House Prices

Advanced Regression Techniques

Arno Zeng

2024-10-14

Contents

Preprocessing 1

```
knitr::opts_chunk$set(echo = TRUE)
```

```
library(readr) # read_csv
library(dplyr) # filter, mutate, select, ...
```

```
##
## 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
```

```
library(tidyr) # gather
```

Preprocessing

Import data

```
train <- read_csv("../data/raw/train.csv", col_names = TRUE)
```

```
## Rows: 1460 Columns: 81
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (43): MSZoning, Street, Alley, LotShape, LandContour, Utilities, LotConf...
## dbl (38): Id, MSSubClass, LotFrontage, LotArea, OverallQual, OverallCond, Ye...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
test  <- read_csv("../data/raw/test.csv" , col_names = TRUE)
```

```
## Rows: 1459 Columns: 80
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (43): MSZoning, Street, Alley, LotShape, LandContour, Utilities, LotConf...
## dbl (37): Id, MSSubClass, LotFrontage, LotArea, OverallQual, OverallCond, Ye...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(train)
```

```
## # A tibble: 6 x 81
##      Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape
##   <dbl>      <dbl> <chr>          <dbl>   <dbl> <chr>  <chr> <chr>
## 1     1         60 RL                65    8450 Pave   <NA>  Reg
## 2     2         20 RL                80    9600 Pave   <NA>  Reg
## 3     3         60 RL                68   11250 Pave   <NA>  IR1
## 4     4         70 RL                60    9550 Pave   <NA>  IR1
## 5     5         60 RL                84   14260 Pave   <NA>  IR1
## 6     6         50 RL                85   14115 Pave   <NA>  IR1
## # i 73 more variables: LandContour <chr>, Utilities <chr>, LotConfig <chr>,
## #   LandSlope <chr>, Neighborhood <chr>, Condition1 <chr>, Condition2 <chr>,
## #   BldgType <chr>, HouseStyle <chr>, OverallQual <dbl>, OverallCond <dbl>,
## #   YearBuilt <dbl>, YearRemodAdd <dbl>, RoofStyle <chr>, RoofMatl <chr>,
## #   Exterior1st <chr>, Exterior2nd <chr>, MasVnrType <chr>, MasVnrArea <dbl>,
## #   ExterQual <chr>, ExterCond <chr>, Foundation <chr>, BsmtQual <chr>,
## #   BsmtCond <chr>, BsmtExposure <chr>, BsmtFinType1 <chr>, ...
```

```
head(test)
```

```
## # A tibble: 6 x 80
##      Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape
##   <dbl>      <dbl> <chr>          <dbl>   <dbl> <chr>  <chr> <chr>
## 1  1461         20 RH                80   11622 Pave   <NA>  Reg
## 2  1462         20 RL                81   14267 Pave   <NA>  IR1
## 3  1463         60 RL                74   13830 Pave   <NA>  IR1
## 4  1464         60 RL                78    9978 Pave   <NA>  IR1
## 5  1465        120 RL                43    5005 Pave   <NA>  IR1
## 6  1466         60 RL                75   10000 Pave   <NA>  IR1
## # i 72 more variables: LandContour <chr>, Utilities <chr>, LotConfig <chr>,
## #   LandSlope <chr>, Neighborhood <chr>, Condition1 <chr>, Condition2 <chr>,
## #   BldgType <chr>, HouseStyle <chr>, OverallQual <dbl>, OverallCond <dbl>,
## #   YearBuilt <dbl>, YearRemodAdd <dbl>, RoofStyle <chr>, RoofMatl <chr>,
## #   Exterior1st <chr>, Exterior2nd <chr>, MasVnrType <chr>, MasVnrArea <dbl>,
## #   ExterQual <chr>, ExterCond <chr>, Foundation <chr>, BsmtQual <chr>,
## #   BsmtCond <chr>, BsmtExposure <chr>, BsmtFinType1 <chr>, ...
```

It is observed that the train and test datasets differ by one column, `SalePrice`, which is the target variable we aim to predict.

Based on our initial inspection of the imported train and test datasets, we observe that many columns are categorized as strings. At this point, it is necessary to compare the descriptions of these data to determine whether converting the string variables into factors is appropriate.

By comparing the descriptions provided on kaggle, also in `../data/raw/data_description.txt` for each column, we identified that all columns with string values should indeed be treated as factors. Additionally, we discovered that certain columns marked as double, such as `MSSubClass`, `OverallQual`, and `OverallCond`, should actually be considered as factors based on the nature of their data.

Additionally, the Id column is not required for our analysis, so we will remove it from both the train and test datasets.

```r
train <- train |>
    mutate(across(c(MSSubClass, OverallQual, OverallCond), as.factor)) |>
    mutate(across(where(is.character), as.factor)) |>
    select(-Id) # remove Id

test <- test |>
    mutate(across(c(MSSubClass, OverallQual, OverallCond), as.factor)) |>
    mutate(across(where(is.character), as.factor)) |>
    select(-Id) # remove Id
```

Let's take another look at our data to ensure that we have made the necessary adjustments and understand its structure before proceeding further.

```r
head(train)
```

```
## # A tibble: 6 x 80
##   MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape LandContour
##   <fct>      <fct>          <dbl>   <dbl> <fct>  <fct> <fct>    <fct>
## 1 60         RL                65    8450 Pave   <NA>  Reg      Lvl
## 2 20         RL                80    9600 Pave   <NA>  Reg      Lvl
## 3 60         RL                68   11250 Pave   <NA>  IR1      Lvl
## 4 70         RL                60    9550 Pave   <NA>  IR1      Lvl
## 5 60         RL                84   14260 Pave   <NA>  IR1      Lvl
## 6 50         RL                85   14115 Pave   <NA>  IR1      Lvl
## # i 72 more variables: Utilities <fct>, LotConfig <fct>, LandSlope <fct>,
## #   Neighborhood <fct>, Condition1 <fct>, Condition2 <fct>, BldgType <fct>,
## #   HouseStyle <fct>, OverallQual <fct>, OverallCond <fct>, YearBuilt <dbl>,
## #   YearRemodAdd <dbl>, RoofStyle <fct>, RoofMatl <fct>, Exterior1st <fct>,
## #   Exterior2nd <fct>, MasVnrType <fct>, MasVnrArea <dbl>, ExterQual <fct>,
## #   ExterCond <fct>, Foundation <fct>, BsmtQual <fct>, BsmtCond <fct>,
## #   BsmtExposure <fct>, BsmtFinType1 <fct>, BsmtFinSF1 <dbl>, ...
```

```r
head(test)
```

```
## # A tibble: 6 x 79
##   MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape LandContour
##   <fct>      <fct>          <dbl>   <dbl> <fct>  <fct> <fct>    <fct>
## 1 20         RH                80   11622 Pave   <NA>  Reg      Lvl
## 2 20         RL                81   14267 Pave   <NA>  IR1      Lvl
## 3 60         RL                74   13830 Pave   <NA>  IR1      Lvl
## 4 60         RL                78    9978 Pave   <NA>  IR1      Lvl
## 5 120        RL                43    5005 Pave   <NA>  IR1      HLS
## 6 60         RL                75   10000 Pave   <NA>  IR1      Lvl
## # i 71 more variables: Utilities <fct>, LotConfig <fct>, LandSlope <fct>,
## #   Neighborhood <fct>, Condition1 <fct>, Condition2 <fct>, BldgType <fct>,
## #   HouseStyle <fct>, OverallQual <fct>, OverallCond <fct>, YearBuilt <dbl>,
## #   YearRemodAdd <dbl>, RoofStyle <fct>, RoofMatl <fct>, Exterior1st <fct>,
## #   Exterior2nd <fct>, MasVnrType <fct>, MasVnrArea <dbl>, ExterQual <fct>,
## #   ExterCond <fct>, Foundation <fct>, BsmtQual <fct>, BsmtCond <fct>,
## #   BsmtExposure <fct>, BsmtFinType1 <fct>, BsmtFinSF1 <dbl>, ...
```

At this point, the data import process is complete, and we are ready to proceed with the next steps.

Fill NA

```
# na info for train data set
train_na_info <- train |>
  select(where(~ any(is.na(.)))) |> # select cols with na
  summarise_all(~ class(.)) |>      # mark corresponding data type
  pivot_longer(                     # formatting
    cols = everything(),
    names_to = "ColumnName",
    values_to = "DataType"
  ) |>
  mutate(Source = "train")          # mark source

# na info for test data set
test_na_info <- test |>
  select(where(~ any(is.na(.)))) |> # select cols with na
  summarise_all(~ class(.)) |>      # mark corresponding data type
  pivot_longer(                     # formatting
    cols = everything(),
    names_to = "ColumnName",
    values_to = "DataType"
  ) |>
  mutate(Source = "test")           # mark source

# combined na info for train and test
combined_na_info <- full_join(
    train_na_info, test_na_info,
    by = c("ColumnName", "DataType")
  ) |>
  mutate(Source = case_when(
    !is.na(Source.x) & !is.na(Source.y) ~ "both",  # if this col has na in both
    !is.na(Source.x) ~ "train",                    # train and test, we mark
    !is.na(Source.y) ~ "test"                      # the source as "both"
  )) |>
  select(ColumnName, DataType, Source) |>  # drop col Source.x and Source.y
  distinct()                               # make sure no duplicated rows

# output
combined_na_info
```

```
## # A tibble: 34 x 3
##    ColumnName   DataType Source
##    <chr>        <chr>    <chr>
##  1 LotFrontage  numeric  both
##  2 Alley        factor   both
##  3 MasVnrType   factor   both
##  4 MasVnrArea   numeric  both
##  5 BsmtQual     factor   both
##  6 BsmtCond     factor   both
##  7 BsmtExposure factor   both
##  8 BsmtFinType1 factor   both
##  9 BsmtFinType2 factor   both
## 10 Electrical   factor   train
## # i 24 more rows
```

Outlier Detection and Handling