loding the package

loading the data

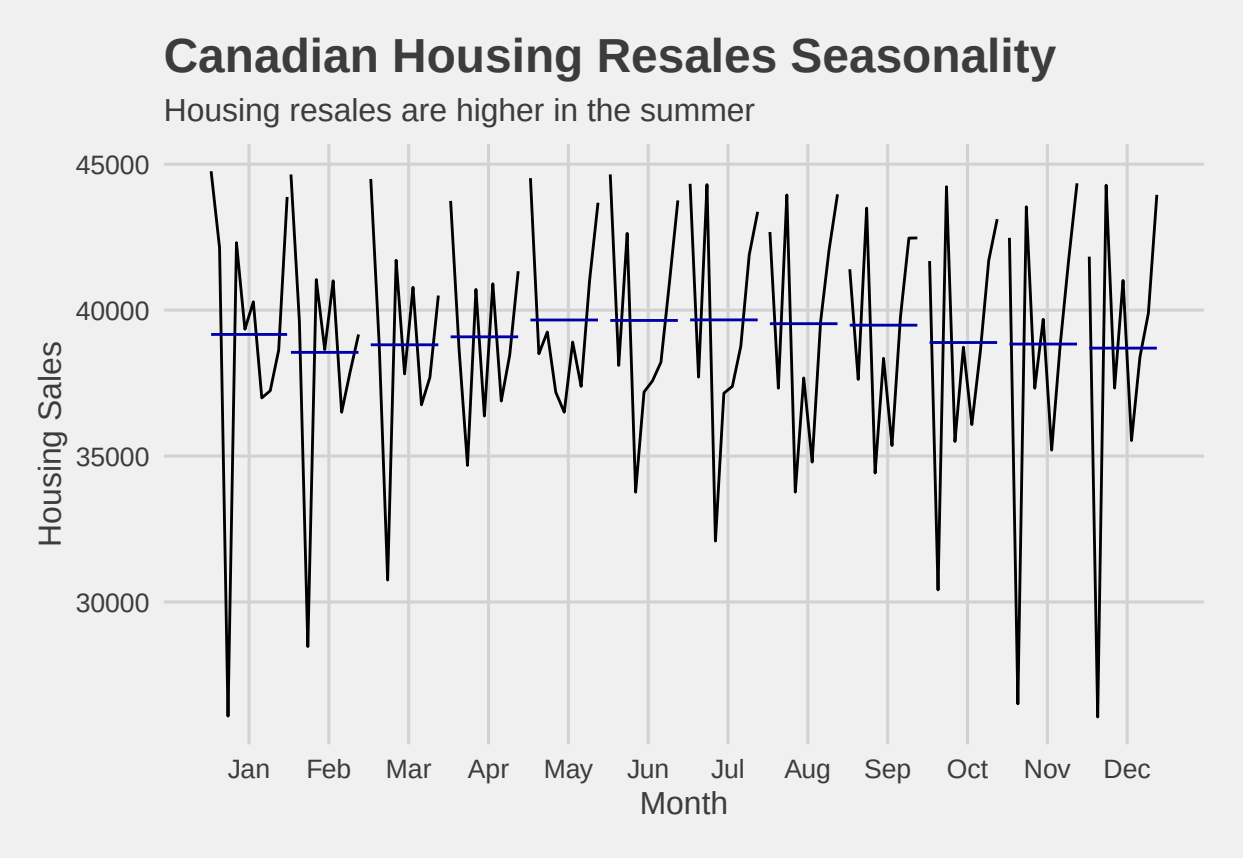
```
#1.import the data
Can_housing_sell_data.raw <- read_excel("/Users/tie/Documents/GitHub/ECON-493-forcasting

#transfer data to ts form
Can_month_housing_sell.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1), en
```

#The overall data and seasonal plot

```
autoplot(Can_month_housing_sell.ts) +
theme_fivethirtyeight() +
xlab("Year") +
ylab("Housing Sales") +
ggtitle("Canadian Housing Resales from 2007 to 2023") +
theme(axis.title = element_text())
```


```
#1.1.2 the seasonal plot
Can_month_housing_sell_plot_ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2010, 1

#1.1.3 the seasonality graph
ggsubseriesplot(Can_month_housing_sell_plot_ts) +
```

```
ylab("Number of Homes Sold") +
theme_fivethirtyeight() +
xlab("Month") +
ylab("Housing Sales") +
ggtitle("Canadian Housing Resales Seasonality") +
labs(subtitle = "Housing resales are higher in the summer") +
theme(axis.title = element_text())
```




```
#plot the bank rate.

 Bank_rate_1_ARIMA_raw <- "v122530" #Financial market statistics, last Wednesday unless
 Bank_rate_1_ARIMA.st <- get_cansim_vector( Bank_rate_1_ARIMA_raw, start_time = "2007/01
```

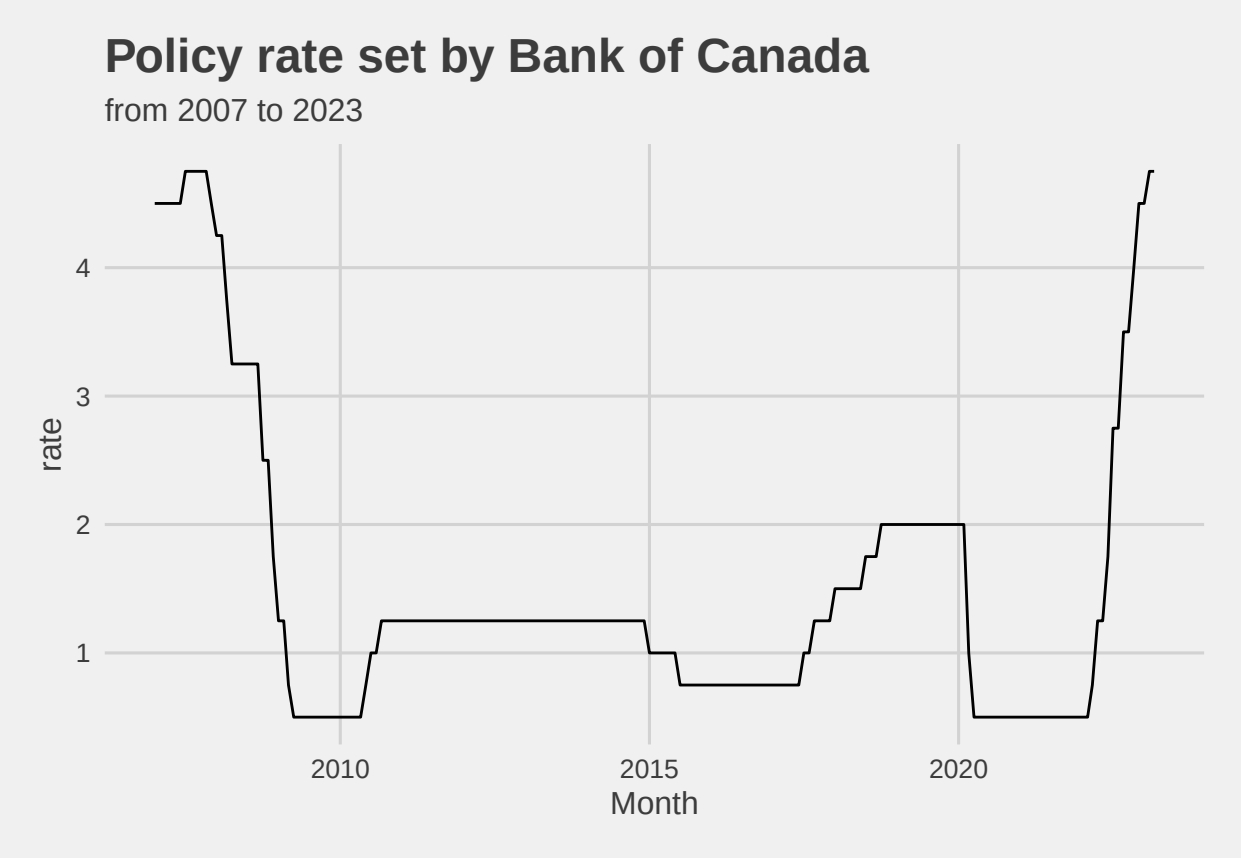
```
## Accessing CANSIM NDM vectors from Statistics Canada
```

```
 Bank_rate_1_ARIMA_year.st <- year( Bank_rate_1_ARIMA.st$REF_DATE[1])
 Bank_rate_1_ARIMA_month.st <- month( Bank_rate_1_ARIMA.st$REF_DATE[1])
 c(Bank_rate_1_ARIMA_year.st,  Bank_rate_1_ARIMA_month.st)
```

```
## [1] 2007    1
```

```
 Bank_rate_1_ARIMA.ts<- ts( Bank_rate_1_ARIMA.st$VALUE, start = c( Bank_rate_1_ARIMA_yea
```

```
autoplot(Bank_rate_1_ARIMA.ts) +
ylab("Policy rate") +
theme_fivethirtyeight() +
xlab("Month") +
ylab("rate") +
ggtitle("Policy rate set by Bank of Canada ") +
labs(subtitle = "from 2007 to 2023") +
theme(axis.title = element_text())
```



The part one: ARIMA Lode all ARIMA combination data

```
##### covid start at 2019/12
##### 2008 recession end in 2008/07

#excel is way better on doing the covid dummy variable
The_covid_dummy <- read_excel("/Users/tie/Documents/GitHub/ECON-493-forcasting-economy/T
    col_types = c("skip", "numeric"))
```

```
#construed model
#note: first I realize with those fancy settingmay help them I realize this is waste o

######## all data model
```

```r
#The entire data
Can_month_housing_sell.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1), en
The_covid_dummy_ARIMA.ts <- ts(The_covid_dummy, start = c(2007, 1), end = c(2023, 2), fr

#moding time
all_data_model_1 <- auto.arima(Can_month_housing_sell.ts, approximation = FALSE, paralle
                               num.cores = 10, start.p = 0, start.q = 0, start.P = 0,
                               start.Q = 0, max.Q = 5, max.P = 5, max.D = 5,
                               max.d = 5, max.p = 5, max.q = 5)

all_data_model_2 <- auto.arima(Can_month_housing_sell.ts)

# all data plus dummy variable

all_data_dummy_1 <- auto.arima(Can_month_housing_sell.ts, xreg = The_covid_dummy_ARIMA.t
                               num.cores = 10, start.p = 0, start.q = 0, start.P = 0,
                               start.Q = 0, max.Q = 5, max.P = 5, max.D = 5,
                               max.d = 5, max.p = 5, max.q = 5, d = 1)
#Regression with ARIMA(2,1,0) errors

all_data_dummy_2 <- auto.arima(Can_month_housing_sell.ts, xreg = The_covid_dummy_ARIMA.t

#print(all_data_dummy_2)
#Regression with ARIMA(0,1,1) errors
```

```r
#############

#The data set without
after_2008_forecasting_data <- window(Can_month_housing_sell.ts, start= c(2009,7), end=
The_covid_dummy_after_208.ts <- ts(The_covid_dummy, start = c(2009, 7), end = c(2023, 2)


####### after_2008_forecasting_model_1
after_2008_forecasting_model_1 <- auto.arima(after_2008_forecasting_data, d = 1,
                                             approximation = FALSE, parallel = TRUE, ste
                                             num.cores = 10, start.p = 0, start.q = 0, s
                                             start.Q = 0, max.Q = 5, max.P = 5, max.D =
                                             max.d = 5, max.p = 5, max.q = 5)

after_2008_forecasting_model_2 <- auto.arima(after_2008_forecasting_data)



past_2008_dummy_1 <- auto.arima(after_2008_forecasting_data, d = 1, xreg = The_covid_dum
```

```
                                                approximation = FALSE, parallel = TRUE, ste
                                                num.cores = 10, start.p = 0, start.q = 0, s
                                                start.Q = 0, max.Q = 5, max.P = 5, max.D =
                                                max.d = 5, max.p = 5, max.q = 5)
#print(past_2008_dummy_1)
#Regression with ARIMA(1,1,2) errors

past_2008_dummy_2 <- auto.arima(after_2008_forecasting_data, xreg = The_covid_dummy_afte

#print(past_2008_dummy_2)
#Regression with ARIMA(0,1,1) errors

#I did do the ARIMA with Covid dummy variable but it does not including in the paper b
#it just keep tell me it cannot found model :(
#the three years before and after does not included within the paper..
```

```
########## entire data without covid shock

#The data from 2007 to 2019 (the entire data without Covid shock)
Can_month_housing_sell_without_covid_shock.ts <- ts(Can_housing_sell_data.raw$Canada, st

##### model


entil_data_without_covid_shock_1<- auto.arima(Can_month_housing_sell_without_covid_shock
                                                approximation = FALSE, parallel = TRUE, st
                                                num.cores = 10, start.p = 0, start.q = 0,
                                                start.Q = 0, max.Q = 5, max.P = 5, max.D =
                                                max.d = 5, max.p = 5, max.q = 5)

entil_data_without_covid_shock_2 <- auto.arima(Can_month_housing_sell_without_covid_shoc
```

```
#########

#The data from 2008 to 2019 (After recession without covid shock)
after_2008_forecasting_without_covid_shock_data <- window(Can_month_housing_sell.ts, sta


test_model_one_after_2008_forecasting_without_covid_shock_1 <- auto.arima(after_2008_for
                                                                        approximation
                                                                        num.cores = 10
                                                                        start.Q = 0, m
                                                                        max.d = 5, max
```

```
test_model_one_after_2008_forecasting_without_covid_shock_2 <- auto.arima(after_2008_for
```

```
from_2007_to_2017 <- window(Can_month_housing_sell.ts, start= c(2007,1), end= c(2017, 1)

from_2007_to_2017_1 <- auto.arima(from_2007_to_2017,
                                  approximation = FALSE, parallel = TRUE, stepwise = FAL
                                  num.cores = 10, start.p = 0, start.q = 0, start.P = 0,
                                  start.Q = 0, max.Q = 5, max.P = 5, max.D = 5,
                                  max.d = 5, max.p = 5, max.q = 5)

from_2007_to_2017_2 <- auto.arima(from_2007_to_2017)
```

```
####################
# Create a data frame with the names and ARIMA models
ARIMA_models_table <- data.frame(
  ARIMA = c(as.character(all_data_model_1),
            as.character(all_data_model_2),
            as.character(after_2008_forecasting_model_1),
            as.character(after_2008_forecasting_model_2),
            as.character(test_model_one_after_2008_forecasting_without_covid_shock_1),
            as.character(test_model_one_after_2008_forecasting_without_covid_shock_2),
            as.character(entil_data_without_covid_shock_1),
            as.character(entil_data_without_covid_shock_2),
            as.character(from_2007_to_2017_1),
            as.character(from_2007_to_2017_2))
)

# Print the table
print(ARIMA_models_table)
```

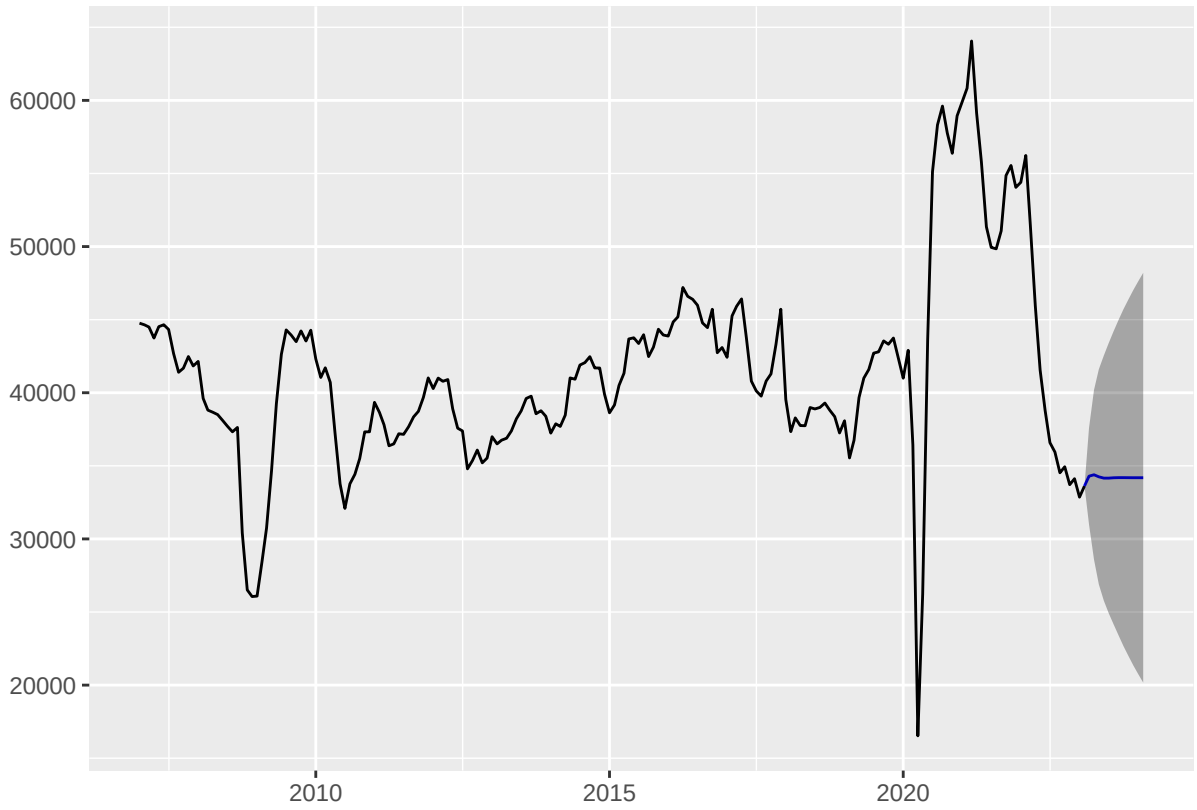
```
##                       ARIMA
## 1               ARIMA(2,1,0)
## 2               ARIMA(0,1,1)
## 3               ARIMA(2,1,0)
## 4               ARIMA(0,1,1)
## 5               ARIMA(0,1,1)
## 6               ARIMA(0,1,0)
## 7  ARIMA(2,1,2)(0,0,1)[12]
## 8  ARIMA(2,1,2)(0,0,1)[12]
## 9  ARIMA(2,1,2)(0,0,1)[12]
## 10 ARIMA(2,1,2)(0,0,1)[12]
```

Trying to do the ARIMA with dummy variable (its work! but no idea how to do cross validation of it) give up on it

```r
#maybe put some noise?
#generate the random noise
set.seed(123)
error <- rnorm(196, mean = 0, sd = 1)
error_evil <- (error - mean(error))*1/10
#make the noises smaller?

#transfer the error data into the time series
error_evil.ts <- ts(error_evil, start = c(2007, 1), end = c(2023, 2), frequency = 12)

#The covid dummy variable
The_covid_dummy.raw <- ts(The_covid_dummy$Covid_dummy, start = c(2007, 1), end = c(2023,

#put them together!
The_covid_dummy.ts <- (error_evil.ts + The_covid_dummy.raw )


ARIMA_with_Covid_dummy <- auto.arima(Can_month_housing_sell.ts, xreg = The_covid_dummy.t
print(ARIMA_with_Covid_dummy)
```

```
## Series: Can_month_housing_sell.ts
## Regression with ARIMA(2,1,0) errors
##
## Coefficients:
##          ar1      ar2      xreg
##       0.4445  -0.2708  -57.4741
## s.e.  0.0692   0.0689  810.2598
##
## sigma^2 = 6630664:  log likelihood = -1788.23
## AIC=3584.46   AICc=3584.67   BIC=3597.51
```

```r
# ARIMA(2,1,0).......
# Create future covid dummy for the next 12 months
covid_dummy_future <- numeric(length = 12)
for (i in 1:12) {
  covid_dummy_future[i] <- 0
}

# Convert covid_dummy_future to a time series object
covid_dummy_future.ts <- ts(covid_dummy_future, start = c(2023, 3), frequency = 12)

model_dummy_forecast <- forecast(ARIMA_with_Covid_dummy, xreg = covid_dummy_future.ts, h

autoplot(model_dummy_forecast)
```




```
#It does not make things better.
```

```
ARIMA_model_table_2 <- data.frame(
  Model = c("ARIMA(2,1,0)",
            "ARIMA(0,1,1)",
            "ARIMA(2,1,2)(0,0,1)[12]",
            "ARIMA(0,1,0)")
)
print(ARIMA_model_table_2)
```

```
##                     Model
## 1             ARIMA(2,1,0)
## 2             ARIMA(0,1,1)
## 3 ARIMA(2,1,2)(0,0,1)[12]
## 4             ARIMA(0,1,0)
```

Compare the AIC and BIC on data without covid_19 impact

```
#fit the model into the entire data.
Can_month_housing_sell_without_Covid.ts <- ts(Can_housing_sell_data.raw$Canada, start =

# Fit the ARIMA(2,1,0) model
model1_Without_Covid<- Arima(Can_month_housing_sell_without_Covid.ts, order=c(2,1,0))

# Fit the ARIMA(0,1,1) model
```

```
model2_Without_Covid<- Arima(Can_month_housing_sell_without_Covid.ts, order=c(0,1,1))

# Fit the ARIMA(2,1,2)(0,0,1)[12] model with zero mean
model3_Without_Covid <- Arima(Can_month_housing_sell_without_Covid.ts, order=c(2,1,2), s

# Fit the ARIMA(0,1,0) mod
model4_Without_Covid <- Arima(Can_month_housing_sell_without_Covid.ts, order=c(0,1,0))



#4.3 AIC and BIC for data without covid.
AIC_and_BIC_Matrix_for_without_covid<- matrix (ncol = 2, nrow = 4)
colnames(AIC_and_BIC_Matrix_for_without_covid) <-c("AIC","BIC")
rownames(AIC_and_BIC_Matrix_for_without_covid) <-c("ARIMA(2,1,0)", "ARIMA(0,1,1)", "ARIM
AIC_and_BIC_Matrix_for_without_covid[1,1] <- AIC(model1_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[2,1] <- AIC(model2_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[3,1] <- AIC(model3_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[4,1] <- AIC(model4_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[1,2] <- BIC(model1_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[2,2] <- BIC(model2_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[3,2] <- BIC(model3_Without_Covid)
AIC_and_BIC_Matrix_for_without_covid[4,2] <- BIC(model4_Without_Covid)
print(AIC_and_BIC_Matrix_for_without_covid)
```

```
##                               AIC      BIC
## ARIMA(2,1,0)            2515.189 2524.098
## ARIMA(0,1,1)            2514.656 2520.596
## ARIMA(2,1,2)(0,0,1)[12] 2505.988 2523.807
## ARIMA(0,1,0)            2528.349 2531.319
```

Compare the AIC and BIC on time data with covid impact

```
#fit the model into the entire data.
Can_month_housing_sell.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1), en

# ARIMA(2,1,0)
model1<- Arima(Can_month_housing_sell.ts, order=c(2,1,0))

# ARIMA(0,1,1)
model2<- Arima(Can_month_housing_sell.ts, order=c(0,1,1))

# ARIMA(2,1,2)(0,0,1)[12]
model3 <- Arima(Can_month_housing_sell.ts , order=c(2,1,2), seasonal=list(order=c(0,0,1)

# ARIMA(0,1,0)
```

```r
model4<- Arima(Can_month_housing_sell.ts , order=c(0,1,0))

#AIC and BIC for the entire data
AIC_and_BIC_Matrix <- matrix (ncol = 2, nrow = 4)
colnames(AIC_and_BIC_Matrix) <-c("AIC","BIC")
rownames(AIC_and_BIC_Matrix) <-c("ARIMA(2,1,0)", "ARIMA(0,1,1)", "ARIMA(2,1,2)(0,0,1)[12
AIC_and_BIC_Matrix[1,1] <- AIC(model1)
AIC_and_BIC_Matrix[2,1] <- AIC(model2)
AIC_and_BIC_Matrix[3,1] <- AIC(model3)
AIC_and_BIC_Matrix[4,1] <- AIC(model4)
AIC_and_BIC_Matrix[1,2] <- BIC(model1)
AIC_and_BIC_Matrix[2,2] <- BIC(model2)
AIC_and_BIC_Matrix[3,2] <- BIC(model3)
AIC_and_BIC_Matrix[4,2] <- BIC(model4)
print(AIC_and_BIC_Matrix)
```

```
##                              AIC      BIC
## ARIMA(2,1,0)            3582.467 3592.255
## ARIMA(0,1,1)            3583.642 3590.168
## ARIMA(2,1,2)(0,0,1)[12] 3579.123 3598.699
## ARIMA(0,1,0)            3618.446 3621.709
```

Cross validation one The training set 2007 - 2015 the test set 2016 - 2018

```r
#put the data here so i will not forget!
Can_month_housing_sell_test.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1

#(2023 - 2018)
n.end <- 2016 #when the training set end
n_test <- 12 * 3
# the training set end + how many month

n_models <- 4 #
h.val <- 1 # how many step forcast

#set the martix pred
pred <- matrix(rep(NA, n_test * (n_models + 1)), nrow=n_test, ncol=(n_models + 1))

# the for loop
for (i in 1: n_test ) {
  tmp0 <- 2007 #the training start at 2007
  tmp1 <- n.end + (i - 2) * (1/12) #the end of training windows

  tmp <- window(Can_month_housing_sell_test.ts, start=tmp0, end=tmp1)
  #the training data set from 2007 to the date to its en
```

```r
  #2007 to 2016 + (observation windos)


#The real value
  pred[i, 1] <- window(Can_month_housing_sell.ts, start=tmp1 + (1/12), end=tmp1 + (1/12)

#moding time!
  fit1 <- Arima(tmp, order=c(2,1,0))
  fit2 <- Arima(tmp, order=c(0,1,1))
  fit3 <- Arima(tmp, order=c(2,1,2), seasonal=list(order=c(0,0,1), period=12))
  fit4 <- Arima(tmp, order=c(0,1,0))

#one step
  pred[i, 2] <- forecast(fit1, h= h.val)$mean[h.val]
  pred[i, 3] <- forecast(fit2, h= h.val)$mean[h.val]
  pred[i, 4] <- forecast(fit3, h= h.val)$mean[h.val]
  pred[i, 5] <- forecast(fit4, h= h.val)$mean[h.val]
}

#calculate the error

error <- (pred[, -1] - pred[, 1])
#pred[.-1]
#note here: pred[, -2] does not mean using all the value expect second one...
  #it mean delete all the second col!
  #I love matrix and matrix makes me happy

mse <- colMeans(error)
#reture the mean of coln

rmse <- sqrt(colMeans(error^2))
mae <- colMeans(abs(error))
mpe <- colMeans((error/ pred[, 1]) * 100)
mape <- colMeans(abs((error/ pred[, 1]) * 100))

#The outcome
The_evil_df <- data.frame(
  Model = c("ARIMA(2,1,0)", "ARIMA(0,1,1)", "ARIMA(2,1,2)(0,0,1)[12]",  "ARIMA(0,1,0)"),
  ME = mse,
  RMSE = rmse,
  MAE = mae,
  MPE = mpe,
  MAPE = mape
)
```

```r
print(The_evil_df)
```

```
##                       Model       ME     RMSE      MAE       MPE     MAPE
## 1             ARIMA(2,1,0) 129.9997 1780.430 1142.709 0.3668394 2.718208
## 2             ARIMA(0,1,1) 144.8701 1737.525 1153.426 0.4149895 2.726496
## 3 ARIMA(2,1,2)(0,0,1)[12] 192.4608 1879.473 1203.989 0.5349698 2.865922
## 4             ARIMA(0,1,0) 186.1667 1683.713 1156.556 0.5387319 2.752935
```

Cross validation two The training set: 2007 - 2020 The test set: 2020 - 2022

```r
Can_month_housing_sell_test.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1

#(2023 - 2018)
n.end <- 2019
n_test <- 12 * 4#the times we need go
# the training set end + how many month

n_models <- 4 #
h.val <- 1 # how many step forcast

#set the martix pred
pred <- matrix(rep(NA, n_test * (n_models + 1)), nrow=n_test, ncol=(n_models + 1))
# the for loop
for (i in 1: n_test ) {
  tmp0 <- 2007 #the training start at 2007
  tmp1 <- n.end + (i - 2) * (1/12) #the end of training windows

  tmp <- window(Can_month_housing_sell_test.ts, start=tmp0, end=tmp1)
  #the training data set from 2007 to the date to its en
  #2007 to 2016 + (observation windos)


#The real value
  pred[i, 1] <- window(Can_month_housing_sell.ts, start=tmp1 + (1/12), end=tmp1 + (1/12)

#moding time!
  fit1 <- Arima(tmp, order=c(2,1,0))
  fit2 <- Arima(tmp, order=c(0,1,1))
  fit3 <- Arima(tmp, order=c(2,1,2), seasonal=list(order=c(0,0,1), period=12))
  fit4 <- Arima(tmp, order=c(0,1,0))

#one step
  pred[i, 2] <- forecast(fit1, h= h.val)$mean[h.val]
  pred[i, 3] <- forecast(fit2, h= h.val)$mean[h.val]
  pred[i, 4] <- forecast(fit3, h= h.val)$mean[h.val]
```

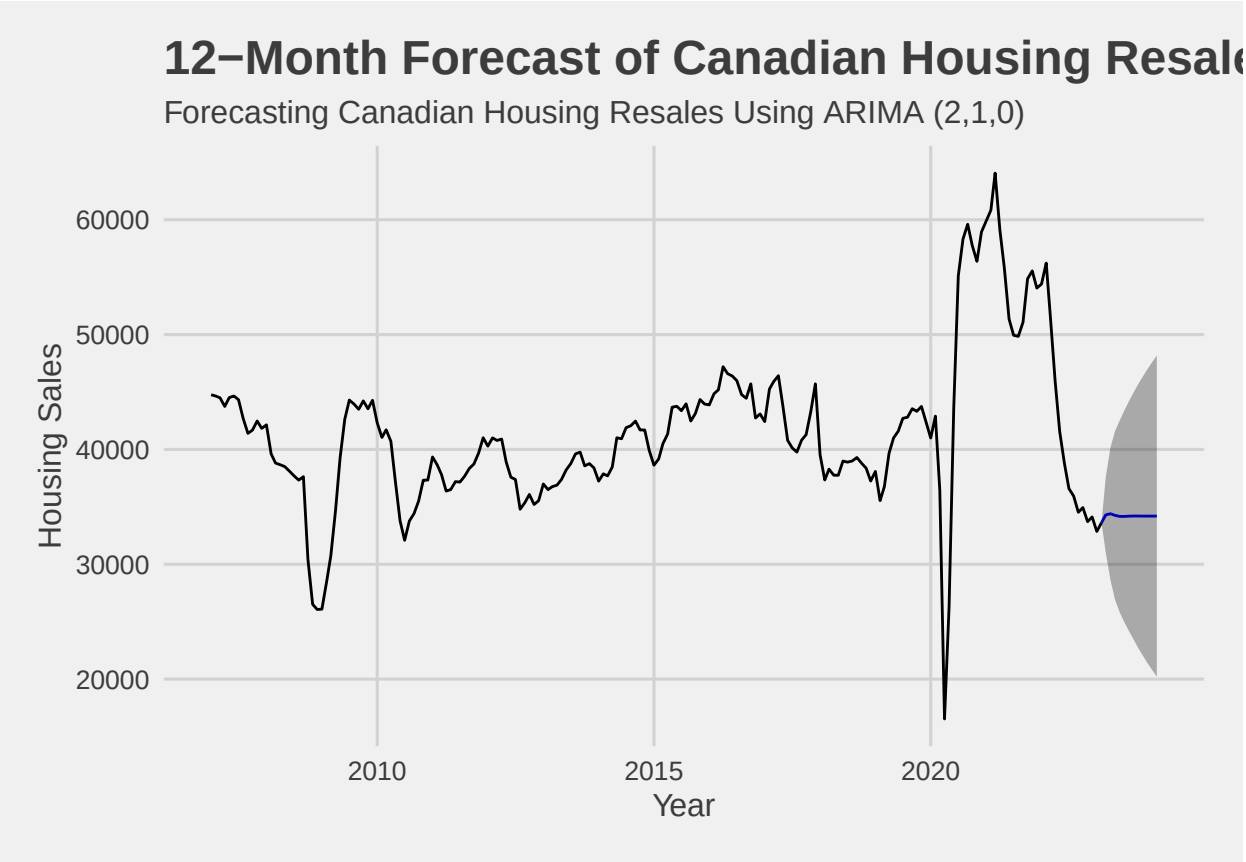
```
  pred[i, 5] <- forecast(fit4, h= h.val)$mean[h.val]
}

#calculate the error

error <- (pred[, -1] - pred[, 1])
#pred[.-1]
#note here: pred[, -2] does not mean using all the value expect second one...
  #it mean delete all the second col!
  #I love matrix and matrix makes me happy

mse <- colMeans(error)
#reture the mean of coln

rmse <- sqrt(colMeans(error^2))
mae <- colMeans(abs(error))
mpe <- colMeans((error/ pred[, 1]) * 100)
mape <- colMeans(abs((error/ pred[, 1]) * 100))

#The outcome
The_evil_df <- data.frame(
  Model = c("ARIMA(2,1,0)", "ARIMA(0,1,1)", "ARIMA(2,1,2)(0,0,1)[12]",  "ARIMA(0,1,0)"),
  ME = mse,
  RMSE = rmse,
  MAE = mae,
  MPE = mpe,
  MAPE = mape
)

print(The_evil_df)
```

```
##                     Model          ME     RMSE      MAE         MPE     MAPE
## 1            ARIMA(2,1,0) -280.15277 5299.208 3090.913 0.45058969 8.832481
## 2            ARIMA(0,1,1) -115.50045 5039.164 2836.092 0.70921596 8.269805
## 3 ARIMA(2,1,2)(0,0,1)[12] -540.56263 5341.976 3295.696 0.02977158 9.074204
## 4            ARIMA(0,1,0)   65.16667 4999.413 3008.583 1.64960807 8.407091
```

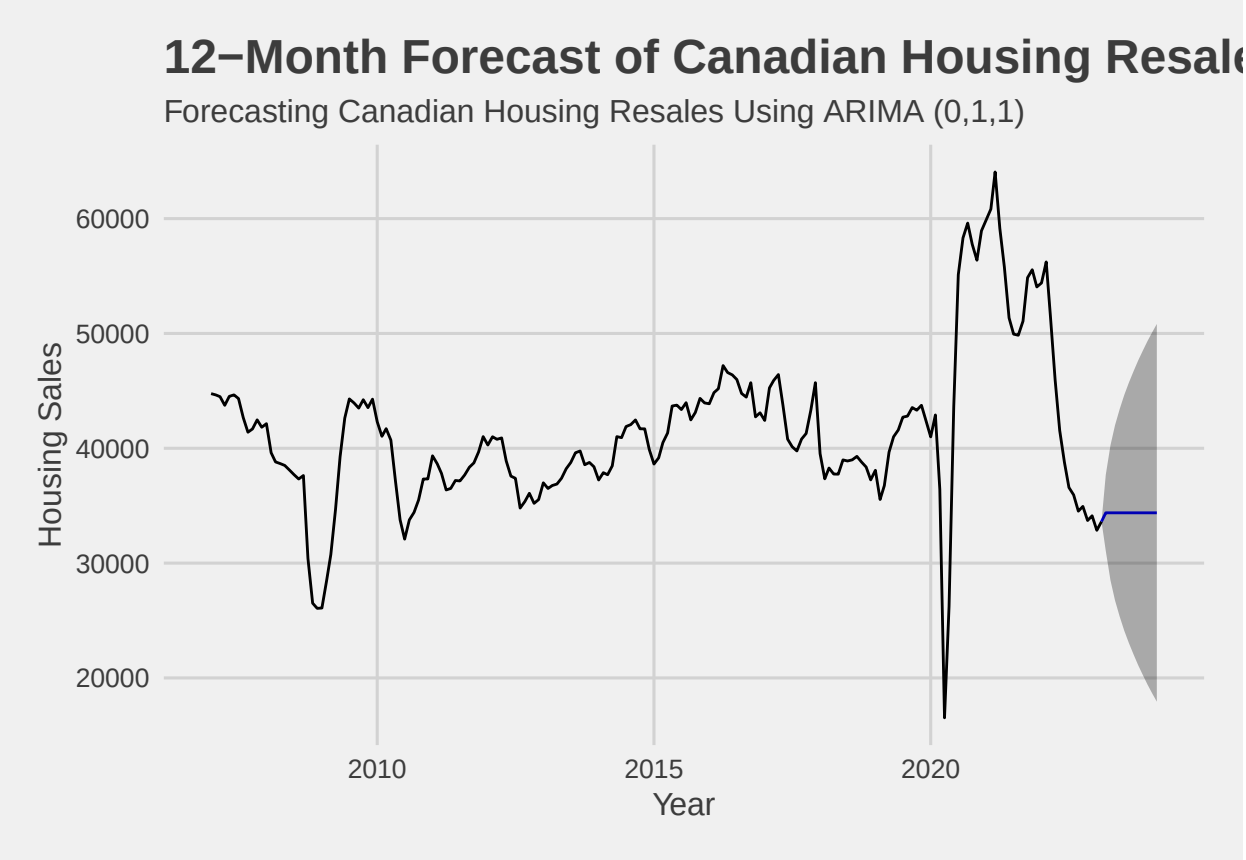
ARIMA (210) and ARIMA (011)

```
model1<- Arima(Can_month_housing_sell.ts, order=c(2,1,0))
model1_forecast <- forecast(model1, h = 12)
autoplot(model1_forecast) +
  theme_fivethirtyeight() +
  labs(title = "12-Month Forecast of Canadian Housing Resales",
       x = "Year",
```

```
      y = "Housing Sales") +
  labs(subtitle = "Forecasting Canadian Housing Resales Using ARIMA (2,1,0)") +
  theme(axis.title = element_text())
```




```
model_evil<- Arima(Can_month_housing_sell.ts, order=c(0,1,1))
model1_forecast <- forecast(model_evil, h = 12)
autoplot(model1_forecast) +
  theme_fivethirtyeight() +
  labs(title = "12-Month Forecast of Canadian Housing Resales",
       x = "Year",
       y = "Housing Sales") +
  labs(subtitle = "Forecasting Canadian Housing Resales Using ARIMA (0,1,1)") +
  theme(axis.title = element_text())
```



Part 2 VAR model

First: data collection

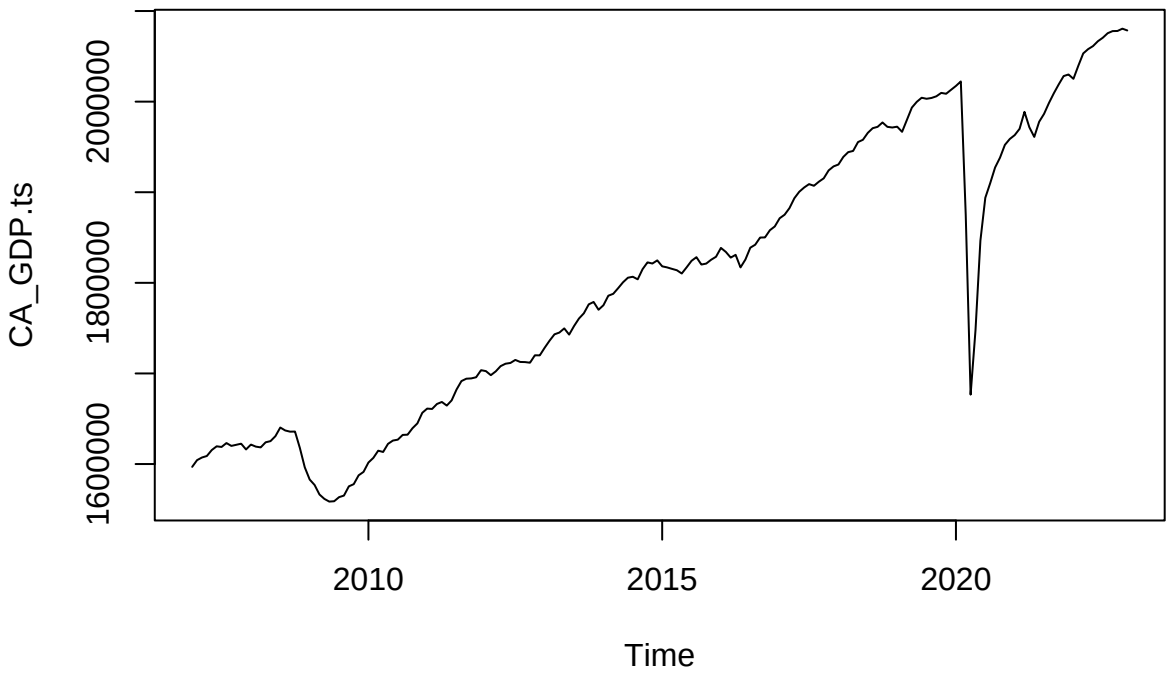
```
#Statistics Canada is pure evil!
Evil.begain <- "2007/01/01"
Evil.end <- "2022/12/31"

#The GDP data Canada [11124]; Seasonally adjusted at annual rates; 2012 constant price
CA_GDP_raw <- "v65201483"
CA_GDP.st <- get_cansim_vector(CA_GDP_raw, start_time =  "2007/01/01", end_time = "2022/
```

```
## Accessing CANSIM NDM vectors from Statistics Canada
```

```
CA_GDP_year.st <- year(CA_GDP.st$REF_DATE[1])
CA_GDP_month.st <- month(CA_GDP.st$REF_DATE[1])
#transfer data to the time series time
c(CA_GDP_year.st, CA_GDP_month.st)
```

```
## [1] 2007    1
```

```
CA_GDP.ts <-ts(CA_GDP.st$VALUE, start = c(CA_GDP_year.st, CA_GDP_month.st), freq = 12)
plot(CA_GDP.ts)
```




```
#autoplot(CA_GDP.ts)

#get the data form statistic Canada
unemployment_rate_raw <- "v2062815" #Unemployment rate (Percentage); Both sexes; 15 yea
unemployment.st <- get_cansim_vector(unemployment_rate_raw, start_time = "2008/01/01", e
```

```
## Accessing CANSIM NDM vectors from Statistics Canada
```

```
unemployment_rate_year.st <- year(unemployment.st$REF_DATE[1])
unemployment_rate_month.st <- month(unemployment.st$REF_DATE[1])
#transfer data to the time series time
c(unemployment_rate_year.st, unemployment_rate_month.st)
```

```
## [1] 2008    1
```

```
unemployment_rate.ts<- ts(unemployment.st$VALUE, start = c(unemployment_rate_year.st, un
#now its time series data!
#plot(unemployment_rate.ts)

#we are assuming we are in no late 2023/1/17
#Consumer Price Index (CPI)
CPI_raw <- "v41690914" #Consumer Price Index, monthly, seasonally adjusted, monthly (20
CPI.st <- get_cansim_vector(CPI_raw, start_time = "2007/01/01", end_time = "2022/12/31")
```

```
## Accessing CANSIM NDM vectors from Statistics Canada
```

```
CPI_year.st <- year(CPI.st$REF_DATE[1])
CPI_month.st <- month(CPI.st$REF_DATE[1])
#transfer data to the time series time
```

```r
c(CPI_year.st, CPI_month.st)
```

```
## [1] 2007    1
```

```r
CPI.ts<- ts(CPI.st$VALUE, start = c(CPI_year.st, CPI_month.st), freq = 12)
#now its time series data!
#plot(CPI.ts)

####
#The bank rate
Bank_rate_raw <- "v122530" #Financial market statistics, last Wednesday unless otherwis
Bank_rate.st <- get_cansim_vector(Bank_rate_raw, start_time = "2008/01/01", end_time = "
```

```
## Accessing CANSIM NDM vectors from Statistics Canada
```

```r
Bank_rate_year.st <- year(Bank_rate.st$REF_DATE[1])
Bank_rate_month.st <- month(Bank_rate.st$REF_DATE[1])

#transfer data to the time series time
c(Bank_rate_year.st, Bank_rate_month.st)
```

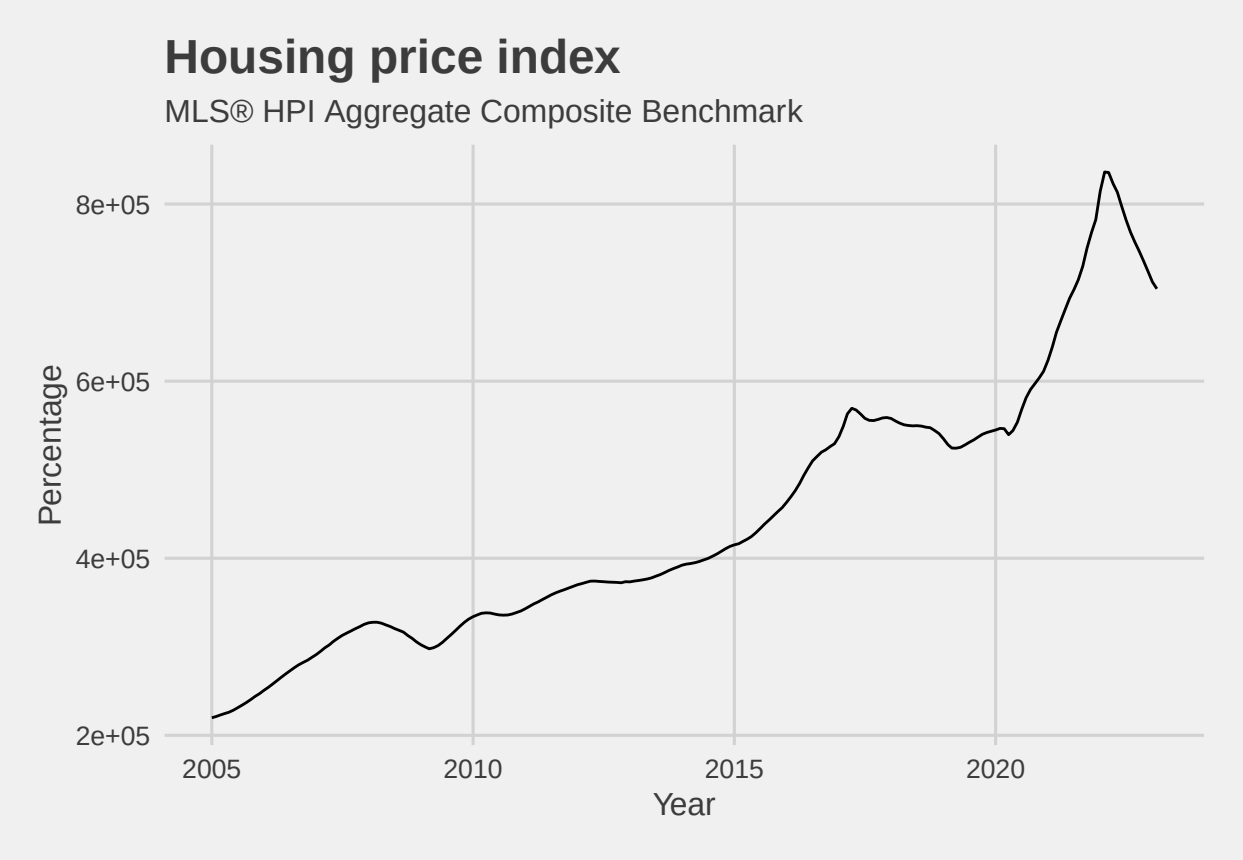
```
## [1] 2008    1
```

```r
Bank_rate.ts<- ts(Bank_rate.st$VALUE, start = c(Bank_rate_year.st, Bank_rate_month.st),
#now its time series data!
#plot(Bank_rate.ts)

Can_month_housing_sell_var<- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1), en

#########
#we are assuming we are in no late 2023/1/17
#New housing price index, monthly, monthly (Index, 201612=100)
#New_housing_price_index_raw <- "v111955442"
#New_housing_price_index.st <- get_cansim_vector(New_housing_price_index_raw, start_ti
#New_housing_price_index_year.st <- year(New_housing_price_index.st$REF_DATE[1])
#New_housing_price_index_month.st <- month(New_housing_price_index.st$REF_DATE[1])
##transfer data to the time series time
#c(New_housing_price_index_year.st, New_housing_price_index_month.st)
#New_housing_price_index.ts<- ts(New_housing_price_index.st$VALUE, start = c(New_housi
#now its time series data!
#autoplot()
#New_housing_price_index_VAR.raw <- 100* diff(log(New_housing_price_index.ts), lag = 1
# <- window(New_housing_price_index_VAR.raw, start= c(2008,1), end= c(2022,12))
```

```
#####
Can_month_housing_sell_var.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1)
##########

#### The housing price index

The_housing_price<- read_excel("/Users/tie/Documents/GitHub/ECON-493-forcasting-economy/
    sheet = "Chart 4", col_types = c("date", "skip",  "numeric", "numeric", "text"))
```

```
## New names:
## * `` -> `...3`
```

```
The_housing_price_raw.ts <- ts(The_housing_price$`MLS® HPI Aggregate Composite Benchmark


#the housing price graphy!
autoplot(The_housing_price_raw.ts) +
 theme_fivethirtyeight() +
  xlab("Year") +
  ylab("Percentage") +
  ggtitle("Housing price index") +
  labs(subtitle = "MLS® HPI Aggregate Composite Benchmark") +
  theme(axis.title = element_text())
```




```
#Variable Transformation##############################################
  #5.2.1 stationary CA_GDP.ts
stationary_CA_GDP.ts <- 100 *diff(log(CA_GDP.ts), lag = 12)
  #5.2.2 leacve unemployment rate as it is
      #unemployment_rate.ts
  #5.2.3 stationary CPI.ts
  Stationary_CPI.ts<- 100 * diff(log(CPI.ts), lag = 12)
  #5.2.4 leave bank rate as it is
  #5.2.5 stationary the housing sell
stationary_Can_month_housing_sell.ts <- 100*diff(log(Can_month_housing_sell_var.ts), lag
  #stationary the housing price index
  The_housing_price_VAR_stationary.raw <-100* diff(log(The_housing_price_raw.ts), lag =
  #cut to the suitable size
    HPI <- window(The_housing_price_VAR_stationary.raw, start= c(2008,1), end= c(2022,12


  #The prime rate
    #note: something went wrong on it
   #


#5.2.5 put all data together
```

```
all_data <- cbind(stationary_CA_GDP.ts, Stationary_CPI.ts, HPI, Bank_rate.ts, unemployme
```

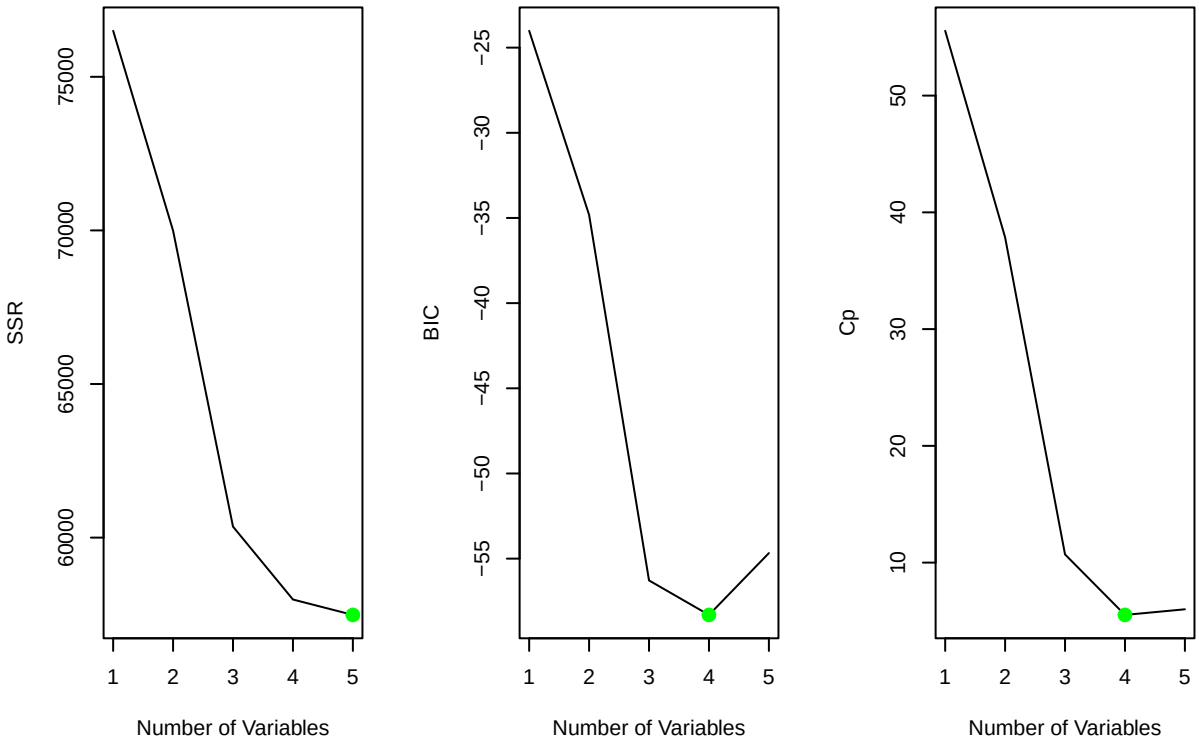
doing the best subselection why? because var model crashed....

```
#stationary_Can_month_housing_sell.ts

# Transfer data into the df
Best_subselect<- cbind(stationary_CA_GDP.ts, Stationary_CPI.ts, HPI, unemployment_rate.t

all_data_df <- data.frame(Best_subselect)


# Select specific columns from all_data_df
# need to using short verson of name...

x <- all_data_df %>%
  dplyr::select(
   SGDP = stationary_CA_GDP.ts,
   SCPI =  Stationary_CPI.ts,
   HPI,
   UI = unemployment_rate.ts,
   BK =  Bank_rate.ts
  ) %>%
  data.matrix()

data2 <- data.frame(x, stationary_Can_month_housing_sell.ts)



#best subset selection time!
regfit.all <- regsubsets(stationary_Can_month_housing_sell.ts~ ., data = data2, nvmax =
reg.summary <- summary(regfit.all)
print(reg.summary)
```

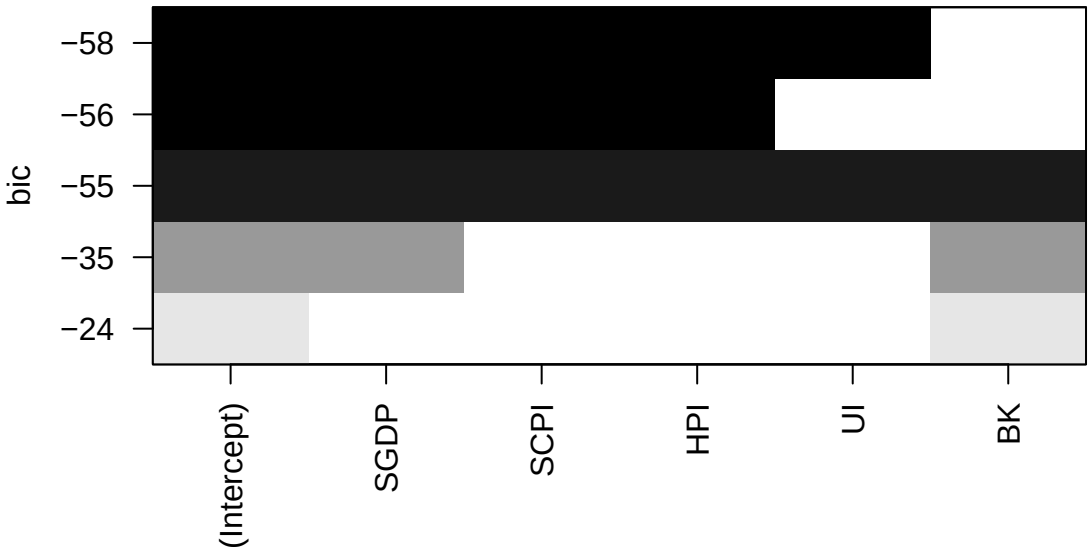
```
## Subset selection object
## Call: regsubsets.formula(stationary_Can_month_housing_sell.ts ~ .,
##     data = data2, nvmax = 100, method = "exhaustive")
## 5 Variables  (and intercept)
##      Forced in Forced out
## SGDP     FALSE      FALSE
## SCPI     FALSE      FALSE
## HPI      FALSE      FALSE
## UI       FALSE      FALSE
## BK       FALSE      FALSE
## 1 subsets of each size up to 5
```

```
## Selection Algorithm: exhaustive
##          SGDP SCPI HPI UI  BK
## 1  ( 1 ) " "  " "  " " " " "*"
## 2  ( 1 ) "*"  " "  " " " " "*"
## 3  ( 1 ) "*"  "*"  "*" " " " "
## 4  ( 1 ) "*"  "*"  "*" "*" " "
## 5  ( 1 ) "*"  "*"  "*" "*" "*"
```

```r
# Plot results
par(mfrow = c(1, 3))
plot(reg.summary$rss, xlab = "Number of Variables", ylab = "SSR", type = "l")
m.rss <- which.min(reg.summary$rss)
points(m.rss, reg.summary$rss[m.rss], col = "green", cex = 2, pch = 20)
plot(reg.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
m.bic <- which.min(reg.summary$bic)
points(m.bic, reg.summary$bic[m.bic], col = "green", cex = 2, pch = 20)
plot(reg.summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
m.cp <- which.min(reg.summary$cp)
points(m.cp, reg.summary$cp[m.cp], col = "green", cex = 2, pch = 20)
```




```r
layout(1)

# Plot the regfit.all object with BIC
plot(regfit.all, scale = "bic", title = "best subset selection")
```




```
#not working?


#transfer the data into number because time series does not working with it....
y_numeric <- as.numeric(stationary_Can_month_housing_sell.ts)


# Fit lasso model
# Why lasso?
# Because it can forcase some variable to zero...

lasso.mod <- glmnet(x, y_numeric, alpha = 1, lambda = 100^seq(3, -3, length = 1000))

# graphy time?
#I have no idea why we need this grpahy?
#autoplot(lasso.mod, xvar = "lambda", label = TRUE)

# 5 fold cross validation for lasso...
lasso.cv <- cv.glmnet(x, y_numeric, alpha = 1, nfolds = 5)

#using the the smalled to fit thegrahy....
model3 <- glmnet(x, y_numeric, alpha = 1, lambda = lasso.cv$lambda.min)

# lasso! lasso! lasso!
coef3 <- coef(model3)


print(coef3)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
```

```
##                       s0
## (Intercept) -11.9415156
## SGDP          3.1616785
## SCPI         -6.2747824
## HPI           0.6281255
## UI            2.7624375
## BK           -3.3871335
```

```
# only one variable?
# only one?
```

#The backward search suggest that SCPIand HP are variable been picked

```
#base on the cauality test pick the best variable for further evoluation.

#the Var data
Var_all_data <- cbind(stationary_Can_month_housing_sell.ts, Stationary_CPI.ts, HPI, stat


#leg selection
#VARselect(Var_all_data, type="const", lag.max = 11)
var_model_test <- VAR(Var_all_data, p = 3, type="const")
#roots(var_model)
#serial.test(var_model_test, lags.pt = 10, type = "PT.asymptotic")



# doing the granger test....
housing_CPI_pvalue <- causality(var_model_test, cause = "Stationary_CPI.ts")$Granger$p.v
housing_houing_price_pvalue <- causality(var_model_test, cause = "HPI")$Granger$p.value
housing_gdp_pvalue <- causality(var_model_test, cause = "stationary_CA_GDP.ts")$Granger$
housing_unemployment_pvalue <- causality(var_model_test, cause = "unemployment_rate.ts")

# put the p value into the matrix
pv_matrix <- matrix(c(housing_CPI_pvalue,
                      housing_houing_price_pvalue,
                      housing_gdp_pvalue,
                      housing_unemployment_pvalue),
                    nrow = 1, ncol = 4, byrow = TRUE)

# Set the column names of the matrix
colnames(pv_matrix) <- c("CPI", "HPI", "GDP", "Unemployment")
rownames(pv_matrix) <- c("p-value")

# Print the matrix
print(pv_matrix)
```

```
##                  CPI          HPI          GDP Unemployment
## p-value 0.006568881 2.803769e-06 1.154632e-14    0.3391877
```

```
Var_final_data <- cbind(stationary_Can_month_housing_sell.ts, Stationary_CPI.ts, HPI, st

#The covid dummy variable
The_covid_dummy_VAR.ts <- ts(The_covid_dummy$Covid_dummy, start = c(2008, 1), end = c(20

#put bank rate with covid dummy variable together.
exogen_variable <- cbind(The_covid_dummy_VAR.ts, Bank_rate.ts)

#we can see how the bank rate and covid dummy variable impact on the entire housing re


VARselect(Var_final_data, exogen = exogen_variable,  type="const", lag.max = 11)
```

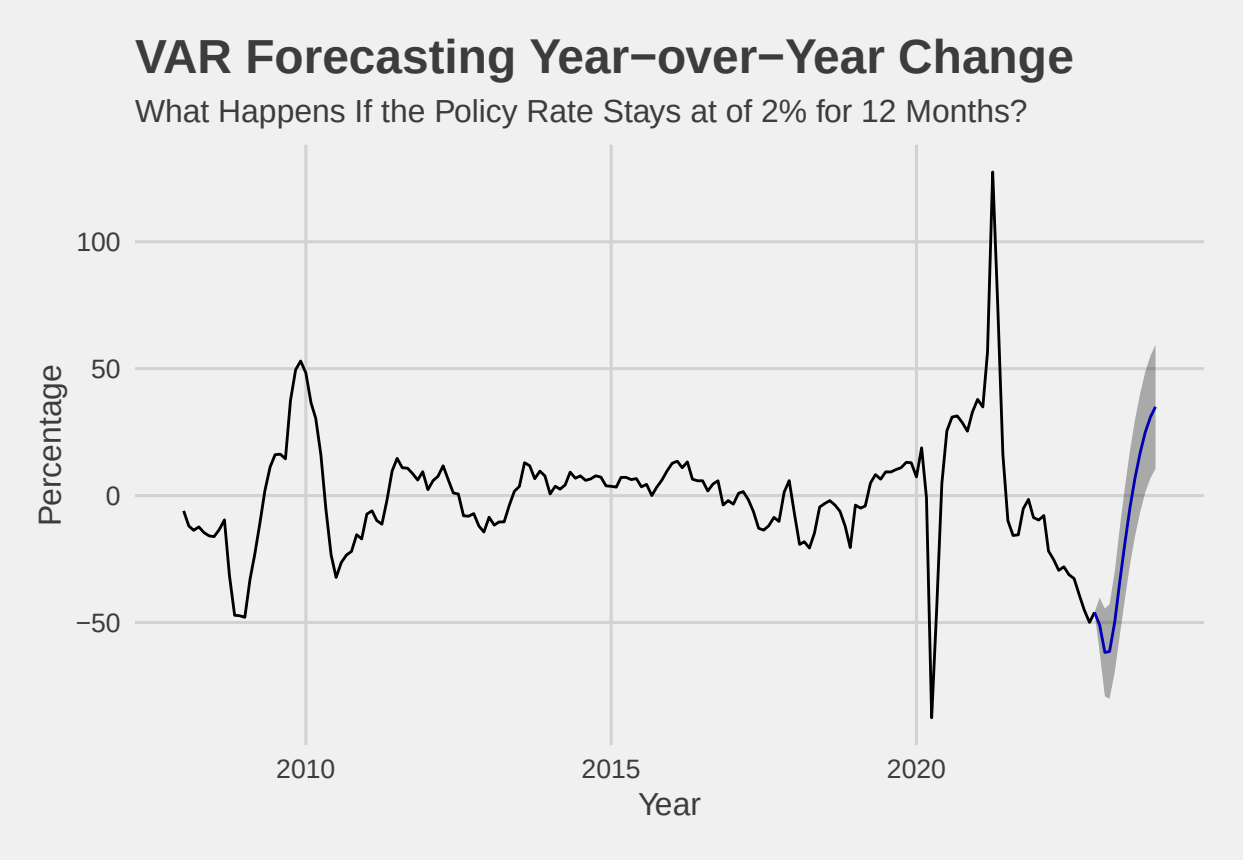
```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      4      3      2      4
##
## $criteria
##                1         2         3         4         5         6         7
## AIC(n) 2.255728 0.7574027 0.5179802 0.4800431 0.5739332 0.6628192 0.7038097
## HQ(n)  2.466171 1.0880984 0.9689288 1.0512446 1.2653877 1.4745266 1.6357701
## SC(n)  2.774291 1.5722876 1.6291868 1.8875715 2.2777834 2.6629911 3.0003034
## FPE(n) 9.544050 2.1343026 1.6817815 1.6223269 1.7873442 1.9618013 2.0556843
##                8         9       10        11
## AIC(n) 0.7396418 0.7542875 0.749676 0.6186805
## HQ(n)  1.7918551 1.9267538 2.042395 2.0316527
## SC(n)  3.3324573 3.6434248 3.935135 4.1004613
## FPE(n) 2.1467559 2.1993199 2.215306 1.9716246
```

```
var_final <- VAR(Var_final_data, exogen = exogen_variable, p = 3, type="const")

serial.test(var_final, lags.pt = 10, type = "PT.asymptotic")
```

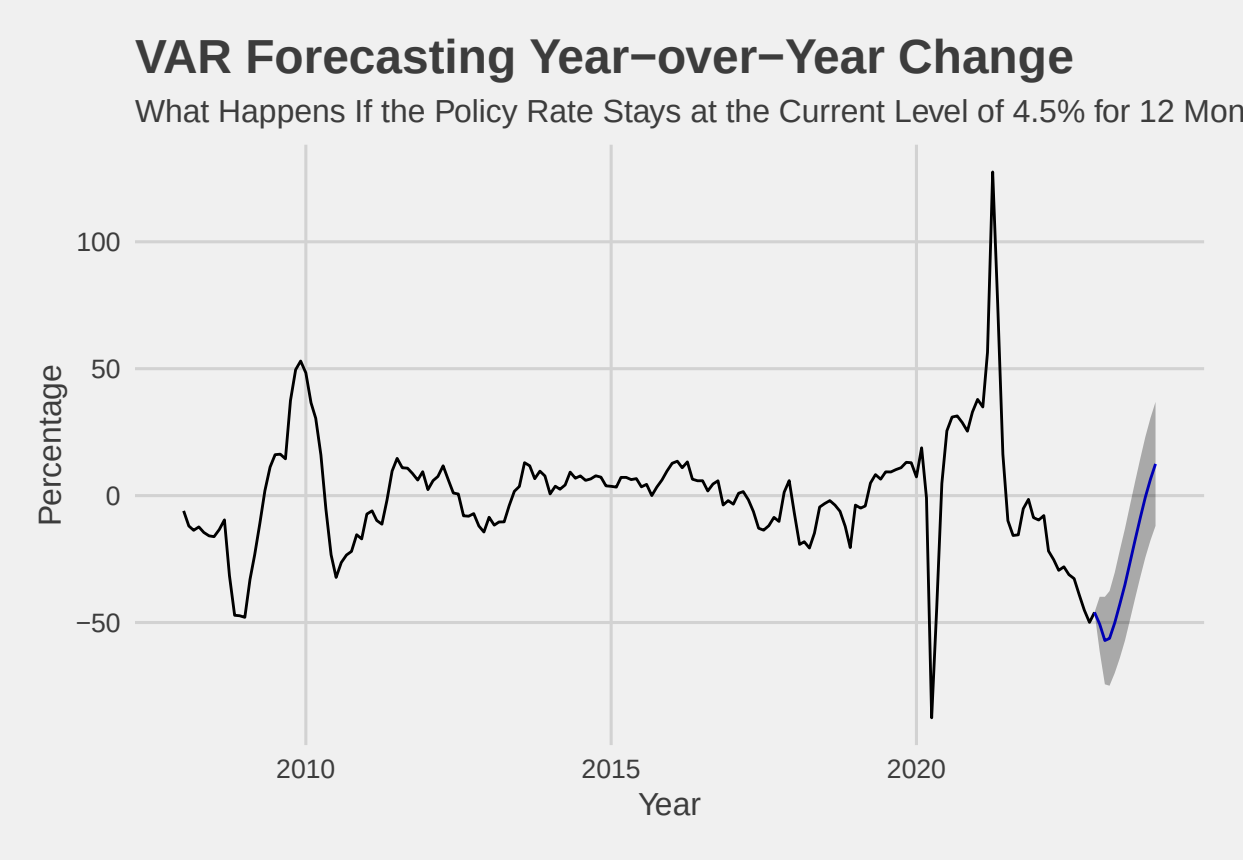
```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var_final
## Chi-squared = 128.27, df = 112, p-value = 0.1395
```

```
###ready for the dummy varialbe

### The forecasting
The_future_covid_dummy.ts <-ts(rep(0, 12), start = c(2022, 12), frequency = 12)
```

```r
#The bank rate set at 2%
Bank_rate_1  <-ts(rep(2, 12), start = c(2022, 12), frequency = 12)

#The bank rate set at 4.5%
Bank_rate_2  <-ts(rep(4.5, 12), start = c(2022, 12), frequency = 12)


##########
Var_exogen_one <- as.matrix(cbind(The_future_covid_dummy.ts, Bank_rate_1))

##########
Var_exogen_two <- cbind(The_future_covid_dummy.ts, Bank_rate_2)


######## now its conditonal forecasting
#forget to change the colname

colnames(Var_exogen_one) <- colnames(exogen_variable)
colnames(Var_exogen_two) <- colnames(exogen_variable)


#forecasting_Time!
Var_forecasting_one <- forecast(var_final, dumvar = Var_exogen_one , h = 12 )

Var_forecasting_two <- forecast(var_final, dumvar = Var_exogen_two , h = 12)

# Print the results
autoplot(Var_forecasting_one$forecast$stationary_Can_month_housing_sell.ts) +
  theme_fivethirtyeight() +
  xlab("Year") +
  ylab("Percentage") +
  ggtitle("VAR Forecasting Year-over-Year Change") +
  labs(subtitle = "What Happens If the Policy Rate Stays at of 2% for 12 Months?") +
  theme(axis.title = element_text())
```




```
autoplot(Var_forecasting_two$forecast$stationary_Can_month_housing_sell.ts)+
 theme_fivethirtyeight() +
  xlab("Year") +
  ylab("Percentage") +
  ggtitle("VAR Forecasting Year-over-Year Change") +
  labs(subtitle = "What Happens If the Policy Rate Stays at the Current Level of 4.5% fo
  theme(axis.title = element_text())
```



now, I can finally rest in the peace

Tie Ma 2023/04/15

The additional part not included in the paper from here to part 2 var forecasting are unused code. you can just jump to part 2 var model. . . .

The conditional forecasting 1 base on ARIMA but give up on it due to the frecasting almost same

```
Can_month_housing_sell.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007, 1), en

 #V80691311
 Bank_rate_ARIMA_raw <- "v122530" #Financial market statistics, last Wednesday unless c
 Bank_rate_ARIMA.st <- get_cansim_vector( Bank_rate_ARIMA_raw, start_time = "2008/01/01"
```

```
## Reading CANSIM NDM vectors from temporary cache
```

```
 Bank_rate_ARIMA_year.st <- year( Bank_rate_ARIMA.st$REF_DATE[1])
 Bank_rate_ARIMA_month.st <- month( Bank_rate_ARIMA.st$REF_DATE[1])
c( Bank_rate_ARIMA_year.st,  Bank_rate_ARIMA_month.st)
```

```
## [1] 2008    1
```

```
 Bank_rate_ARIMA.ts<- ts( Bank_rate_ARIMA.st$VALUE, start = c( Bank_rate_ARIMA_year.st,
################
```

The conditional forecasting 2: The CPI with the bank rate (The plane B for the ARIMA forecasting)

```
#we are assuming we are in no late 2023/1/17
#Consumer Price Index (CPI)
CPI_ARIMA.raw <- "v41690914" #Consumer Price Index, monthly, seasonally adjusted, month
CPI_ARIMA.st <- get_cansim_vector(CPI_ARIMA.raw, start_time = "2007/01/01", end_time = "
```

```
## Reading CANSIM NDM vectors from temporary cache
```

```
CPI_ARIMA_year.st <- year(CPI_ARIMA.st$REF_DATE[1])
CPI_ARIMA_month.st <- month(CPI_ARIMA.st$REF_DATE[1])
#transfer data to the time series time
c(CPI_ARIMA_year.st, CPI_ARIMA_month.st)
```

```
## [1] 2007    1
```

```
CPI_ARIMA.ts<- ts(CPI_ARIMA.st$VALUE, start = c(CPI_ARIMA_year.st, CPI_ARIMA_month.st),
#now its time series data!
#plot(CPI.ts)
####
```

```
Can_month_housing_sell_ARIMA.ts <- ts(Can_housing_sell_data.raw$Canada, start = c(2007,

###### station CPI and housing resale#
#stationary_CPI_ARIMA.ts <- (100 * diff(log(CPI_ARIMA.ts), lag = 12))
Stationary_housing_resale_ARIMA.ts <- 100* diff(log(Can_month_housing_sell_ARIMA.ts), la
#adf.test(Stationary_housing_resale_ARIMA.ts)
Housing_sell_Stationary_CPI.ts <- cbind(Stationary_housing_resale_ARIMA.ts, Bank_rate_AR
```

Try the lag test

```
# The_lage_test
BankRate <- cbind(
  BankRateLag0 = Housing_sell_Stationary_CPI.ts[,"Bank_rate_ARIMA.ts"],
  BankRateLag1 = stats::lag(Housing_sell_Stationary_CPI.ts[,"Bank_rate_ARIMA.ts"], 1),
  BankRateLag2 = stats::lag(Housing_sell_Stationary_CPI.ts[,"Bank_rate_ARIMA.ts"], 2),
  BankRateLag3 = stats::lag(Housing_sell_Stationary_CPI.ts[,"Bank_rate_ARIMA.ts"], 3),
  BankRateLag4 = stats::lag(Housing_sell_Stationary_CPI.ts[,"Bank_rate_ARIMA.ts"], 4)
) %>% head(NROW(Housing_sell_Stationary_CPI.ts))

# Fit ARIMA models
fit1 <- auto.arima(Housing_sell_Stationary_CPI.ts[4:40, 1], xreg=BankRate[4:40, 1], stat
fit2 <- auto.arima(Housing_sell_Stationary_CPI.ts[4:40, 1], xreg=BankRate[4:40, 1:2], st
fit3 <- auto.arima(Housing_sell_Stationary_CPI.ts[4:40, 1], xreg=BankRate[4:40, 1:3], st
```

```
fit4 <- auto.arima(Housing_sell_Stationary_CPI.ts[4:40, 1], xreg=BankRate[4:40, 1:4], st

# Show AICc values
cat("AICc values:", fit1[["aicc"]], fit2[["aicc"]], fit3[["aicc"]], fit4[["aicc"]], "\n"
```

```
## AICc values: 258.4383 261.1235 262.9175 265.9327
```

```
#only lag one?
#I spend 40 min it is only lag one?
```

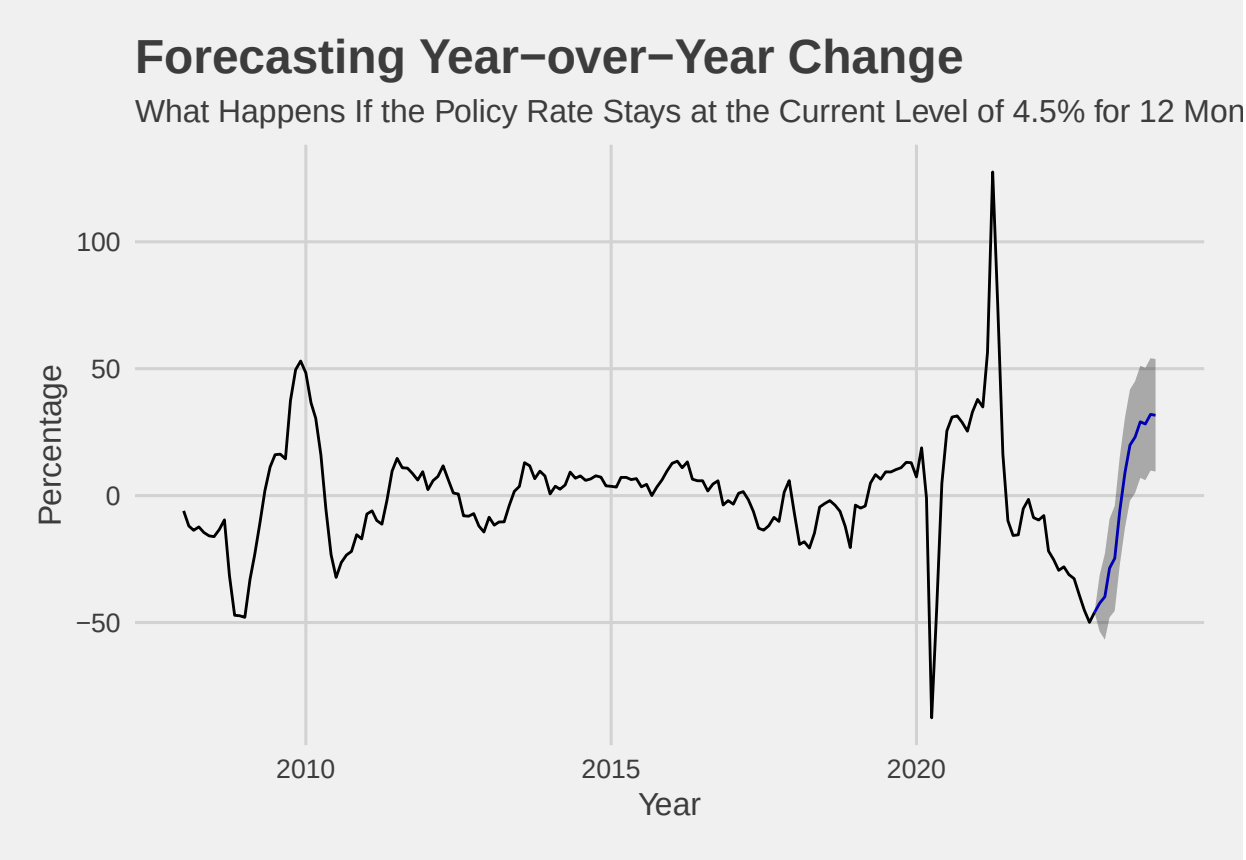
now is generate the dynamatic regression's acutally arima model.

```
The_conditional_forecasting_2 <- auto.arima(Stationary_housing_resale_ARIMA.ts, xreg = B

print(The_conditional_forecasting_2)
```

```
## Series: Stationary_housing_resale_ARIMA.ts
## Regression with ARIMA(1,0,1)(2,0,1)[12] errors
##
## Coefficients:
##          ar1     ma1     sar1     sar2     sma1    xreg
##       0.7472  0.3927  -0.1038  -0.0968  -0.8451  0.6709
## s.e.  0.0581  0.0787   0.1196   0.1110   0.1176  0.6029
##
## sigma^2 = 75.84:  log likelihood = -652.05
## AIC=1318.1   AICc=1318.75   BIC=1340.45
```

What if the policy rate remain 4.5 for next 12 month

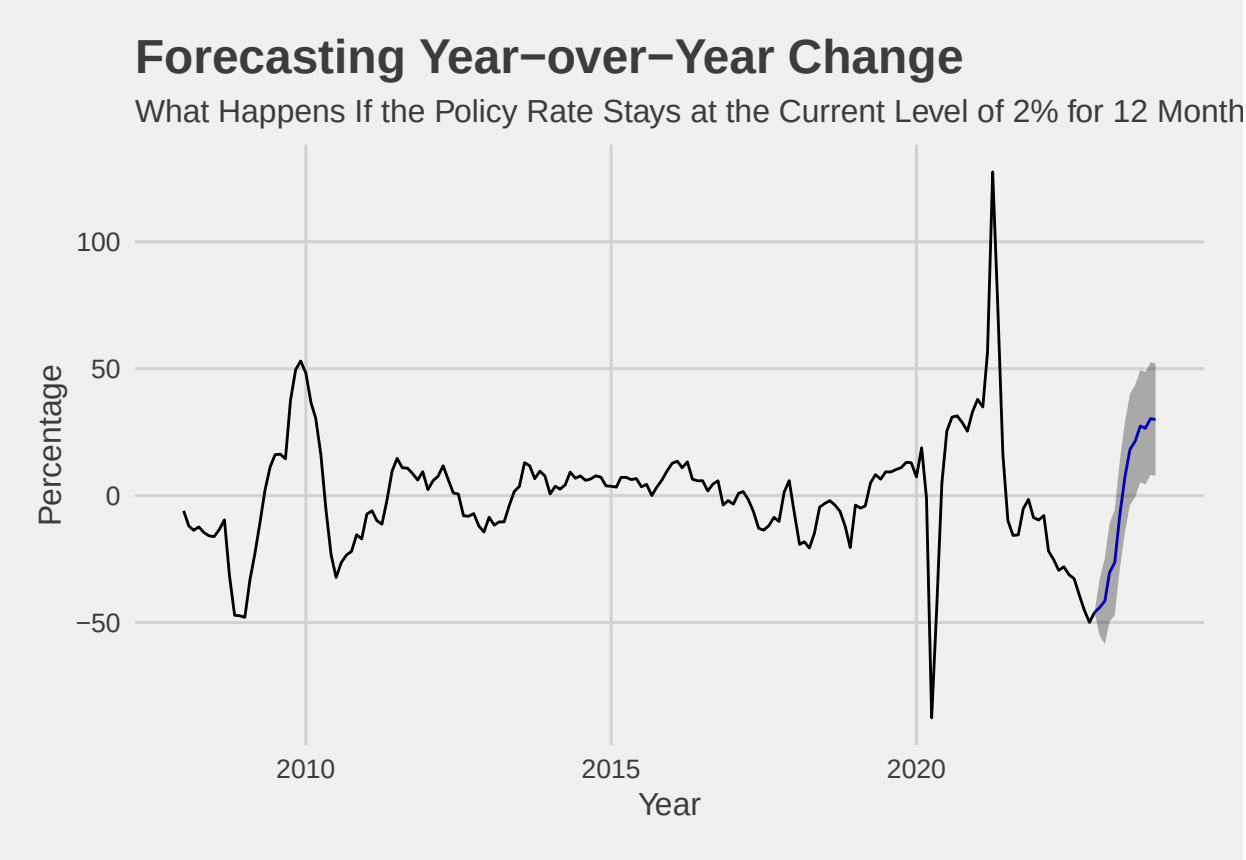
```
Stationary_Bank_ARIMA_forecasting_1.ts <- matrix(rep(4.5, times = 12), ncol = 1)


Stationary_Bank_ARIMA_forecast_1.ts <- matrix(rep(4.5, times = 12), ncol = 1)

autoplot(forecast(The_conditional_forecasting_2,
                  h = 12,
                  xreg = Stationary_Bank_ARIMA_forecast_1.ts)) +
  theme_fivethirtyeight() +
  xlab("Year") +
  ylab("Percentage") +
  ggtitle("Forecasting Year-over-Year Change") +
  labs(subtitle = "What Happens If the Policy Rate Stays at the Current Level of 4.5% fo
  theme(axis.title = element_text())
```




```
#why is it not performing well?


#it not working very good?
#but why?
```

What if the policy rate remain 2 for next 12 month (the average in 2019)

```
Stationary_Bank_ARIMA_forecasting_2.ts <- matrix(rep(2, times = 12), ncol = 1)


autoplot(forecast(The_conditional_forecasting_2,
                  h = 12,
                  xreg = Stationary_Bank_ARIMA_forecasting_2.ts)) +
  theme_fivethirtyeight() +
  xlab("Year") +
  ylab("Percentage") +
  ggtitle("Forecasting Year-over-Year Change") +
  labs(subtitle = "What Happens If the Policy Rate Stays at the Current Level of 2% for
  theme(axis.title = element_text())
```




```
#it not working very good?
#but why?
```