers', xlab= 'Observation',
     ylab ='Traditional Mahalanobis distance ')
abline(h = res.mout.test$cutoff, col = "red", lwd = 5, lty = 2)
```

```r
plot(res.mout.test$rd, col="lightblue", pch = 19, xlab= 'Observation',
     ylab ='Robust Mahalanobis distance ')
abline(h = res.mout.test$cutoff, col = "red", lwd = 5, lty = 2)
```




```r
par(mfrow=c(1,1))

outliers.test = which(res.mout.test$md>res.mout.test$cutoff & res.mout.test$rd > res.mout.test$cutoff)
length(outliers.test)
```

```
## [1] 115
```

4. EDA

The last step of the preprocessing was the exploratory data analysis. This step was done automatically using the reports generated with the "SmartEDA" library that you can find in the annexes. The reports were generated considering "train" and "test" files after imputation and just after loading them, without any transformation.

The most relevant conclusions of EDA, considering all numerical values are:

1 - "Train" and "test" datasets contains observations that follows a similar distribution for all variables, numerical and categorical. There are also similarities in the % of missings and all the other summaries.

2 - Both datasets are highly unbalanced in almost all categories. This is specially relevant in variables like "ExterQual" or "Foundation", where only 2 out of 6 categories retains 86% of the accumulative probability.

3 - Numerical variables have a non normal distribution according to Shapiro–Wilk and Kolmogorov-Smirnov

tests. This is specially relevant when modelling as linear models requires normality.

```
# Tests for normality (done in all numerical variables)
ks.test(train$LotArea, y = 'pnorm')
```

```
## Warning in ks.test.default(train$LotArea, y = "pnorm"): ties should not be
## present for the Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  train$LotArea
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
shapiro.test(train$LotArea)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  train$LotArea
## W = 0.97623, p-value = 1.552e-14
```

5. Profiling and selection of categorical features

Once we have the data clean and preprocesed, we have selected the 10 most relevant categories using the profiling of FactoMiner. More precisely, we alinised the relationship between variables in "train" datasets with "SalePrice" and selected the categorical variables with an smaller p-value.

The variables that we selected, sorted starting with the smallest p-value, are: OverallQual, ExterQual, BsmtQual, KitchenQual, Neighborhood, GarageFinish
FireplaceQu, Foundation, GarageType and MSSubClass.

```
# Profiling: selecting only the 10 more significative qualitative variables
res.con = condes(train, 80)
res.con$quali[1:10,]
```

```
##                     R2       p.value
## OverallQual  0.5922373 3.473593e-279
## ExterQual    0.4733253 1.877145e-200
## BsmtQual     0.4649512 3.653937e-194
## KitchenQual  0.4450772 4.390173e-184
## Neighborhood 0.4368561 6.644801e-181
## GarageFinish 0.3343801 4.112106e-127
## FireplaceQu  0.3126242 1.061616e-114
## Foundation   0.2808730 1.248280e-100
## GarageType   0.2788127  1.080007e-98
## MSSubClass   0.2675472  9.455507e-87
```

Additionally, we analysed the correlation of numerical variables with the target. According to the profile all numerical variables have a R^2 of $p < 0.05$ except for "YrSold". Furthermore, we have used cor.test() to test against H0="correlation between"YrSold" and "SalePrice" is 0" and we have failed to reject H0. Therefore, "YrSold" cannot be used to model "SalePrice".

```
res.con$quanti
```

```
##               correlation       p.value
## GrLivArea       0.7070060 1.652301e-219
```

```
## GarageCars     0.6536628 3.170082e-177
## GarageArea     0.6434346 1.114605e-169
## TotalBsmtSF    0.6302457 1.052778e-160
## X1stFlrSF      0.6060534 1.267868e-145
## YearBuilt      0.5567403 1.206933e-118
## FullBath       0.5520825 2.671049e-116
## YearRemodAdd   0.5347647 6.736306e-108
## GarageYrBlt    0.5080399  1.368963e-90
## TotRmsAbvGrd   0.5056546  9.137159e-95
## Fireplaces     0.4615814  2.650203e-77
## MasVnrArea     0.4191751  1.451057e-61
## LotArea        0.4028692  2.198729e-56
## BsmtFinSF1     0.3852252  2.139610e-52
## OpenPorchSF    0.3753766  4.494389e-49
## WoodDeckSF     0.3365277  1.352474e-39
## X2ndFlrSF      0.2874548  6.019348e-29
## HalfBath       0.2788580  2.847591e-27
## BsmtFullBath   0.2428457  7.010476e-21
## BsmtUnfSF      0.2112517  4.526221e-16
## LotFrontage    0.1906714  2.643060e-13
## BedroomAbvGr   0.1655379  2.369043e-10
```

```
res.con$category
```

```
##                           Estimate       p.value
## Neighborhood=Rich       61160.8991 2.441575e-138
## OverallQual=Good        54089.2122 4.946699e-105
## Foundation=PConc        59402.2553 2.598174e-100
## BsmtQual=Ex            130045.6980  4.418471e-98
## ExterQual=Gd            30364.3613  4.981582e-97
## OverallQual=VGood      170648.1442  4.738994e-74
## BsmtFinType1=GLQ        69651.0750  3.764633e-73
## HeatingQC=Ex            65164.1630  6.863248e-72
## KitchenQual=Ex         109779.2104  4.714134e-69
## GarageFinish=Fin        61651.1070  2.226381e-62
## KitchenQual=Gd          22504.5172  8.960956e-55
## ExterQual=Ex           133752.1733  7.713440e-53
## MSSubClass=60           78999.3606  7.784113e-51
## GarageType=Attchd       42590.3830  3.455320e-50
## SaleType=New            78709.9357  2.226052e-45
## SaleCondition=Partial   94873.1064  5.937807e-44
## FireplaceQu=Gd          23101.2176  6.952567e-43
## Exterior2nd=VinylSd     40364.6034  6.878857e-40
## Exterior1st=VinylSd     45157.2362  1.076732e-39
## OverallCond=5           54990.9107  2.015749e-39
## MasVnrType=Stone        53500.0334  2.596110e-35
## BsmtQual=Gd             28700.1097  3.233346e-31
## GarageCond=TA           48560.0660  8.006433e-31
## GarageQual=TA           42293.5782  6.185226e-26
## CentralAir=Y            38738.0116  6.935080e-26
## BsmtExposure=Gd         58316.8663  9.765665e-26
## MSZoning=RL             40809.3108  7.638070e-24
## Electrical=SBrkr        59135.2833  1.336342e-23
## FireplaceQu=Ex         123004.0355  5.871069e-23
## PavedDrive=Y            39493.4337  9.724193e-22
```

```
## HouseStyle=2Story        44420.6725  9.682246e-21
## GarageType=BuiltIn       84126.8371  9.041628e-18
## MasVnrType=BrkFace        2988.2456  2.521668e-17
## GarageFinish=RFn         31497.0521  1.800349e-15
## RoofStyle=Hip            18581.8909  4.478283e-14
## Fence=NFen               28792.4199  1.029037e-13
## EnclosedPorch=No         18954.8647  5.221018e-13
## FireplaceQu=TA            5197.2187  8.802030e-12
## OverallCond=6             7894.4137  9.442320e-11
## BsmtExposure=Av          21562.8581  1.235217e-08
## KitchenAbvGr=1           42964.9224  1.383624e-08
## Exterior2nd=CmentBd      54340.3523  1.728815e-07
## SaleCondition=Normal      7272.1946  2.260609e-07
## BldgType=1Fam            29550.5675  2.676084e-07
## Functional=Typ           39985.2306  3.485115e-07
## HeatingQC=Gd             13032.5213  5.091930e-07
## LotConfig=CulDSac        22705.8282  7.919789e-07
## Alley=NAlley             23254.4422  1.111285e-06
## Exterior1st=CemntBd      56189.8337  1.396892e-06
## LandContour=HLS          38064.2380  1.470497e-06
## BsmtFinType2=Unf         17844.2886  1.725825e-06
## ExterCond=TA             35932.5467  2.091446e-06
## Condition1=Norm           1164.9973  7.023630e-06
## LotShape=IR2             24621.7561  1.322261e-05
## OverallCond=7            14472.8856  1.890786e-05
## MSZoning=FV              67609.1259  2.093545e-05
## BsmtCond=Gd              76583.6139  2.600367e-05
## BsmtFinType1=ALQ          1245.8841  5.002764e-05
## BsmtCond=TA              43002.8424  1.684356e-04
## Heating=GasA             60201.5895  2.026407e-04
## BsmtFinType1=Unf          9647.9015  1.047159e-03
## MSSubClass=120           48761.0164  1.626011e-03
## RoofMatl=WdShngl         77841.5353  4.478674e-03
## Functional=Min2           4015.2535  5.236360e-03
## MiscFeature=N             9744.8994  5.765604e-03
## OverallCond=8            11979.1339  6.685094e-03
## LowQualFinSF=No          19282.4098  6.798186e-03
## ExterCond=Gd             18613.1037  1.188167e-02
## Functional=Min1           6160.0903  1.271527e-02
## MasVnrType=NA            38331.0375  1.818862e-02
## Condition1=PosN          35370.9756  1.976861e-02
## ScreenPorch=Yes           8084.5479  1.998985e-02
## LandContour=Low          16445.8185  2.533396e-02
## X3SsnPorch=Yes           16031.0689  2.748430e-02
## Condition2=PosN         104991.2678  3.175911e-02
## Condition2=PosA         145116.2678  3.698653e-02
## BsmtFinSF2=No             6078.8610  3.699234e-02
## MoSold=September         16320.4257  3.841936e-02
## GarageQual=Gd            74018.3948  4.208020e-02
## MSSubClass=20            30586.2686  4.232762e-02
## RoofMatl=WdShake         41366.5353  4.332869e-02
## Exterior2nd=Other       148780.1998  4.540842e-02
## BsmtExposure=Mn           9267.7962  4.776378e-02
## HouseStyle=1Story        14804.1729  4.976631e-02
```

```
## Condition2=Feedr       -58717.0656  4.974558e-02
## Exterior1st=WdShing    -14147.4453  4.929138e-02
## Electrical=FuseP       -26834.8576  4.874983e-02
## LandSlope=Gtl          -14085.6544  4.198287e-02
## Fence=MnWw             -21266.0958  4.101333e-02
## SaleCondition=AdjLand  -61822.4019  3.714881e-02
## BsmtFinSF2=Yes          -6078.8610  3.699234e-02
## Exterior1st=BrkComm    -93802.5222  3.266487e-02
## BsmtFinType2=BLQ       -12286.8584  2.895393e-02
## X3SsnPorch=No          -16031.0689  2.748430e-02
## BsmtCond=Po            -73016.2938  2.280883e-02
## Exterior2nd=Wd Shng    -19071.1516  2.077500e-02
## ScreenPorch=No          -8084.5479  1.998985e-02
## RoofMatl=CompShg       -22947.1108  1.788889e-02
## Heating=Wall           -26391.0369  1.528328e-02
## Neighborhood=Moderate   -6079.8252  1.136311e-02
## GarageCond=Po          -27292.2757  9.410360e-03
## MSZoning=RH            -14846.5607  8.658802e-03
## MiscFeature=Shed       -17733.0157  7.608460e-03
## LotConfig=Inside       -15682.3548  7.048111e-03
## LowQualFinSF=Yes       -19282.4098  6.798186e-03
## GarageType=CarPort     -47049.6113  3.914085e-03
## Functional=Maj2        -54425.3936  3.569318e-03
## FireplaceQu=Po         -66801.3797  2.251345e-03
## SaleType=COD           -40610.2529  1.490626e-03
## BsmtFinType1=LwQ        -6755.6370  1.245330e-03
## Exterior2nd=HdBoard     -7654.2197  9.524505e-04
## MSSubClass=180         -49718.0641  7.032960e-04
## MSSubClass=45          -43426.3974  6.635215e-04
## PavedDrive=P           -11101.2779  3.772289e-04
## HouseStyle=1.5Unf      -49049.2385  3.214583e-04
## HouseStyle=SFoyer      -24124.7520  1.996344e-04
## MSSubClass=190         -22404.7307  1.634815e-04
## Heating=Grav           -43219.6084  1.181945e-04
## BldgType=2fmCon        -23916.4601  8.552320e-05
## BldgType=Twnhs         -16437.0902  8.133216e-05
## Fence=GdWo             -15173.1446  7.532095e-05
## LandContour=Bnk        -44111.2132  6.996114e-05
## OverallQual=VBad      -124992.5158  5.837516e-05
## Exterior2nd=AsbShng    -56159.2502  4.873868e-05
## Exterior1st=HdBoard     -6515.2767  9.567480e-06
## Exterior1st=AsbShng    -57416.9722  7.097700e-06
## MSSubClass=160         -13370.6831  7.032856e-06
## BldgType=Duplex        -18807.6412  4.255512e-06
## MSSubClass=90          -18476.9871  4.255512e-06
## Condition1=Feedr       -37337.7534  3.738023e-06
## MSZoning=C (all)       -71876.9357  3.399173e-06
## ExterQual=Fa          -110236.3169  1.695220e-06
## Condition1=Artery      -51953.6604  8.277135e-07
## Foundation=Slab        -52259.7737  8.173159e-07
## BsmtFinType1=BLQ        -9114.6843  2.793634e-07
## Electrical=FuseF       -16492.7465  1.834742e-07
## BsmtQual=Fa            -57752.8860  1.341087e-07
## BsmtFinType1=Rec       -11719.0916  1.216843e-07
```

```
## GarageCond=Fa          -21138.2471  8.214616e-08
## BsmtCond=Fa            -15206.7605  6.323184e-08
## GarageQual=Fa          -18268.9653  6.086791e-08
## OverallCond=3          -41743.2022  5.672341e-08
## HeatingQC=Fa           -19906.8603  5.342500e-08
## KitchenAbvGr=2          -5924.2049  4.766703e-08
## SaleCondition=Abnorml  -25405.5119  4.553221e-08
## Alley=Grvl             -34517.9738  1.401731e-08
## ExterCond=Fa           -42396.2326  1.171309e-08
## BsmtFinType2=NBsmt     -53041.6216  2.189547e-09
## Fence=MnPrv            -10623.4787  7.281491e-10
## MSSubClass=50           -8715.0918  6.254255e-10
## BsmtExposure=NBsmt     -73086.0829  5.648703e-10
## OverallCond=4          -23234.1636  3.488793e-10
## Exterior2nd=MetalSd    -20416.6273  3.289469e-10
## BsmtFinType1=NBsmt     -52955.4478  2.654437e-10
## BsmtCond=NBsmt         -31363.4019  2.654437e-10
## BsmtQual=NBsmt         -67792.0227  2.654437e-10
## HouseStyle=1.5Fin      -16082.4982  1.058714e-10
## Exterior1st=MetalSd    -15380.3449  9.316776e-11
## KitchenQual=Fa         -83340.5170  7.959364e-11
## Exterior2nd=Wd Sdng    -23500.1318  3.197052e-11
## Exterior1st=Wd Sdng    -19472.2330  1.194769e-12
## EnclosedPorch=Yes      -18954.8647  5.221018e-13
## RoofStyle=Gable        -16985.7423  4.242368e-13
## Electrical=FuseA        -1971.2973  1.872662e-15
## Foundation=BrkTil      -29697.8332  3.966152e-18
## PavedDrive=N           -28392.1557  1.486338e-18
## LotShape=IR1             -787.1697  9.468147e-19
## SaleType=WD            -13519.5191  1.128697e-20
## BsmtExposure=No        -16061.4377  7.990210e-22
## GarageCond=NGar        -32474.9918  4.028266e-23
## GarageQual=NGar        -38525.0355  4.028266e-23
## GarageFinish=NGar      -65718.1164  4.028266e-23
## GarageType=NGar        -53694.4385  4.028266e-23
## MSSubClass=30          -56188.3394  1.231412e-23
## LotShape=Reg           -38334.6887  1.077595e-25
## CentralAir=N           -38738.0116  6.935080e-26
## MSZoning=RM            -21694.9403  1.226925e-34
## HeatingQC=TA            -1463.4739  1.201358e-36
## OverallQual=Bad        -70774.4879  5.073439e-38
## Foundation=CBlock       -9819.6842  1.870320e-42
## GarageType=Detchd      -23803.7431  9.407607e-51
## MasVnrType=None        -42984.1707  1.162397e-52
## GarageFinish=Unf       -27430.0427  8.718595e-67
## OverallQual=Moderate   -28970.3527  4.847193e-77
## BsmtQual=TA            -33200.8991  5.575320e-83
## FireplaceQu=NFp        -55234.0471  1.589343e-88
## KitchenQual=TA         -48943.2106 3.405426e-111
## Neighborhood=Poor      -55081.0740 2.211351e-123
## ExterQual=TA           -53880.2177 4.220849e-148
```

```r
# Test the correlation between the target and YrSold
cor.test(train$YrSold, train$SalePrice)
```

```
##
##  Pearson's product-moment correlation
##
## data:  train$YrSold and train$SalePrice
## t = -1.3048, df = 1446, p-value = 0.1922
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.08565483  0.01725336
## sample estimates:
##         cor
## -0.03429163
```

6. Analysis of correlation of numerical variables

Using the basic profiling of Factominer we discover that the most correlated numerical variables with the target, with more than 50 % of R^2 are: GrLivArea, GarageCars, GarageArea, TotalBsmtSF, X1stFlrSF, YearBuilt, FullBath, YearRemodAdd, GarageYrBlt and TotRmsAbvGrd.

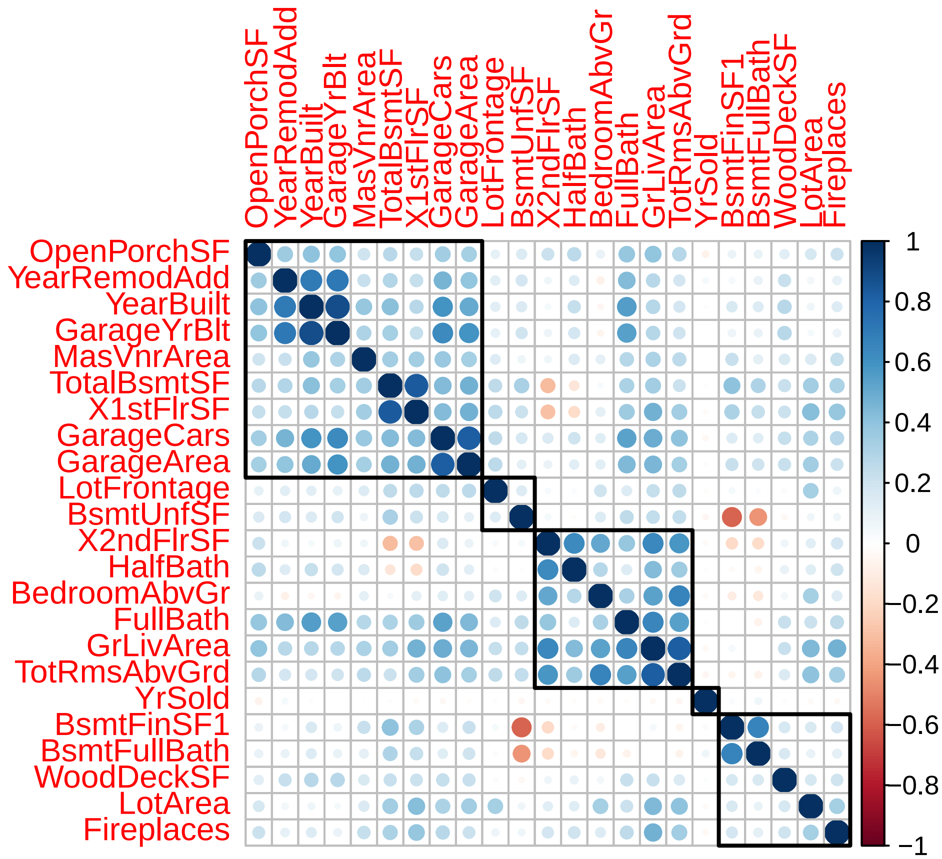
```
res.con = condes(train, 80)
res.con$quanti
```

```
##              correlation       p.value
## GrLivArea      0.7070060 1.652301e-219
## GarageCars     0.6536628 3.170082e-177
## GarageArea     0.6434346 1.114605e-169
## TotalBsmtSF    0.6302457 1.052778e-160
## X1stFlrSF      0.6060534 1.267868e-145
## YearBuilt      0.5567403 1.206933e-118
## FullBath       0.5520825 2.671049e-116
## YearRemodAdd   0.5347647 6.736306e-108
## GarageYrBlt    0.5080399  1.368963e-90
## TotRmsAbvGrd   0.5056546  9.137159e-95
## Fireplaces     0.4615814  2.650203e-77
## MasVnrArea     0.4191751  1.451057e-61
## LotArea        0.4028692  2.198729e-56
## BsmtFinSF1     0.3852252  2.139610e-52
## OpenPorchSF    0.3753766  4.494389e-49
## WoodDeckSF     0.3365277  1.352474e-39
## X2ndFlrSF      0.2874548  6.019348e-29
## HalfBath       0.2788580  2.847591e-27
## BsmtFullBath   0.2428457  7.010476e-21
## BsmtUnfSF      0.2112517  4.526221e-16
## LotFrontage    0.1906714  2.643060e-13
## BedroomAbvGr   0.1655379  2.369043e-10
```

As variables are not normally distributed, we created the correlation matrix of all numerical variables using spearman. The result is ploted in a correlation plot, where we performed a cluster analysis to sort the variables, so that variables that are more correlated are placed closer to each other. Additionally, we decided to create 5 clusters, as we do not expect to work with a model with more than 5 numerical variables. Also, note that in this plot the target variable is not included as this analysis was already done.

The interpretation of this plot suggest that positive correlations are more common than negative, where the most important is between BsmtFullBath and BsmtFin with BsmtUnfSF. Also, there are some important positve correlarions that must be considered when making the model, for example, GarageArea is hightly correlated with GarageCars, so both variables should not be included in the same model.

```r
# Calculate the correlation matrix and then plot it
corr_mat = cor(train_num, method = 'spearman', use = "complete.obs")

corrplot(corr_mat, order = 'hclust', addrect = 5)
```



7. Preparation of data for modelling

The last step of the preprocessing was to create a new file with all the variables that we will use to make our model. To do so, we added the 10 categorical variables to the imputed dataframe. The same process was done with "test" to predict the target variable using the model that we will create.

```r
train_impute$OverallQual <- train$OverallQual
train_impute$Neighborhood <- train$Neighborhood
train_impute$ExterQual <- train$ExterQual
train_impute$BsmtQual <- train$BsmtQual
train_impute$KitchenQual <- train$KitchenQual
train_impute$GarageFinish <- train$GarageFinish
train_impute$FireplaceQu <- train$FireplaceQu
train_impute$Foundation <- train$Foundation
train_impute$GarageType <- train$GarageType
train_impute$MSSubClass <- train$MSSubClass
train_impute$YrSold <- NULL

write.csv(train_impute, file='train_impute.csv', row.names = FALSE)
```

```r
test_impute$OverallQual <- test$OverallQual
test_impute$Neighborhood <- test$Neighborhood
test_impute$ExterQual <- test$ExterQual
test_impute$BsmtQual <- test$BsmtQual
test_impute$KitchenQual <- test$KitchenQual
test_impute$GarageFinish <- test$GarageFinish
test_impute$FireplaceQu <- test$FireplaceQu
test_impute$Foundation <- test$Foundation
test_impute$GarageType <- test$GarageType
test_impute$MSSubClass <- test$MSSubClass
test_impute$YrSold <- NULL

write.csv(test_impute, file='test_impute.csv', row.names = FALSE)
```