**TIME SERIES ANALYSIS PROJECT**

**Ruairi Kenealy, C20359883**

Contents

[Title Part 2, Seasonal Time Series Analysis: 1](#_Toc101093147)

[Executive summary 1](#_Toc101093148)

[Exploratory Data Analysis 3](#_Toc101093149)

[Sources for Data sets 3](#_Toc101093150)

[Data Last Updated 4](#_Toc101093151)

[Labels on Graphs 4](#_Toc101093152)

[Software used 4](#_Toc101093153)

[R Packages used 4](#_Toc101093154)

[Reading the data 4](#_Toc101093155)

[Initial plot 5](#_Toc101093156)

[Holdout 5](#_Toc101093157)

[Smoothing the data 6](#_Toc101093158)

[Fitting a moving average, kernel smoothing, loess and splines smoothing: 6](#_Toc101093159)

[Decomposing the data 8](#_Toc101093160)

[Is the data stationary? 12](#_Toc101093161)

[Differencing 12](#_Toc101093162)

[Choosing the Model 16](#_Toc101093163)

[Holt Winters 20](#_Toc101093164)

[Model evaluation and forecasting Accuracy 24](#_Toc101093165)

[Forecasting using the chosen model(s) 24](#_Toc101093166)

[Seasonal ARIMA 24](#_Toc101093167)

[Conclusion 28](#_Toc101093168)

# Title Part 2, Seasonal Time Series Analysis:

## Executive summary

I chose the analysis of this data because I know I’ll be looking to buy an apartment so I’ve an interest in checking out the market. I also thought it would be a good idea to see what’s going on in a country outside of Ireland to see if it’s similar to here.

This study was done to examine the Quebec condo sales in Montreal. I found the data from a blog posted by Spencer Hayes in June of 2021, where in that blog there was a link containing the csv file of the information on prices of the condo sales in Quebec and the data was published by the Quebec professional Association of real estate Brokers. The data was observed from 2014 to 2021. The data is seasonal and observed month by month over 8 years.

The data was smoothed out using various smoothing methods such as moving average, kernel smoothing, LOESS and GAM. All these methods defined and fitted onto a plot made the data much more readable and easier to predict. To check if the data was stationary or not, ADF and ACF tests were used on the dataset and since the data was non-stationary differencing was applied in an attempt to transform and make the data more stationary.

A holdout was created from 2020 to 2021 which could be used to compare with the forecast on data from 2014-2019. All analysis on the data is based on the observations from 2014-2019. I fitted 4 different SARIMA models and 2 exponential Holt Winters models to fit to the data. I tested their accuracy and chose both the 4th SARIMA and ETS model for the forecast. That forecast matches the holdout plot.

The graphs from plotting the data show a sharp rise in median prices from 2019 onwards, no doubt the spread of COVID had a part to play in this though prices were steadily increasing from 2014. The prices also seemed to spike in Spring, suggesting the prices were increased at that time of year because the real estate market is traditionally active in Spring. On average the prices were 5% higher in the spring so that’s a good time to sell. The sharp rise in prices in 2020 and 2021 is mirrored in the rest of the Canadian market – see [here](https://apciq.ca/en/montreals-residential-real-estate-market-compared-to-other-major-canadian-markets/).

# Exploratory Data Analysis

## Sources for Data sets

The original source of the data is the Quebec Association of Realtors website which has data from 2014 to March 2022 here:

<https://apciq.ca/en/real-estate-market/statistiques-mensuelles-detaillees/>

This data was collated into a Google docs spreadsheet by Spencer Hayes in May 2021. See:

<https://towardsdatascience.com/finding-seasonal-trends-in-time-series-data-with-python-ce10c37aa861>

Here is the Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Date | Province\_Median\_Price | Province\_Average\_Time | Montreal\_Median\_Price | Montreal\_Average\_Time | |
| 01/01/2014 | 210000 | 135 | 220000 | 124 |  |
| 02/01/2014 | 212000 | 128 | 225000 | 117 |  |
| 03/01/2014 | 210000 | 130 | 225000 | 118 |  |
| 04/01/2014 | 214000 | 120 | 225000 | 113 |  |
| 05/01/2014 | 215000 | 125 | 226287 | 115 |  |
| 06/01/2014 | 220000 | 129 | 234525 | 122 |  |
| 07/01/2014 | 223000 | 130 | 235000 | 128 |  |
| 08/01/2014 | 214000 | 139 | 230150 | 128 |  |
| 09/01/2014 | 220000 | 134 | 234000 | 121 |  |
| 10/01/2014 | 221000 | 136 | 249000 | 119 |  |
| 11/01/2014 | 218000 | 132 | 230750 | 120 |  |
| 12/01/2014 | 215000 | 134 | 235000 | 129 |  |
| 13/01/2014 | 214000 | 134 | 230485 | 129 |  |
| 14/01/2014 | 212000 | 139 | 230000 | 125 |  |
| 15/01/2014 | 218500 | 132 | 235000 | 120 |  |
| 16/01/2014 | 215000 | 130 | 231950 | 123 |  |
| 17/01/2014 | 224900 | 127 | 239000 | 122 |  |
| 18/01/2014 | 224500 | 142 | 240000 | 118 |  |
| 19/01/2014 | 225000 | 127 | 245000 | 119 |  |

Data Source, Headers, and Labels

Headers in the data:

Date

Province\_median\_price

Province\_average\_time

Montreal\_median\_price

Montreal\_average\_time

Values:

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

2014 220.000 225.000 225.000 225.000 226.287 234.525 235.000 230.150 234.000 249.000 230.750 235.000

2015 230.485 230.000 235.000 231.950 239.000 240.000 245.000 240.000 248.000 240.000 239.000 240.000

2016 238.000 226.000 231.600 235.000 240.000 245.000 250.000 250.000 250.000 245.000 248.000 245.000

2017 240.000 238.900 243.232 240.000 243.000 257.500 256.000 251.500 253.000 255.000 254.510 265.000

2018 244.750 250.000 240.000 245.350 257.000 256.000 265.000 265.000 263.000 265.000 264.900 272.863

2019 248.271 250.000 248.000 260.000 261.000 261.000 278.600 279.000 290.000 280.000 290.000 285.000

2020 275.000 275.000 287.000 289.900 280.000 305.000 312.000 312.000 318.000 323.500 316.000 325.000

2021 322.000 340.000 347.065 357.750 365.000

## Data Last Updated

May 2021

## Labels on Graphs

Y axis: Sale Price (thousands)

X axis: Year

## Software used

R Studio version: 2022.02.1 Build 461

R.version(): R version 4.1.3 (2022-03-10)"

Excel version: Microsoft 365

## R Packages used

library(readr)

library(forecast)

library(tseries)

library(mgcv)

## Reading the data

setwd("C:/Users/ruair/Downloads/Forecasting\_Time\_series")

condo\_sales = read.csv("quebec\_real\_estate.csv",header=T)

str(condo\_sales)

## Initial plot

values = condo\_sales[4]

values = ts(values, start=c(2014,1), frequency=12)

ts.plot(values, main="Montreal Median Price", ylab="Number of People", type="l")

Chart

Description automatically generated

## Holdout

I created a holdout for 2020 to 2021 in order to be able to see how observed data compares to forecasted data, though for much of the other analysis the data runs all the way to 2021.

values.holdout <- window(values, start=2020,end=2021)

values <- window(values, start=1967,end=2019)

ts.plot(cbind(values, values.holdout), main="Quebec condo market ",

ylab="Price (1000's)", type="l", col=c("red", "blue"), lty=c(1, 2))

Chart, line chart, histogram

Description automatically generated

# Smoothing the data

## Fitting a moving average, kernel smoothing, loess and splines smoothing:

Smoothing was done in order to view and analyse the observed data in an easier and more predictable way, this was done through the various smoothing methods defined below, moving average(MA), kernel smoothing, etc.

Time points for fitting trend:

time.pts = c(1:length(values))

time.pts = c(time.pts - min(time.pts))/max(time.pts)

Defining smoothing methods:

values.mafilter.fit = filter(values, filter = rep(1/4, 4), sides = 2)

MA:

ma.fit = ma(values, order=2, centre=TRUE)

values.fit.ma = ts(ma.fit, start=c(2008, 1), frequency=12)

Kernel:

ksmooth.fit = ksmooth(time.pts, values, kernel = "box", bandwidth = 0.2)

values.fit.ksmooth = ts(ksmooth.fit$y,start=c(2008, 1),frequency=12)

LOESS:

loess.fit = loess(as.matrix(values)~time.pts, data=values, span=0.2)

values.fit.loess = ts(predict(loess.fit), start=c(2008, 1), frequency=12)

GAM:

gam.fit = gam(values~s(time.pts))

values.fit.gam = ts(fitted(gam.fit),start=c(2008, 1),frequency=12)

plotting smoothing values:

lines(values.fit.ksmooth,lwd=1, lty=4 ,col="purple")

lines(values.mafilter.fit, col="red")

lines(values.fit.loess, col="orange", lty=4)

lines(values.fit.gam, col="violet")

lines(values.fit.ma, col="cyan", lwd=3)

legend(x="topleft",c("kernel smoothing", "filter", "loess", "gam", "mav"),lty = c(4, 1), col=c("purple"))

ts.plot(values,ylab="sale price", main="Observed Values vs smoothing methods")

Chart, histogram

Description automatically generated

# Decomposing the data

Decomposing a time series separates out trend, seasonality and residual/random/white noise. There are two forms of classical decomposition- additive and multiplicative. The R decompose() function performs classical seasonal decomposition using moving averages. We assume that this time series is multiplicative because the periods appear to be getting bigger.

values.ma.decomp = decompose(values.fit.ma, type=c("multiplicative"))

plot(values.ma.decomp)

A more detailed decomposition of this time-series can be seen below by using STL - Seasonal Decomposition of Time Series by LOESS:Chart, line chart

Description automatically generated

STL is a versatile and robust method for decomposing time series. STL is an acronym for “Seasonal and Trend decomposition using Loess”, while loess is a method for estimating nonlinear relationships. The STL method was developed by R. B. Cleveland et al.  See plot below for the output produced using STL.

values.decomp.stl <- stl(na.omit(values.fit.ma[,1]), s.window="periodic")

plot(values.decomp.stl)

Chart

Description automatically generated

The R deseason function calculates anomalies of a RasterStack by supplying a suitable seasonal window. E. g. to create monthly anomalies of a raster stack of 12 layers per year, use **cycle.window = 12**.

values.deseason = seasadj(values.decomp.stl)

plot(values.deseason, main="Deseasonalized time series")

Chart, line chart, histogram

Description automatically generated

We can also print out the values from the STL.

print(values.decomp.stl)

Call:

stl(x = na.omit(values.fit.ma[, 1]), s.window = "periodic")

Components

seasonal trend remainder

Feb 2014 -6.936089 229.1336 1.55246856

Mar 2014 -7.056369 229.5972 2.45913934

Apr 2014 -5.733749 230.0608 0.99466048

May 2014 -1.934334 230.5758 -0.61675384

Jun 2014 2.627983 231.0908 -1.13456889

Jul 2014 4.644695 231.6358 -2.61172578

Aug 2014 4.225757 232.1807 -4.08148268

Sep 2014 4.513514 232.7354 -0.46146416

Oct 2014 4.409271 233.2902 2.98805487

Nov 2014 3.492821 234.0432 -1.16103982

Dec 2014 1.335672 234.7963 -3.32318506

Jan 2015 -3.589169 235.5897 -0.50801877

Feb 2015 -6.936089 236.3831 1.92422656

Mar 2015 -7.056369 236.9564 3.08750430

Apr 2015 -5.733749 237.5296 2.67913241

May 2015 -1.934334 237.8559 1.56588666

Jun 2015 2.627983 238.1823 0.18974016

Jul 2015 4.644695 238.2993 -0.44395399

Aug 2015 4.225757 238.4162 0.60800185

Sep 2015 4.513514 238.4053 1.08114267

Oct 2015 4.409271 238.3944 -1.05371601

Nov 2015 3.492821 238.5574 -2.55020911

Dec 2015 1.335672 238.7203 -0.80600275

Jan 2016 -3.589169 239.1497 -0.06057395

Feb 2016 -6.936089 239.5792 -2.24306611

Mar 2016 -7.056369 240.1298 -2.02344054

Apr 2016 -5.733749 240.6805 0.45328538

May 2016 -1.934334 241.2466 0.68774767

Jun 2016 2.627983 241.8127 0.55930921

Jul 2016 4.644695 242.3900 1.71528595

Aug 2016 4.225757 242.9673 2.80691267

Sep 2016 4.513514 243.5654 0.67107937

Oct 2016 4.409271 244.1635 -1.57275343

Nov 2016 3.492821 244.7167 -1.70950026

Dec 2016 1.335672 245.2699 -2.10554762

Jan 2017 -3.589169 245.7949 -1.23070026

Feb 2017 -6.936089 246.3199 0.87422615

Mar 2017 -7.056369 246.8925 1.50487763

Apr 2017 -5.733749 247.4651 -0.17337052

May 2017 -1.934334 248.2026 -0.39324559

Jun 2017 2.627983 248.9400 1.93197859

Jul 2017 4.644695 249.6519 0.95336231

Aug 2017 4.225757 250.3638 -1.58960398

Sep 2017 4.513514 250.8857 -2.27418933

Oct 2017 4.409271 251.4075 -1.43927418

Nov 2017 3.492821 251.9094 1.85279980

Dec 2017 1.335672 252.4113 3.56807323

Jan 2018 -3.589169 253.0670 1.64715658

Feb 2018 -6.936089 253.7228 -0.59918102

Mar 2018 -7.056369 254.5410 -3.64711417

Apr 2018 -5.733749 255.3592 -2.70044696

May 2018 -1.934334 256.2473 -0.47547316

Jun 2018 2.627983 257.1354 -1.26340011

Jul 2018 4.644695 258.0399 0.06537710

Aug 2018 4.225757 258.9444 1.32980429

Sep 2018 4.513514 259.8826 -0.39611510

Oct 2018 4.409271 260.8208 -0.75503399

Nov 2018 3.492821 261.7933 1.62963893

Dec 2018 1.335672 262.7658 0.62276130

# Is the data stationary?

In the process of checking whether the data is stationary or not, a stationary model is applied to the data. If there is evidence of a seasonal component or a trend it is removed.

values.fit.ma <- ts(na.omit(values.fit.ma), frequency=12, start=c(2014, 1))

acf(values.fit.ma)

Chart, bar chart

Description automatically generated

adf.test(values.fit.ma)

Result:

Augmented Dickey-Fuller Test

data: values.fit.ma

Dickey-Fuller = 1.5393, Lag order = 4, p-value = 0.99

alternative hypothesis: stationary

The data will need to be transformed to make it more stationary, this conclusion was derived from the ACF plot showing a sharp drop in values and from the result of the ADF test with the p-value being 0.99 meaning we can’t reject the null-hypothesis of non-stationarity.

## Differencing

To make the data stationary various mathematical transforms must be used. We are using a lag of 12 because the data are monthly, and the period of the seasonality is 12 months.

values\_diff12 = diff(values.fit.ma, lag = 12)

tm <- cbind(values, values\_diff12)

plot(tm)

Chart, histogram

Description automatically generated

values\_diff12\_1 = diff(values\_diff12)

tm <- cbind(values, values\_diff12, values\_diff12\_1)

plot(tm)

Chart, histogram

Description automatically generated

acf(na.omit(values\_diff12\_1), main="ACF for series after 1st difference")

Chart, box and whisker chart

Description automatically generated

adf.test(na.omit(values\_diff12\_1))

Result:

Augmented Dickey-Fuller Test

data: na.omit(values\_diff12\_1)

Dickey-Fuller = -2.5724, Lag order = 4, p-value = 0.3421

alternative hypothesis: stationary

The data appear to be non-stationary because the p-value is above 0.05. There is also a seasonal component in the ACF.

diff.values <- na.omit(values\_diff12\_1)

par(mfrow=c(2,1))

acf(diff.values, main="")

pacf(diff.values, main="")

Timeline

Description automatically generated

# Choosing the Model

Model 1: Sarima

values.fit.sarima1 <- arima(values.fit.ma, order=c(1, 1, 0), seasonal = list(order = c(1, 1, 0), period = 12))

values.fit.sarima1

Result:

Call:

arima(x = values.fit.ma, order = c(1, 1, 0), seasonal = list(order = c(1, 1,

0), period = 12))

Coefficients:

ar1 sar1

0.4366 -0.1459

s.e. 0.1088 0.1442

sigma^2 estimated as 10.03: log likelihood = -190.53, aic = 387.05

Residuals

tsdisplay(residuals(values.fit.sarima1), lag.max=45, main='SARIMA Model 1 Residuals')

Graphical user interface

Description automatically generated with medium confidence

forecasting accuracy  
forecast.sarima1 <- forecast(values.fit.sarima1, h=12) #12months  
accuracy(forecast.sarima1)

Result:

ME RMSE MAE MPE MAPE MASE ACF1

Training set 0.4523532, 2.921444, 2.180838, 0.1446918, 0.8157319 ,0.1501135, 0.1516219

Model 2

values.fit.sarima2 <- arima(values.fit.ma, order=c(3, 1, 0), seasonal = list(order = c(1, 1, 0), period = 12))  
values.fit.sarima2

Result:

Call:

arima(x = values.fit.ma, order = c(3, 1, 0), seasonal = list(order = c(1, 1,

0), period = 12))

Coefficients:

ar1 ar2 ar3 sar1

0.8571 -0.7208 0.6050 -0.0898

s.e. 0.1018 0.1063 0.1036 0.1350

sigma^2 estimated as 5.875: log likelihood = -171.47, aic = 352.94

residuals  
tsdisplay(residuals(values.fit.sarima2), lag.max=45, main='SARIMA Model 2 Residuals')

Chart

Description automatically generated

forecasting accuracy

forecast.sarima2 <- forecast(values.fit.sarima2, h=12) #12months

accuracy(forecast.sarima2)

Result:

ME RMSE MAE MPE MAPE MASE ACF1

Training set 0.3061826, 2.237134 ,1.675537, 0.0977514, 0.6224838, 0.1153322, 0.1388304

Model 3

values.fit.sarima3 <- arima(values.fit.ma, order=c(6, 1, 0), seasonal = list(order = c(1, 1, 0), period = 12))  
values.fit.sarima3

Results:

Call:

arima(x = values.fit.ma, order = c(6, 1, 0), seasonal = list(order = c(1, 1,

0), period = 12))

Coefficients:

ar1 ar2 ar3 ar4 ar5 ar6 sar1

1.0738 -1.1323 1.0706 -0.5219 0.2987 0.0237 -0.1295

s.e. 0.1185 0.1696 0.2077 0.2127 0.1695 0.1305 0.1385

sigma^2 estimated as 4.968: log likelihood = -165.76, aic = 347.52

Residuals

tsdisplay(residuals(values.fit.sarima3), lag.max=45, main='SARIMA Model 3 Residuals')

Chart

Description automatically generated with medium confidence

forecasting accuracy

forecast.sarima3 <- forecast(values.fit.sarima3, h=12) #12months  
accuracy(forecast.sarima3)

Results:

ME RMSE MAE MPE MAPE MASE

Training set 0.2719145, 2.057356, 1.519558, 0.08627085, 0.5666311, 0.1045957 –

ACF1 0.004227259

As the results for the previous tests were bad, another method to use is Auto.arima, which can be used to estimate conclusions and view with the conclusions drawn so far.

Auto.arima

values.fit.sarima4 <- auto.arima(ma.fit, seasonal = TRUE)  
values.fit.sarima4

Results:

Series: ma.fit

ARIMA(2,2,0)(0,1,0)[12]

Coefficients:

ar1 ar2

-0.0202 -0.7469

s.e. 0.0774 0.0753

sigma^2 = 6.406: log likelihood = -174.01

AIC=354.02 AICc=354.37 BIC=360.89

Residuals

tsdisplay(residuals(values.fit.sarima4), lag.max=45, main='SARIMA Model 4 Residuals'

Chart, box and whisker chart

Description automatically generatedThe Auto.arima shows 2 non-seasonal terms and 0 seasonal for AR.

## Holt Winters

This model shows exponential smoothing on the data.

Model 1

values.fit.hw1 <- ets(values.fit.ma, model="MAM", damped=FALSE)

values.fit.hw1

Results:

ETS(M,A,M)

Call:

ets(y = values.fit.ma, model = "MAM", damped = FALSE)

Smoothing parameters:

alpha = 0.2845

beta = 0.0372

gamma = 2e-04

Initial states:

l = 230.3517

b = -0.2942

s = 0.9805 1.0102 1.0097 1.0187 1.0225 1.0223

1.0172 1.0108 0.9909 0.9845 0.971 0.9616

sigma: 0.0184

AIC AICc BIC

675.4992 684.3688 717.4196

Plot(values.fit.hw1)

Chart, line chart

Description automatically generated

accuracy(values.fit.hw1)

Results:

ME RMSE MAE MPE MAPE MASE ACF1

Training set 1.785745, 4.822824, 3.169552, 0.5904164, 1.152957, 0.2181696, 0.6855246

Model 2

values.fit.hw2 <- ets(values.fit.ma, model="MAM",  
damped=TRUE)  
values.fit.hw2

Results:

ETS(M,Ad,M)

Call:

ets(y = values.fit.ma, model = "MAM", damped = TRUE)

Smoothing parameters:

alpha = 0.9999

beta = 0.1319

gamma = 1e-04

phi = 0.9534

Initial states:

l = 228.9442

b = 0.1643

s = 0.9791 0.9988 1.0091 1.0141 1.0182 1.0196

1.0232 1.0133 0.9956 0.9834 0.9757 0.9698

sigma: 0.0116

AIC AICc BIC

595.4691 605.5279 639.8554

plot(values.fit.hw2)

Chart

Description automatically generated

accuracy(values.fit.hw2)

Results:

ME RMSE MAE MPE MAPE MASE ACF1

Training set 0.7388153, 2.835673, 2.139511, 0.2428983, 0.8034506, 0.1472689, 0.3975071

# Model evaluation and forecasting Checking Accuracy

values.holdout <- window(values, start=2020,end=2021)

values2021 <- ts(values.holdout, start = c(2020, 1), frequency=12)

plot.ts(values2021)

Chart, line chart

Description automatically generated

## Forecasting using the chosen model(s)

### Seasonal ARIMA

forecast.values.sarima4 <- forecast(values.fit.sarima4, h=12)

plot(forecast.values.sarima4)

lines(values2021, col="cyan", lwd=2, lty=3)

Chart, line chart

Description automatically generated

accuracy(forecast.values.sarima4, values2021)

Results:

ME RMSE MAE MPE MAPE MASE

Training set 0.03603153, 1.139661, 0.8698042, 0.01081753, 0.3539355, 0.1298755 ACF1 0.03935313

Test set 11.46374127 11.463741 11.4637413 4.16863319 4.1686332 1.7117169 NA

Box.test(residuals(values.fit.sarima4), type="Ljung-Box")

Results:

Box-Ljung test

data: residuals(values.fit.sarima4)

X-squared = 0.096098, df = 1, p-value = 0.7566

par(mfrow=c(2,2))  
acf(na.omit(values.fit.sarima4$residuals))  
pacf(na.omit(values.fit.sarima4$residuals))  
plot(na.omit(values.fit.sarima4$residuals))  
qqnorm(na.omit(values.fit.sarima4$residuals))  
qqline(na.omit(values.fit.sarima4$residuals), col="cyan")

Graphical user interface, chart

Description automatically generated

### Holt Winters seasonal smoothing

plot(forecast(values.fit.hw2))

lines(values2021, col="cyan", lwd=2, lty=3)

Chart, histogram

Description automatically generated

accuracy(forecast(values.fit.hw2), values2021)

Results:

ME RMSE MAE MPE MAPE MASE ACF1

Training set 0.3363142 1.876372 1.57032 0.1335901 0.6433088 0.2344735 0.1622858 Theil's U NA

Test set 25.5842728 28.685498 25.58427 8.3294081 8.3294081 3.8201343 0.6093803 2.704669

Box.test(residuals(values.fit.hw2), type="Ljung-Box")

Results:

Box-Ljung test

data: residuals(values.fit.hw2)

X-squared = 1.7218, df = 1, p-value = 0.1895

par(mfrow=c(2,2))  
acf(values.fit.hw2$residuals)  
pacf(values.fit.hw2$residuals)  
plot(values.fit.hw2$residuals)  
qqnorm(values.fit.hw2$residuals)  
qqline(values.fit.hw2$residuals, col="cyan")

Graphical user interface, chart, diagram

Description automatically generated with medium confidence

forecast.values.sarima4

Results:

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

Feb 2019 252.9291 249.2271 256.6310 247.2674 258.5907

Mar 2019 252.2582 248.2761 256.2403 246.1681 258.3483

Apr 2019 254.6536 250.6714 258.6358 248.5633 260.7439

May 2019 260.4908 256.5085 264.4730 254.4005 266.5811

Jun 2019 265.8459 261.8636 269.8281 259.7556 271.9362

Jul 2019 269.2804 265.2982 273.2627 263.1901 275.3707

Aug 2019 269.9264 265.9442 273.9087 263.8361 276.0167

Sep 2019 269.6535 265.6712 273.6357 263.5631 275.7438

Oct 2019 270.0664 266.0842 274.0486 263.9761 276.1567

Nov 2019 272.0145 268.0323 275.9967 265.9242 278.1048

Dec 2019 270.4256 266.4434 274.4078 264.3353 276.5159

Jan 2020 263.5363 259.4126 267.6599 257.2297 269.8428

# Conclusion

I fitted 4 different SARIMA models and 2 exponential Holt Winters models to fit to the data. I tested their accuracy and chose both the 4th SARIMA and ETS model for the forecast. That forecast matches the holdout plot because the prices continued to rise sharply in 2021.

The graphs from plotting the data show a sharp rise in median prices from 2019 onwards, no doubt the spread of COVID had a part to play in this though prices were steadily increasing from 2014. The prices also seemed to spike in Spring, suggesting the prices were increased at that time of year because the real estate market is traditionally active in Spring. On average the prices were 5% higher in the spring so that’s a good time to sell. The sharp rise in prices in 2020 and 2021 is mirrored in the rest of the Canadian market – see [here](https://apciq.ca/en/montreals-residential-real-estate-market-compared-to-other-major-canadian-markets/).

It would be a good idea to check the exact correlation with the general Canadian real estate market and also to check if there is any correlation with mortgage rates.