STAT 151A Project

Predicting Housing Resale Prices in Singapore

Michelle Vuong, Celina Mac, Lewis Chong

April 10th 2024

Introduction

According to the Federal Reserve Board of the United States, the effects of the COVID-19 pandemic on the U.S. housing market have led to a rapid increase in housing prices with almost $9 trillion in the wealth of owner occupied housing between the years 2020-2022. From the perspective of a prospective homeowner, the median housing price in the United States has nearly tripled between 1992 to 2021 with the largest spike in housing prices occurring during after the pandemic.

This consequence of the pandemic is not unique to the United States economy and instead, is omnipresent all throughout the world. With the effects of the pandemic taken into consideration, we are motivated by this recent event to see how the theory of linear modelling could be applied in order to better understand how it affects the price of housing across a large population. In particular, we aim to do this with a country that is smaller relative to the United States on the country of Singapore. Throughout this report, we will attempt to understand the effects of COVID-19 and housing prices utilizing techniques such as linear regression, ridge and lasso models, and the model's effectiveness using cross-validation.

Problem Description

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

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.

```
# Reading the csv file of our data
housing <- read_csv("Resale_Price_2017_2022.csv",show_col_types=FALSE)
```

Data Inspection

```
# Checking for NA values in our data
na_counts <- colSums(is.na(housing))
any_na <- any(na_counts > 0)

cat("Datasets contains NA values : ",any_na)
```

```
## Datasets contains NA values :  FALSE
```

We have found that there are no null values in the dataset, and so it is clean for further analyses that will be conducted for this report.

```
# Inspecting the data
glimpse(housing)
```

```
## Rows: 134,168
## Columns: 11
## $ month               <chr> "2017-01", "2017-01", "2017-01", "2017-01", "2017-~
## $ town                <chr> "ANG MO KIO", "ANG MO KIO", "ANG MO KIO", "ANG MO ~
## $ flat_type           <chr> "2 ROOM", "3 ROOM", "3 ROOM", "3 ROOM", "3 ROOM", ~
## $ block               <chr> "406", "108", "602", "465", "601", "150", "447", "~
## $ street_name         <chr> "ANG MO KIO AVE 10", "ANG MO KIO AVE 4", "ANG MO K~
## $ storey_range        <chr> "10 TO 12", "01 TO 03", "01 TO 03", "04 TO 06", "0~
## $ floor_area_sqm      <dbl> 44, 67, 67, 68, 67, 68, 68, 67, 68, 67, 68, 67, 67~
## $ flat_model          <chr> "Improved", "New Generation", "New Generation", "N~
## $ lease_commence_date <dbl> 1979, 1978, 1980, 1980, 1980, 1981, 1979, 1976, 19~
## $ remaining_lease     <chr> "61 years 04 months", "60 years 07 months", "62 ye~
## $ resale_price        <dbl> 232000, 250000, 262000, 265000, 265000, 275000, 28~
```

Upon our first inspection of the data, we will be using the `resale_price` as the response variable since we are interested in predicting housing resale price. As for our predictors, we will be using the columns `flat_type,floor_area_sqm,remaining_lease,town`.

```
# Selecting the necessary columns for our analysis
# Modifying the time variable into a more easily usable data type
housing <- housing %>% dplyr::select(month,town,flat_type,floor_area_sqm,
                                     remaining_lease,resale_price) %>%
            mutate(month = parse_date(month,format="%Y-%m"))
```

Another thing to note for this dataset is that it contains a lot of categorical variables and the variables that we have chosen also fall under this characterization and so we must employ a technique that will be able to translate our categorical variables into numerical data for the purpose of our models.

1. One-hot-encoding From the above summary, we see that the `flat_type` is a categorical variable. Thus,we will be apply one hot encoding on it for the model to regress.

```
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)
  ) %>%
  dplyr::select(!flat_type)
```

We also must note that for `town`, there are too many distinct values:

```
cat("Distinct values for `town` :",length(unique(housing$town)))
```

```
## Distinct values for ‘town‘ : 26
```

In order to remedy this in a way that will be better utilized for our model, we will categorize the different towns of Singapore into NSEW (North, South, East, West) regions, and further apply one-hot encoding as done above since it is a categorical variable.

```
# 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))) %>%
  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)
  ) %>%
  dplyr::select(!region)
```

Note that for one-hot encoding on the columns `region` and `flat_type` , we can just drop one of the columns as it can be identified by the rest of the columns, i.e `(is_north,is_south,is_west) = (0,0,0)`corresponds to the house being in the East Region.

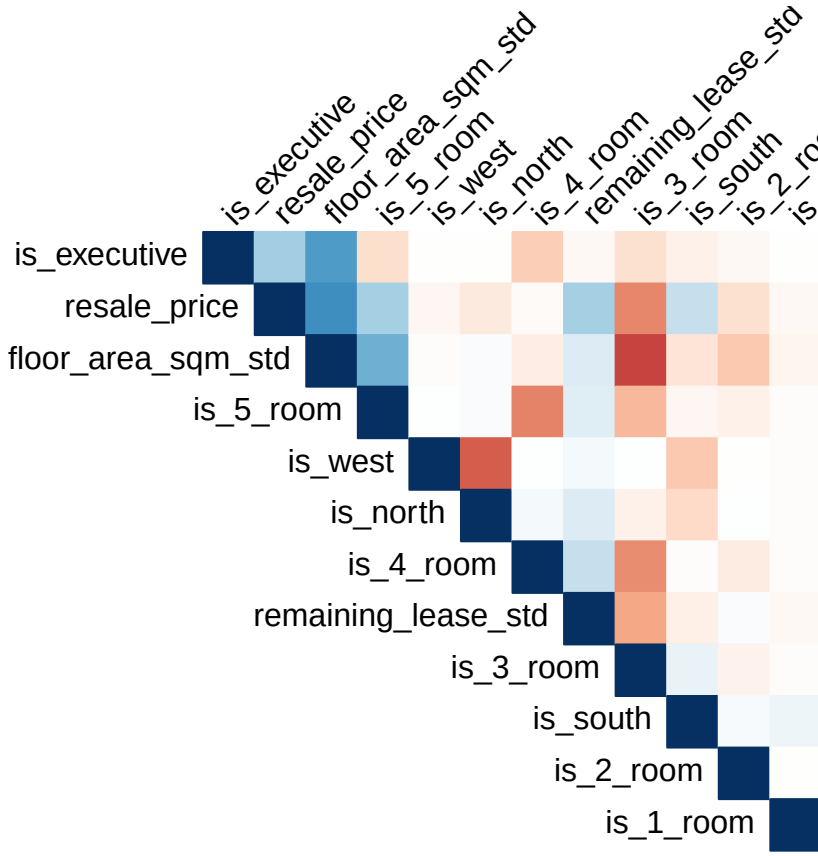
```
housing <- housing %>%
        dplyr::select(!c(is_east,is_multi_generation,town))
```

2. Data Manipulation We will be converting the `remaining_lease` column that contains how long the lease is to be of unit months 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)) %>%
            dplyr::select(!remaining_lease)
```

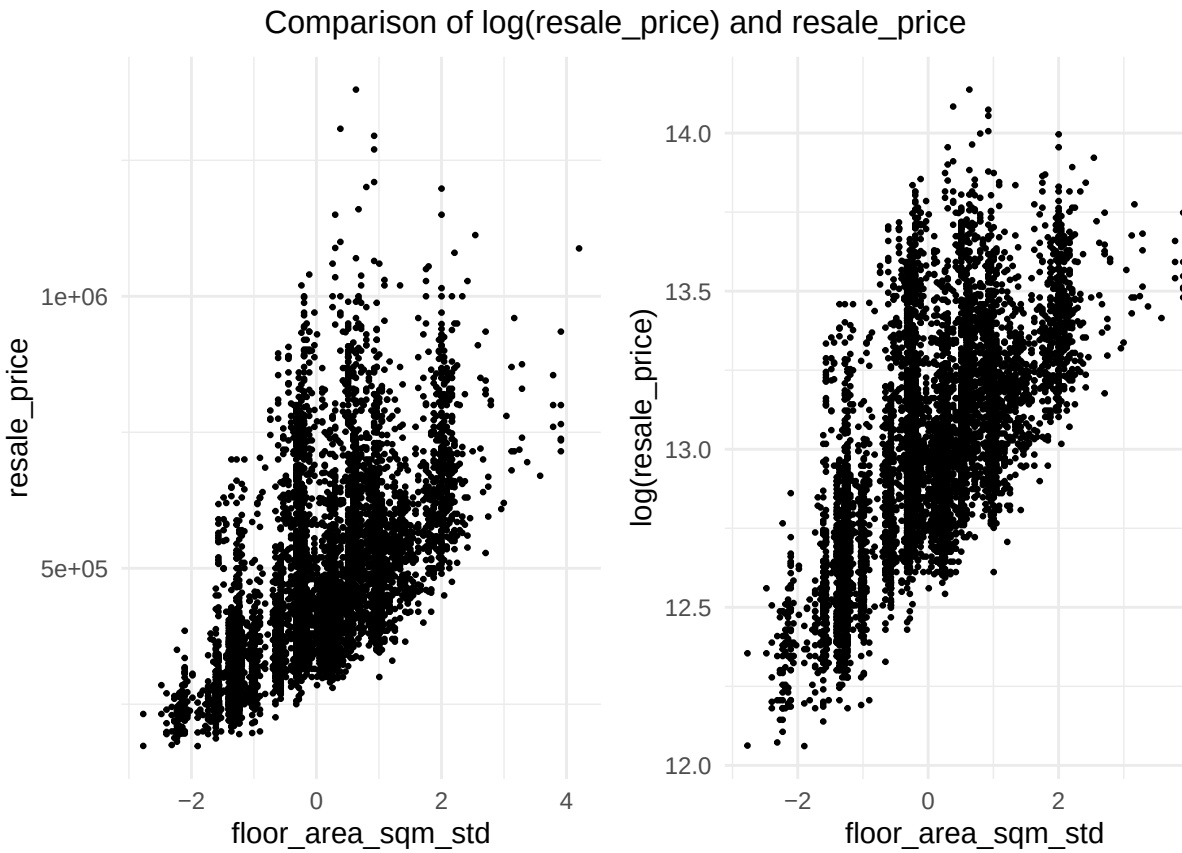
```
#Standardizing predictor columns
X <- housing %>% dplyr::select(floor_area_sqm,remaining_lease_mth) %>%
      scale() %>% as.data.frame()
colnames(X) <- c("floor_area_sqm_std","remaining_lease_std")

housing <- cbind(housing,X) %>%
  dplyr::select(!c(remaining_lease_mth,floor_area_sqm))
```

```
corr_mat<- cor(housing[sapply(housing, is.numeric)])
corrplot(corr_mat, method = "color", type = "upper", order ="hclust",
         tl.col = "black", tl.srt = 45)
```



3. Standardizing Predictor Columns

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

p1 <- ggplot(sample_housing) +
        geom_point(aes(x=floor_area_sqm_std,y=resale_price),
                   size=0.5,position="identity") +
        theme_minimal()
p2 <- ggplot(sample_housing) +
        geom_point(aes(x=floor_area_sqm_std,y=log(resale_price)),
                   size=0.5,position="identity") +
        theme_minimal()

grid.arrange(p1,p2,ncol=2,top="Comparison of log(resale_price) and resale_price")
```



4. Log transform

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).

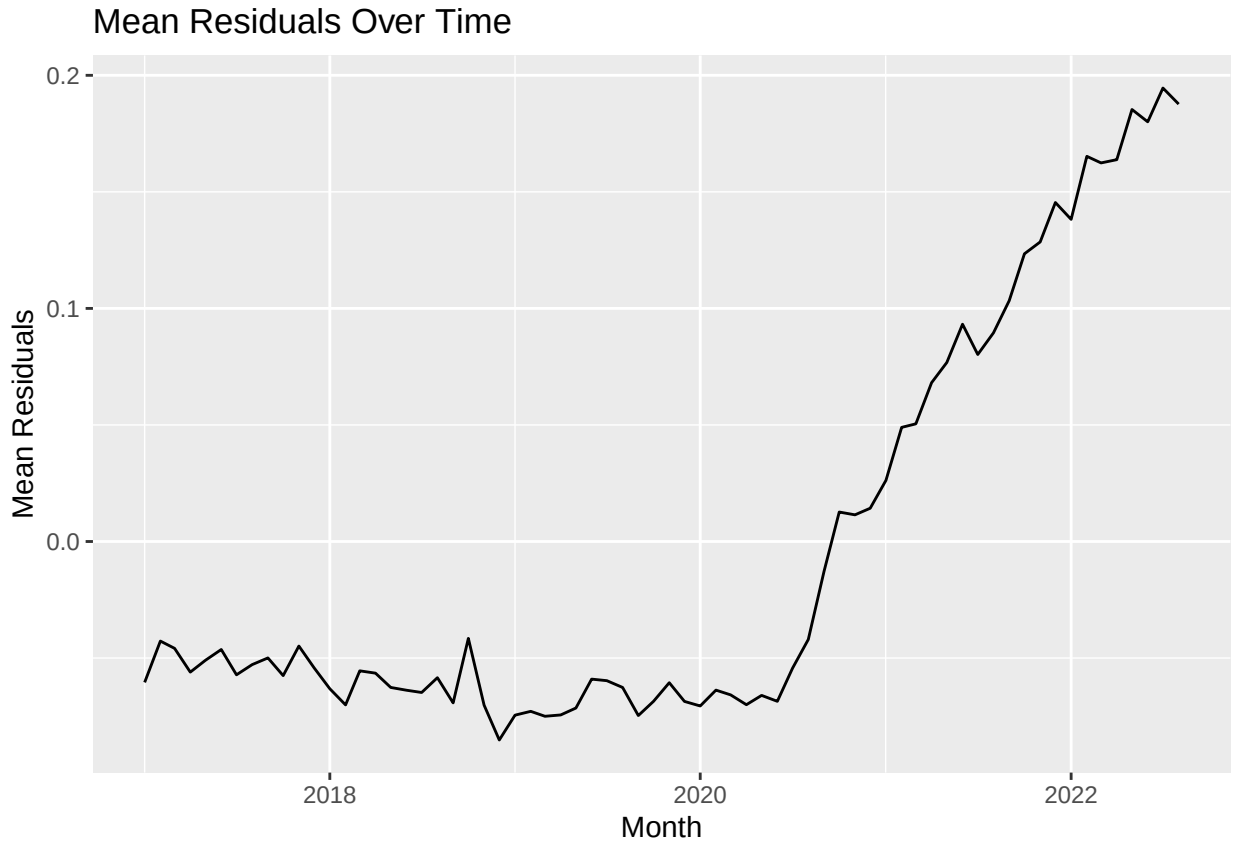
Methods

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:

```
df <- housing %>% dplyr::select(!month)

ori <- lm(log(resale_price) ~ . + 0, data = df)

yhat <- predict(ori)

y <- log(df$resale_price)

MSE_ori <- mean((y-yhat)^2)
cat("MSE from the linear model: ", MSE_ori)
```

```
## MSE from the linear model:  0.1236083
```

```r
epsilon <- data.frame(resid=resid(ori),month=housing$month)

residuals_means <- epsilon %>%
  group_by(month) %>%
  summarise(mean_residual = mean(resid))

# Now you can plot the mean residuals against the months
ggplot(residuals_means, aes(x = month, y = mean_residual)) +
  geom_line() +
  labs(x = "Month", y = "Mean Residuals", title = "Mean Residuals Over Time")
```




```r
# Set seed for reproducibility
set.seed(123)
# Create an index for splitting the data
split <- createDataPartition(y = housing$resale_price, p = 0.8, list = FALSE)
X <- dplyr::select(df,!resale_price)
X_train <- X[split,]
X_test <- as.matrix(X[-split,])
y_train <- df$resale_price[split]
y_test <- df$resale_price[-split]
# Fit Ridge model
ridge <- glmnet::glmnet(x = X_train, y = y_train, alpha = 0)
yhat_ridge <- predict(ridge, newx = X_test)
lambda_ridge <- ridge$lambda
# Min MAE
```

```r
mae_ridge <- apply(abs(yhat_ridge - y_test),2,mean)

#col mean for each lambda
min_ridge <- which.min(mae_ridge) #index corresponding to min MAE
min_lambda_ridge <- lambda_ridge[min_ridge]
coef_min_ridge <- coef(ridge)[,min_ridge] #Corresponding indices of the non-zero coefs
lagged_features <- as.numeric(which(coef_min_ridge != 0))
print("Lagged features present in the MAE-optimal ridge model:")
```

```
## [1] "Lagged features present in the MAE-optimal ridge model:"
```

```r
# Fit lasso model
lasso <- glmnet::glmnet(x = X_train, y = y_train, alpha = 1)
yhat_lasso <- predict(lasso, newx = X_test)
lambda_lasso <- lasso$lambda # Min MAE
mae_lasso <- apply(abs(yhat_lasso - y_test),2,mean)
#col mean for each lambda
min_lasso <- which.min(mae_lasso) #index corresponding to min MAE
min_lambda_lasso <- lambda_lasso[min_lasso]
coef_min_lasso <- as.data.frame(coef(lasso)[,min_lasso])
#Corresponding indices of the non-zero coefs
coef_min_lasso
```

```
##                     coef(lasso)[, min_lasso]
## (Intercept)                         515105.85
## is_2_room                          -123102.83
## is_3_room                           -66230.91
## is_4_room                           -36285.81
## is_5_room                                0.00
## is_executive                         44465.38
## is_1_room                          -227396.52
## is_north                            -58289.92
## is_south                            173089.74
## is_west                             -24142.79
## floor_area_sqm_std                   69083.49
## remaining_lease_std                  46344.82
```

```r
# # Perform Ridge regression #
# ridge_model <- glmnet(X[,-ncol(X)], X[,ncol(X)], alpha = 0)
# predictions <- predict.glm(ridge_model, newdata = X[,-ncol(X)])  # Adjust s (lambda) as needed
# # Select the index with best lambda value.
# mae_ridge <- apply(abs(yhat_ridge - y[second_half]),2,mean)
# min_lambda <- ridge_model$lambda %>%
#                   as.vector %>%
#                     which.min
# best_lambda <- ridge_model$lambda[min_lambda]
# best_lambda
```

F-test for feature selection

Using the F-test to understand the variability between the features in the data set, we want to select the features with most impact on our y variable.

```r
# ANOVA - This is giving high RSE, meaning there is a lot of unexplained variability.
anova_test <- aov(resale_price ~ is_1_room +
                    is_2_room +
                    is_3_room +
                    is_4_room +
                    is_5_room +
                    is_executive +
                    is_north +
                    is_south +
                    is_west +
                    floor_area_sqm_std +
                    remaining_lease_std, df)


# F-test on each features with 2 levels
is_1_room <- var.test(resale_price ~ is_1_room, df,
         alternative = "two.sided")
is_2_room <- var.test(resale_price ~ is_2_room, df,
         alternative = "two.sided")
is_3_room <- var.test(resale_price ~ is_3_room, df,
         alternative = "two.sided")
is_4_room <- var.test(resale_price ~ is_4_room, df,
         alternative = "two.sided")
is_5_room <- var.test(resale_price ~ is_5_room, df,
         alternative = "two.sided")
is_executive <- var.test(resale_price ~ is_executive, df,
         alternative = "two.sided")
is_north <- var.test(resale_price ~ is_north, df,
         alternative = "two.sided")
is_south <- var.test(resale_price ~ is_south, df,
         alternative = "two.sided")
is_west <- var.test(resale_price ~ is_west, df,
         alternative = "two.sided")

F_test_vals <- list(
  is_1_room = is_1_room,
  is_2_room = is_2_room,
  is_3_room = is_3_room,
  is_4_room = is_4_room,
  is_5_room = is_5_room,
  is_executive = is_executive,
  is_north = is_north,
  is_south = is_south,
  is_west = is_west
)

# Extract p-values
p_vals <- sapply(F_test_vals, function(result) {
  result$p.value
})

# Check if any p-value is less than a significance level (e.g., 0.05)
any(p_vals > 0.05)
```

```
## [1] TRUE
```

```
big_p <- which(p_vals>0.05)

cat("Feature with p_value > 0.05: ", names(df)[big_p],"\n")
```

```
## Feature with p_value > 0.05:  is_5_room
```

```
cat("Corresponding p-value: ",p_vals[big_p])
```

```
## Corresponding p-value:  0.1970816
```

```
# More than two levels, won't run with F-test.
#floor_area_sqm_std <- var.test(resale_price ~ floor_area_sqm_std, housing,
     #    alternative = "two.sided")
#remaining_lease_std <- var.test(resale_price ~ remaining_lease_std, housing,
     #    alternative = "two.sided")
```

In the Lasso model the feature "is_5_room" has a zero coefficient and this F-test confirms that we can drop this feature due to the p-value being higher than 0.05. So we will re-run the linear regression ommitting this feature.

```
ori_no_5_rooms <- lm(log(resale_price) ~ is_2_room +
                                         is_3_room +
                                         is_4_room +
                                         is_executive +
                                         is_1_room +
                                         is_north +
                                         is_south +
                                         is_west +
                                         floor_area_sqm_std +
                                         remaining_lease_std +
                                         0, data = df)

rsquared <- summary(ori_no_5_rooms)
rsquared$r.squared
```

```
## [1] 0.9470657
```

Regression Analysis on the Effects of COVID-19 on the Housing Market in Singapore

Now that we have performed ANOVA and F-test for feature selection and determined which variables to utilize in our model, we can move forward with further analyzing the effects of COVID-19 on the housing market in Singapore.

Subsetting our dataset

We will begin by filtering all dates before April 2020 as that was when the Singaporean government began enforcing preventive measures for the pandemic. This date was chosen based on the following information:

1. The first COVID-19 case in Singapore was confirmed on January 23, 2020.
2. COVID-19 clusters in the population were recorded in late March and early April 2020.
3. Singapore enacted the "COVID-19 Control Order" in April 3, 2020 and announced the "circuit breaker lockdown," which was a set of stringent preventive measures to curb the spread of COVID-19.

With all this information in mind, we believe that April 2020 would be the best date to choose as the boundary when subsetting our data into pre-COVID and COVID time periods.

```
# Subsetting our original dataset into the two time periods
pre_covid <- housing %>%
  filter(month < "2020-04-01")

covid <- housing %>%
  filter(month >= "2020-04-01")
```

Running Linear Regression on Both Periods

```
# Linear Regression on Pre-Covid Data
pre_covid_lm <- lm(log(resale_price) ~ is_2_room +
     is_3_room +
     is_4_room +
     is_executive +
     is_1_room +
     is_north +
     is_south +
     is_west +
     floor_area_sqm_std +
     remaining_lease_std +
     0, data = pre_covid)

# Linear Regression on Covid Period Data
covid_lm <- lm(log(resale_price) ~ is_2_room +
     is_3_room +
     is_4_room +
     is_executive +
     is_1_room +
     is_north +
     is_south +
     is_west +
     floor_area_sqm_std +
     remaining_lease_std +
     0, data = covid)

summary(pre_covid_lm)
```

```
##
```

```
## Call:
## lm(formula = log(resale_price) ~ is_2_room + is_3_room + is_4_room +
##     is_executive + is_1_room + is_north + is_south + is_west +
##     floor_area_sqm_std + remaining_lease_std + 0, data = pre_covid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.605  -1.147   0.395   2.647  10.887
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## is_2_room           22.89008    0.11503  198.98   <2e-16 ***
## is_3_room           17.33964    0.05462  317.45   <2e-16 ***
## is_4_room            9.07951    0.02839  319.86   <2e-16 ***
## is_executive        -4.39709    0.05749  -76.48   <2e-16 ***
## is_1_room           26.62775    0.52896   50.34   <2e-16 ***
## is_north             5.08096    0.02984  170.26   <2e-16 ***
## is_south             5.86489    0.04549  128.93   <2e-16 ***
## is_west              5.03082    0.02746  183.21   <2e-16 ***
## floor_area_sqm_std   7.13454    0.02575  277.10   <2e-16 ***
## remaining_lease_std  0.42782    0.01374   31.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.939 on 69755 degrees of freedom
## Multiple R-squared:  0.9484, Adjusted R-squared:  0.9484
## F-statistic: 1.282e+05 on 10 and 69755 DF,  p-value: < 2.2e-16
```

```
summary(covid_lm)
```

```
##
## Call:
## lm(formula = log(resale_price) ~ is_2_room + is_3_room + is_4_room +
##     is_executive + is_1_room + is_north + is_south + is_west +
##     floor_area_sqm_std + remaining_lease_std + 0, data = covid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.531  -0.944   0.353   2.809  10.739
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## is_2_room           24.44998    0.12428  196.73   <2e-16 ***
## is_3_room           18.49991    0.05935  311.71   <2e-16 ***
## is_4_room            9.67229    0.03016  320.65   <2e-16 ***
## is_executive        -5.30438    0.06344  -83.61   <2e-16 ***
## is_1_room           28.80350    0.62887   45.80   <2e-16 ***
## is_north             4.87606    0.03206  152.07   <2e-16 ***
## is_south             5.80531    0.04794  121.09   <2e-16 ***
## is_west              4.94154    0.02896  170.64   <2e-16 ***
## floor_area_sqm_std   7.75964    0.02845  272.76   <2e-16 ***
## remaining_lease_std  0.54374    0.01262   43.07   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 3.035 on 64393 degrees of freedom
## Multiple R-squared:  0.9462, Adjusted R-squared:  0.9462
## F-statistic: 1.133e+05 on 10 and 64393 DF,  p-value: < 2.2e-16
```

Results and Discussion

After sub-setting the data into two categories (pre-covid and post-covid) and running our regression on both categories, we called the R function summary() to gather information on how our model was performing. In our pre-covid model, we achieved a R squared value of 0.9484. In our post-covid model, we achieved a R squared of 0.9462. There is only a slight difference between the two, which tells us....

Limitations and Future Work

Reference Resale flat prices based on registration date from Jan-2017 onwards. Data.gov.sg. https://beta.data.gov.sg/datasets/d_8b84c4ee58e3cfc0ece0d773c8ca6abc/view

Origin of data set for Singaporean homes. Kaggle.com. https://www.kaggle.com/code/ashydv/housing-price-prediction-linear-regression/notebook