MDA 9159 - Statistical Modelling 1 - Fall 2024

Group - Team Bits: Zhangju Xi, Pengwei Xu, Lingyu Zhao

Loading Date and Data Exploratory Analysis

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Read in dataset from github
data = read.csv("https://raw.githubusercontent.com/Panta-Rhei-LZ/MDA_9159_Team_Bits_Project/refs/heads/r

# Remove price=0 entries
data = data[data$PRICE != 0, ]

# Remove rows with NA in all columns except 'YR_RMDL'
data = data %>% filter(!if_any(-YR_RMDL, is.na))

# Remove not useful columns
data = data %>% select(-SSL, -OBJECTID, -GIS_LAST_MOD_DTTM,
                       -QUALIFIED, -SALE_NUM, -BLDG_NUM,
                       -STYLE_D, -STRUCT_D, -GRADE_D,
                       -CNDTN_D, -EXTWALL_D, -ROOF_D,
                       -INTWALL_D, -USECODE, -HEAT_D,
                       -NUM_UNITS, -STRUCT)

head(data)
```

```
##   BATHRM HF_BATHRM HEAT AC ROOMS BEDRM  AYB YR_RMDL  EYB STORIES
## 1      4         1    8  Y    12     6 1911    2021 1989    3.75
## 2      3         1    1  Y    13     5 1912    2009 1978    3.00
## 3      3         1    7  Y     6     4 1910    2022 1993    3.00
## 4      3         1    7  Y    11     4 1912    2000 1978    3.00
## 5      4         1    1  Y    11     5 1912    2007 1993    3.00
## 6      7         1    8  Y    16     7 1895    2014 1993    3.00
##                  SALEDATE   PRICE  GBA STYLE GRADE CNDTN EXTWALL ROOF INTWALL
## 1 2019/08/19 04:00:00+00 3275000 6765    10     8     4      20   11       6
## 2 1999/08/04 04:00:00+00  550000 2282     7     6     4      14    2       6
## 3 2019/07/22 04:00:00+00 1700000 2016     7     6     4      14    6       6
## 4 2021/10/27 04:00:00+00 1500000 2034     7     6     4      14    6       6
## 5 2023/04/18 04:00:00+00 2232500 2655     7     6     5      14    2       6
## 6 2013/12/30 05:00:00+00 1320000 2894     7     6     5      14    6       6
##   KITCHENS FIREPLACES LANDAREA
## 1        1          6     2104
## 2        2          3      936
## 3        2          2      936
## 4        2          2      988
## 5        3          4     1674
## 6        4          1     1674
```

Variable Explanation

We are dealing with housing data in this report, let me go over through the meanings behind each predictor:

1. PRICE: response
2. BATHRM: # bathrooms
3. HF_BATHRM: # half bathrooms
4. HEAT: heating
5. AC: air conditioning
6. ROOMS: # rooms
7. BEDRM: # bedrooms
8. AYB: The earliest time the main portion of the building was built
9. YR_RMDL: Year structure was remodelled
10. EYB: The year an improvement was built
11. STORIES: # stories in primary dwelling
12. SALEDATE: Date of sale
13. GBA: Gross building area in square feet
14. STYLE: House style
15. GRADE: House grade
16. CNDTN: House condition
17. EXTWALL: Exterior wall tyle
18. ROOF: Roof type
19. INTWALL: Interior wall type
20. KITCHENS: # kitchens
21. FIREPLACES: # fireplaces
22. LANDAREA: Land area of property in square feet

NA Data

Now let us explore the percentage of missing data for each predictor:

```
missing_data = round(sapply(data, function(x) mean(is.na(x) * 100)), 3)

missing_data
```

```
##     BATHRM  HF_BATHRM       HEAT         AC      ROOMS      BEDRM        AYB
##      0.000      0.000      0.000      0.000      0.000      0.000      0.000
##    YR_RMDL        EYB    STORIES   SALEDATE      PRICE        GBA      STYLE
##     36.432      0.000      0.000      0.000      0.000      0.000      0.000
##      GRADE      CNDTN    EXTWALL       ROOF    INTWALL   KITCHENS FIREPLACES
##      0.000      0.000      0.000      0.000      0.000      0.000      0.000
##   LANDAREA
##      0.000
```

From the R output above, observe that "YR_RMDL: Year structure was remodeled" has around 36% missing data. A possible explanation for this could be: not all buildings were remodeled.

Preprocessing

- Created dummy variables for categorical predictors:
  - These categorical variables include: "HEAT", "STYLE", "GRADE", "CNDTN", "EXTWALL", "ROOF" and "INTWALL".
- Converted some predictors to numerical values:
  - AC: "Y" and "N" corresponds to "1" and "0".
  - SALEDATE: Transform calendar format values in SALEDATE to numerical values using as.Date().

- Introduced a few new variables:
  - `SALE_YEAR`: The year that the house was sold, it is derived from `SALEDATE`.
  - `SALE_AYB_DIFF`: The difference between the year sold and the year built.
  - `SALE_EYB_DIFF`: The difference between the year sold and the year an improvement was applied.
  - `SALE_RMDL_DIFF`: The difference between the year sold and the year structure was remodeled.'

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
# Transform Yes/No for having AC to numerical values
data$AC = ifelse(data$AC == 'Y', 1, 0)

# Add SALEYEAR
data$SALE_YEAR = year(ymd_hms(data$SALEDATE))

# Add SALEYEAR and AYB diff
data$SALE_AYB_DIFF = data$SALE_YEAR - data$AYB

# Add SALEYEAR and EYB diff
data$SALE_EYB_DIFF = data$SALE_YEAR - data$EYB

# Add SALEYEAR and YR_RMDL diff
data$SALE_RMDL_DIFF = data$SALE_YEAR - data$YR_RMDL

# Convert SALEDATE column to numeric values
data$SALEDATE = as.numeric(as.Date(data$SALEDATE))
```

```
# Replace NA with column median
data = data.frame(lapply(data, function(column) {
  column_median = median(column, na.rm = TRUE)
  column[is.na(column)] = column_median
  column
}))
```

```
# Define box-cox and inverse box-cox transformation
powerfun = function(y, lambda) {
  if (lambda == 0) {
    return(log(y))
  } else {
    return((y^lambda - 1) / lambda)
  }
}

inv_powerfun = function(y_transformed, lambda) {
  if (lambda == 0) {
    return(exp(y_transformed))
  } else {
    return((lambda * y_transformed + 1)^(1/lambda))
```

```
  }
}
```

Data for Training and Validating

```
set.seed(9159)

# Randomly sample 600 data entries for our project
clean_data = data[sample(nrow(data), 600),]

data_train = clean_data[1:500, ]  # First 500 rows for training
data_valid = clean_data[501:600, ]  # Last 100 rows for validation
```