STAT 151A Project

Predicting Housing Resale Prices in Singapore

Michelle Vuong, Celina Mac, Lewis Chong

April 10th 2024

Introduction

Research Objectives

Our research objective is to gain understanding of the effect COVID-19 had on the Singaporean housing market. We aim to create an efficient and concise model that gives us an answer to the question: how do the resale prices differ between the years of 2017-2019 and 2020-2022? In addition, we will analyze the original set of housing metric indicators provided from the data set given to us from Kaggle and determine if we can refine the number of indicators to a minimum. Making use of statistical analysis techniques such as Ridge and Lasso regularization, we aim to extract the most important predictors that can still effectively provide us an understanding of the resale prices of Singaporean homes.

Data Collection

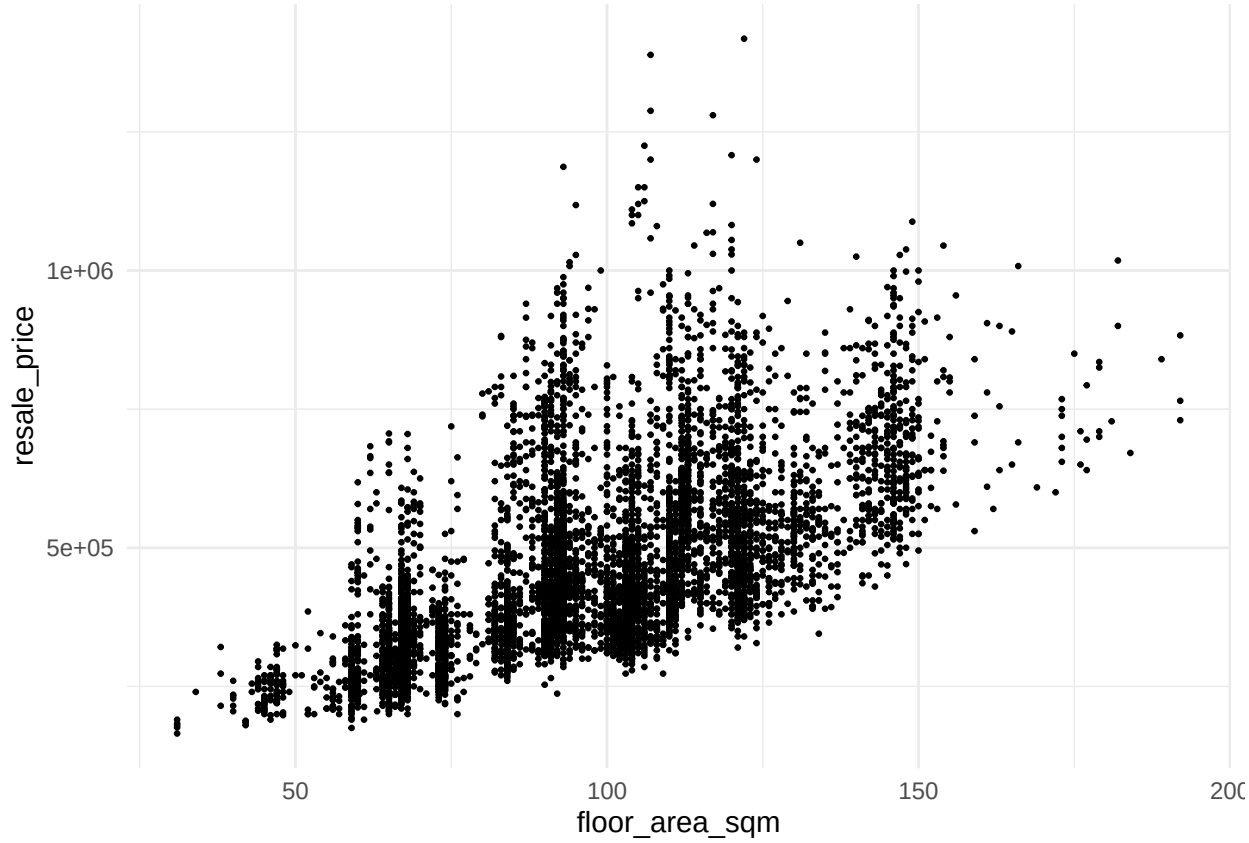
Our data collection process began with an open web research on the housing markets of Singapore. We landed on Kaggle, an open source hub of public data sets uploaded by public users, which can be used for data exploration, building predictive models, and general practice with real-world data. Specifically, our data from Kaggle was transcribed by a user from the Singapore Government Agency Website that studied Resale flat prices based on registration date from Jan-2017 onwards. Data was collected by the Housing and Development Board, commonly referred to as "HDB". It is a statutory board of the Ministry of National Development in Singapore and it seeks to provide support in homeownership and ease in rental processes for residents. Simultaneously as the HDB are providing aid, they are collecting data on what home are being bought, built, and sold for. As this government established board provides public housing for more than 80% of Singapore's population, this project will make the assumption that all data was collected as a random sample of Singapore's population and data quality is up to par with research standards.

EDA + Data Preprocessing

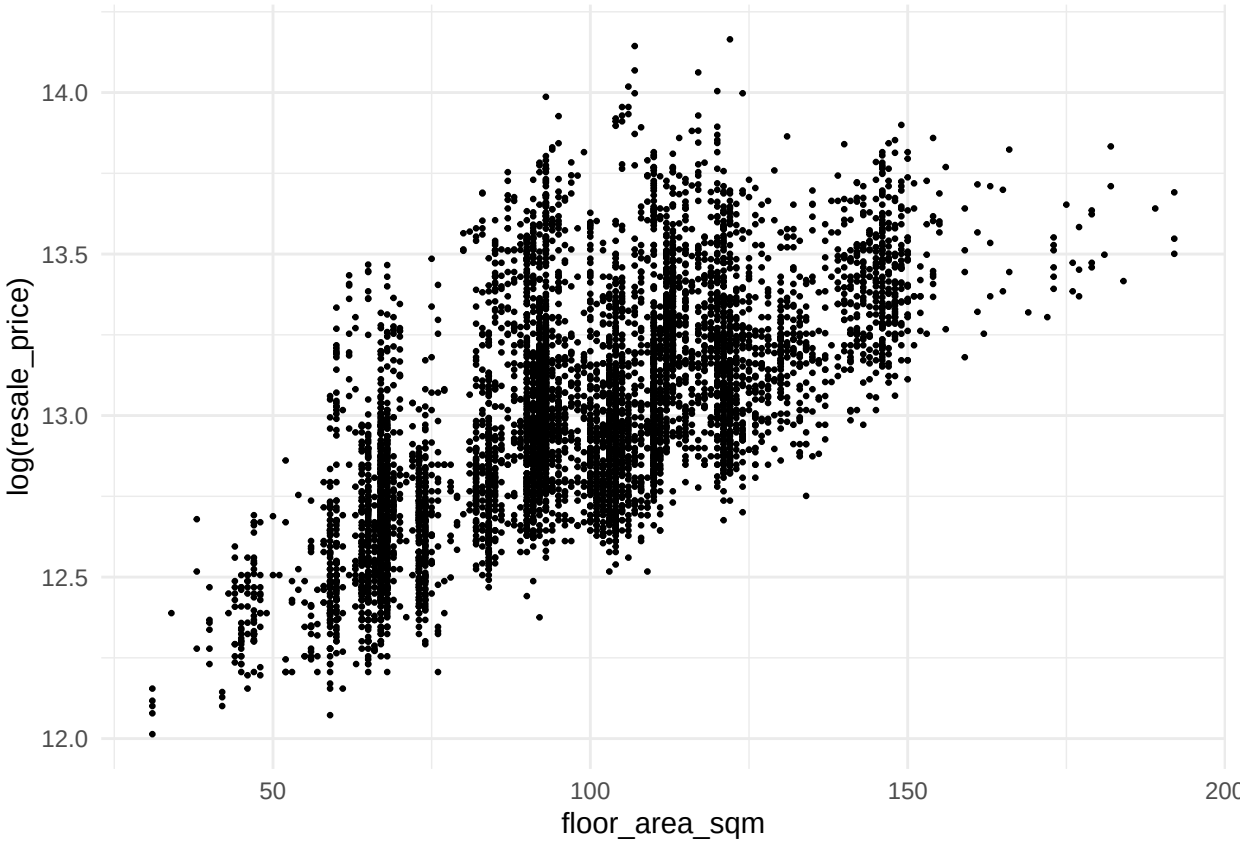
We will begin with Exploratory Data Analysis to understand the data that we are working with. We will remove outliers as necessary, transform categorical columns numeric, and take the log of resale prices to capture its full effects.

1. Log transfrom

```r
housing <- read.csv("Resale_Price_2017_2022.csv")

id <- sample(nrow(housing),7000)
sample_housing <- housing[id,]

## histogram
ggplot(sample_housing) +
  geom_point(aes(x=floor_area_sqm,y=resale_price),size=0.5,position="identity") +
  theme_minimal()
```




```r
ggplot(sample_housing) +
  geom_point(aes(x=floor_area_sqm,y=log(resale_price)),size=0.5,position="identity") +
  theme_minimal()
```



We do a sample of 7000 on the original dataset, to argue that the increase of a small amount of floor area(sqm) doesn't result it a linear amount of resale price being added, but instead some non-linear increase in the price. This is equivalent to adding to a log of the resale prices. So we conclude that it results in better prediction if we do a regression on the log(resale price).

2. One hot encoding for Flat Type

We will be apply one hot encoding for the `flat_type` column to regress on the categorical values

```
housing <- housing %>%
  mutate(
    is_2_room = ifelse(flat_type == "2 ROOM", 1, 0),
    is_3_room = ifelse(flat_type == "3 ROOM", 1, 0),
    is_4_room = ifelse(flat_type == "4 ROOM", 1, 0),
    is_5_room = ifelse(flat_type == "5 ROOM", 1, 0),
    is_executive = ifelse(flat_type == "EXECUTIVE", 1, 0),
    is_1_room = ifelse(flat_type == "1 ROOM", 1, 0),
    is_multi_generation = ifelse(flat_type == "MULTI-GENERATION", 1, 0)
  )
```

###3. Data Manipulation We will be converting the `remaining_lease` column that contains how long the lease is to be of unit month instead of the current year+month.

```
##Function to convert from years+ months to months
extract_months <- function(duration_str) {
```

```
    # Split into components
    components <- strsplit(duration_str, " ", perl = TRUE)[[1]]

    # Extract years and months (if available)
    years <- as.numeric(components[1])
    months <- ifelse(length(components) >= 3, as.numeric(components[length(components)-1]), 0)

    # Return total months
    return(years * 12 + months)
}

housing <- housing %>%
            rowwise() %>%
            mutate(remaining_lease_mth = extract_months(remaining_lease))
```

Then, we will categorize the different towns of Singapore into NSEW regions:

```
# Function to categorize towns into NSEW regions
categorize_town <- function(town) {
  north <- c("ANG MO KIO", "SEMBAWANG", "SENGKANG", "WOODLANDS", "YISHUN","BISHAN")
  south <- c("BUKIT MERAH", "BUKIT TIMAH", "CENTRAL AREA", "QUEENSTOWN")
  east <- c("BEDOK", "MARINE PARADE", "PASIR RIS", "TAMPINES")
  west <- c("BUKIT BATOK", "BUKIT PANJANG", "CHOA CHU KANG", "CLEMENTI", "JURONG EAST", "JURONG WEST",

  if (town %in% north) {
    return("North")
  } else if (town %in% south) {
    return("South")
  } else if (town %in% east) {
    return("East")
  } else if (town %in% west) {
    return("West")
  } else {
    return("Other")
  }
}

# Add a new column for NSEW region
housing <- housing %>%
  rowwise() %>%
  mutate(region = categorize_town(toupper(town)))
```

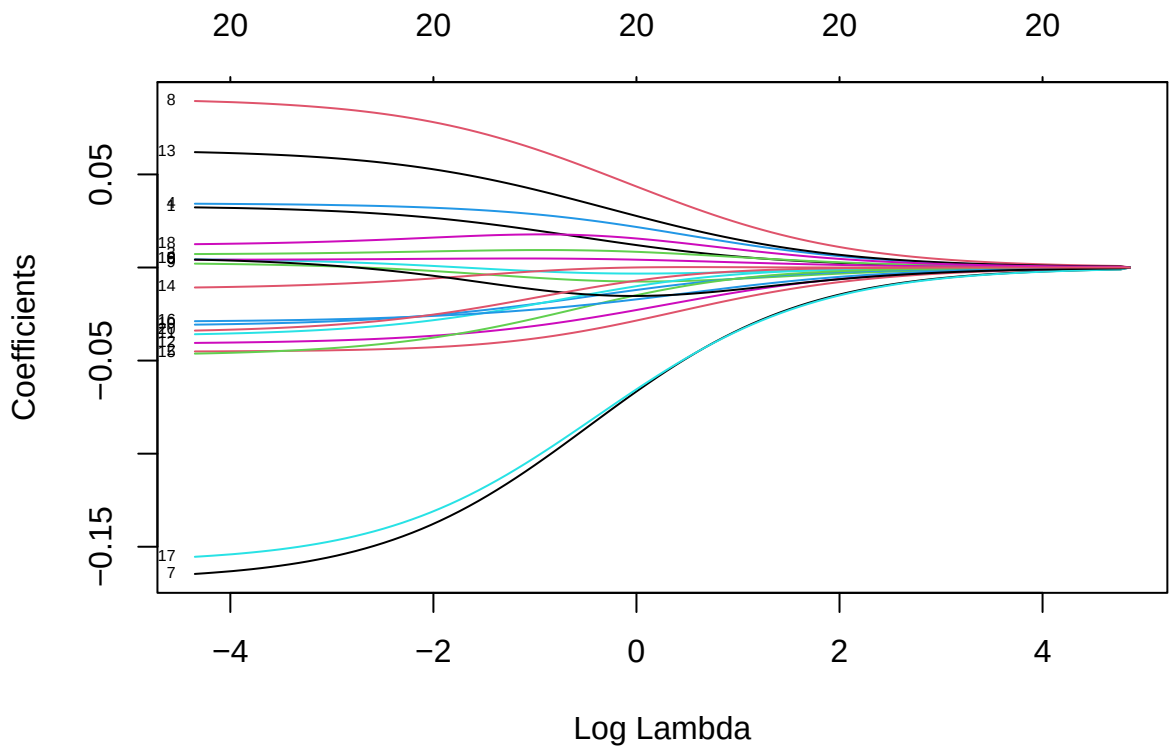
Then, we apply one-hot encoding on the regions as well:

```
housing <- housing %>%
  mutate(
    is_north = ifelse(region == "North", 1, 0),
    is_south = ifelse(region == "South", 1, 0),
    is_west = ifelse(region == "West", 1, 0),
    is_east = ifelse(region == "East", 1, 0)
  )
```

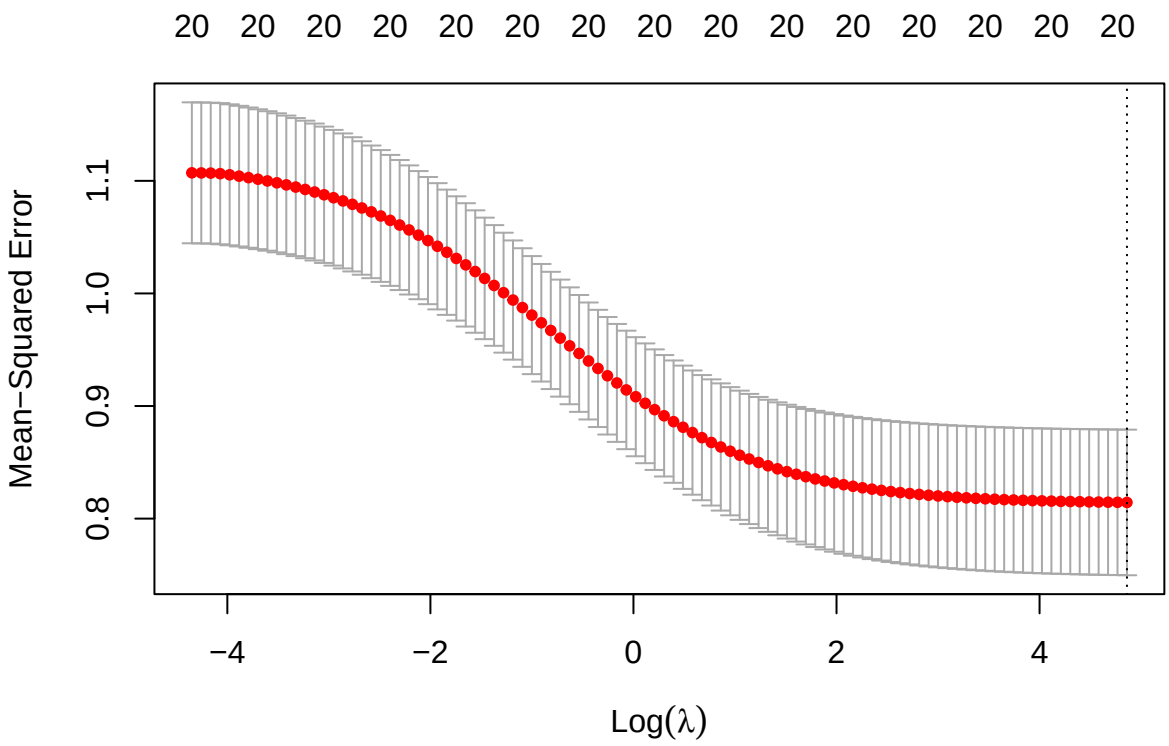
Model Training and Evaluation

In order to evaluate the most statistically significant housing metrics and reduce the number of predictors we have, we will use regularization. Beginning with Ridge Regularization:

```
#### THIS IS GENERAL, NEED TO ADJUST

# Create an X matrix that represents the data frame above, but in linear perspective.
X <- matrix(rnorm(100*20), 100, 20)

# This is our y values column.
y <- rnorm(100)

# Perform Ridge regression
ridge_model <- glmnet(X, y, alpha = 0)
predictions <- predict(ridge_model, newx = X, s = 0.01)  # Adjust s (lambda) as needed

# Visualize the lambda's and select the best one.
plot(ridge_model, xvar = "lambda", label = TRUE)
```




```
cv_ridge <- cv.glmnet(X, y, alpha = 0)
plot(cv_ridge)
```




```
# Select the best lambda value.
best_lambda <- cv_ridge$lambda.min
```

Limitations and Future Work