A “Roof” on Prices: Forecasting the Prices of Houses in San Diego

Tanisha Dighe

Department of Mathematics

University of California, San Diego

Section 1: Introduction

Having grown up watching shows like “House Hunters”, the real estate industry has always intrigued me. However, today, real estate plays a less joyful role in my life – as thousands of students were forced off campus, I found myself, along with many others, in a housing crisis created by the Covid-19 pandemic. Prices surged as the number of options dwindled – trying to block an apartment on a website became the equivalent of trying to buy tickets to a fast-selling concert. These events made me curious about market trends in San Diego. Thus, the aim of this project is to examine the median prices of houses in San Diego and predict their future. I will also take into consideration factors other than the historical data of the median prices such as the percentage change in sales of existing homes, and the median time on market of these homes to see if they can improve the prediction of median prices. I think this would be an interesting project to pursue because I’m curious to know how the incorporation of these external factors affects the accuracy of predictions. Furthermore, even though I have seen other papers which examine housing markets, I have yet to come across one that forecasts the prices of homes.

Section 2: Background

The value of a property is based on what willing buyers in the market will pay for the property – but every buyer is different. Families may weigh the proximity of schools and parks more than elderly couple looking for a quiet neighbourhood. One of the most important considerations when buying a home is the location – appraisers look at three primary indicators when determining the value of the house’s location: the quality of local schools, employment opportunities and the proximity to shopping, entertainment, and recreational centres (Gomez). Another important factor that goes into determining the value of a home is its size and usable space. Garages, attics, and unfinished basements are generally not counted in usable square footage, so if you have 2000 square foot home with 600square foot garage – that would only be considered 1400 square feet of livable space (Gomez).
However, even if the home is in excellent condition and in the best location, the number of other properties for sale in the area and the number of buyers in the market can impact the value of a home – more buyers than properties drives up the prices and less buyers than properties drives down the prices allowing for more negotiations.
Finally, the broader economy also impacts the person’s ability to buy or sell a home. For example, if employment is slow, then fewer people might be able to buy a home (Gomez).

Section 3: Model Description

In this project, I have used time series analysis to predict the median prices of homes. Time series analysis uses data points recorded at consistent intervals over a set period of time to show how variables change over time. Time series analysis was appropriate for this project because I was trying to understand how my data (the median listing price) fluctuates over

time. Specifically, I used the Seasonal Autoregressive Integrated Moving Average (SARIMA) method of time series forecasting because unlike it's simpler predecessor, the ARIMA model that is popularly used for univariate time series data forecasting. The SARIMA model accounts for seasonal variation – a characteristic that is expected in the housing market because home buyers overwhelmingly tend to move in early spring and the summer months (Kushi).
After which, in order to incorporate the exogenous variables, I will be using the Seasonal Auto-Regressive Integrated Moving Average with eXogenous factors (SARIMAX) model. The SARIMAX model is like the SARIMA model except it also incorporates external factors.
In order to understand the SARIMA and SARIMAX models, it is important to explain the terms they are characterized by. The ARIMA model can give a better introduction – it characterized by three terms: p, d, q – where p is the order of the Auto Regressive (AR) term, q is the order of the Moving Average (MA term), and d is the number of differencing required to make the time series stationary which means the data needs to not have trends or seasonality (Prabhakaran). When the time series has seasonal patterns, like mine did, then you need to incorporate the corresponding P, D, Q of the seasonal component and you obtain a SARIMA model.

Section 3.1: Differencing

Taking a difference of the time-series is done by subtracting the previous value from each value, which tends to make the data more stationary (Datascience George). There is a value called "d" which represents how many times the data is to be differenced.

Section 3.2: AR model

In an AR model the model predicts the next data point by looking at previous data points and using a mathematical formula similar to linear regression (Datascience George). The order p determines how many previous data points will be used.
Mathematically, it is shown as follows:
Y_t is a function of its own lags

$$y_t = \beta + \varepsilon_t + \sum_{i=1}^{p} \theta_i y_{t-i}$$

Where p is the number of time lags to regress on, ε_t is the noise at time t and β is a constant. (GitHub Pages).

Section 3.3: MA model

An MA model performs calculations based on noise in the data along with the data's slope (Datascience George).
Thus, mathematically, it is shown as follows:

$$y_t = \mu + \epsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

Where μ represents the mean of y, θ represents the parameters of the model and ε_t is noise at time t.

Thus, ARIMA is given by

$$\Delta^d y_t \;=\; \Delta^d(\,\beta + \sum_{i\,=1}^{p} \theta_i y_{t-i}) + \Delta^d(\mu +\; \epsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}) + \Delta^d \epsilon_t$$

Where Δ^d is an integration operator defined as follows:

$$y_t^{[d]} \;=\; \Delta^d y_t \;=\; y_t^{[d-1]} - y_{t-1}^{[d-1]}$$

and *d* is the order of differencing used (GitHub Pages).
Or, in simpler terms, Predicted y_t = Constant + Linear combination Lags of Y (up to p lags) + Linear Combination of Lagged forecast errors (up to q lags) (Prabhakaran).

Section 3.4: SARIMA model

Takes seasonality into account by applying an ARIMA model to lags that are integer multiples of seasonality (GitHub Pages). Once the seasonality is modelled, an ARIMA model is applied to the leftover to capture the non-seasonal structure. Thus, if I have a time series y with seasonality m (number of time lags comprising one full period of seasonality, i.e. 12 for monthly data having 1 season a year) and differencing operator Δ_s^D that takes the seasonal differences of the time series, the general SARIMA(*p*,*d*,*q*)(*P*,*D*,*Q*,*m*) model is given by (Verma):

$$\Phi_p(L)\phi_P(L^s)\Delta^d\Delta_s^D y_t = \;\Theta_q(L)\theta_Q(L^s)\epsilon_t$$

Where
$\Phi_p(L)$ = Non-seasonal autoregressive lag polynomial
$\phi_P(L^s)$ = Seasonal autoregressive lag polynomial
$\Delta^d\Delta_s^D y_t$= times series, differenced d times and seasonally differenced D times
$\Theta_q(L)$ = non-seasonal moving average lag polynomial
$\theta_Q(L^s)$ = seasonal moving average lag polynomial
ϵ_t = the noise at time t

Section 3.5: SARIMAX Model

Mathematically given by:

$$\Phi_p(L)\phi_P(L^s)\Delta^d\Delta_s^D y_t = \;\mathrm{A(t)} + \Theta_q(L)\theta_Q(L^s)\epsilon_t$$

Where everything is the same as the SARIMA model except A(t) = the trend polynomial (including the intercept) (Verma).

Section 4: Data and Data Processing

My data was obtained from the California association of Realtors' website (California Association of Realtors). This website has the historical data from January 1990 to December 2021 for "Sales of Existing Single Family Homes (percent changes only)", "Median Prices of Existing Single Family Homes" and "Median Time on Market of Existing Single Family Homes" for all of the counties across California, in excel sheets. However, for the purpose of this project, I only required the data specific to San Diego. So I extracted the respective columns from those tables to create a new table that only had San Diego data.

Excerpt of Data:

	A	B	C	D
1	MonYr	MPE	MTM	PCS
2	01/01/90	180484	57	
3	02/01/90	180714	61.8	
4	03/01/90	183701	59.9	
5	04/01/90	181567	58.3	
6	05/01/90	180794	58.3	
7	06/01/90	186733	65.6	
8	07/01/90	185861	67.2	
9	08/01/90	185639	70.6	
10	09/01/90	186272	72.8	
11	10/01/90	179444	66	
12	11/01/90	176980	71.5	
13	12/01/90	181470	80.1	
14	01/01/91	182000	80.9	-0.266
15	02/01/91	178888	88.9	-0.342
16	03/01/91	183111	81.4	-0.344
17	04/01/91	183114	73.7	-0.077
18	05/01/91	188732	74.7	0.022
19	06/01/91	188509	74	0.117

Figure 1: Excerpt of Data

Where MPE, MTM, and PCS are “Median Prices of Existing Single Family Homes”, “Median Time on Market of Existing Single Family Homes” and “Sales of Existing Single Family Homes (percent changes only)” respectively.
Clearly, the data from January 1990 to December 1990 is missing for PCS so that year was not taken for my model nor were the MPE and MTM either in order to maintain a fair contribution of all factors.
On plotting the various MPE values against time, I get the following graph:

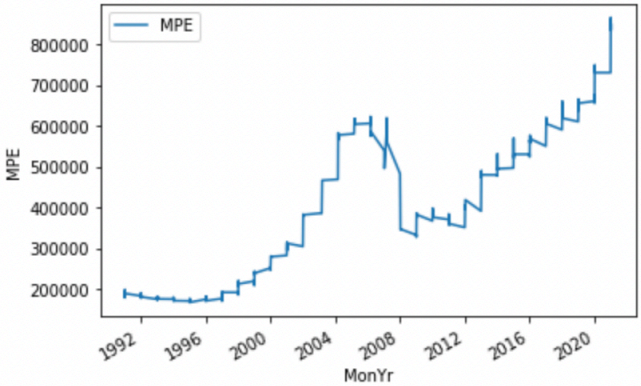

Figure 2: Plotting median prices of homes against time

This data (along with my MTM and PCS values) was further split into an 80:20 training and test set so that I could cross-validate model. Since this was a time series, I could not sample any random values as is usually done for classification algorithms. Instead, I took the first 80% of my data as my training set and kept the later 20% as my test set.
Finally, I have conducted this entire project via python using various packages like pandas, numpy and statsmodels to do my analysis.

Section 5: Computing The Model

SARIMA

Section 5.1: Finding *p*, *d*, *q*

Order of differencing (d):
The right order of differencing is the minimum differencing required to get a near-stationary series which roams around a different mean and the ACF plot of the series reaches to 0 fairly quick. The ACF plot gives us values of auto-correlation of any series with its lagged values. You only need differencing if the series is non-stationary (Prabhakaran).

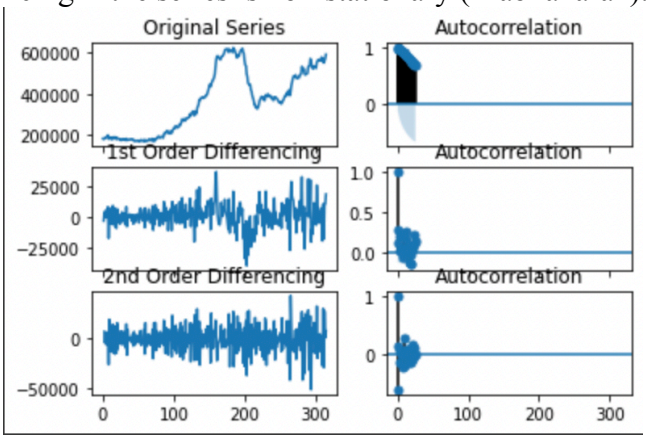

Figure 3: Differencing MPE and respective ACF plots

I see that the series becomes roughly stationary after 2 orders of differencing. So I take $d = 2$.

Order of the AR term (p):
I initially take the order of the AR term to be equal to as many lags that crosses the significance limit in the PACF plot (Prabhakaran). The PACF plot finds correlation of the residuals (which remains after removing the effects which are already explained by the earlier lag(s)) with the next lag value hence 'partial' and not 'complete' as I remove already found variations before I find the next correlation.

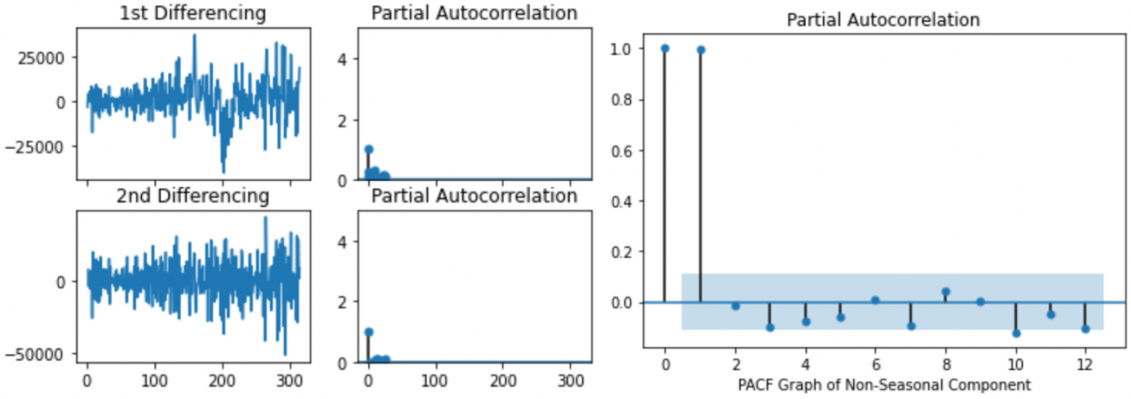

Figure 4: PACF plot of AR term

2 lags appears to be significantly above the confidence interval thus I can fix *p* as 2.

Order of the MA term (q):
The ACF plot of the time series also gives number of MA terms.

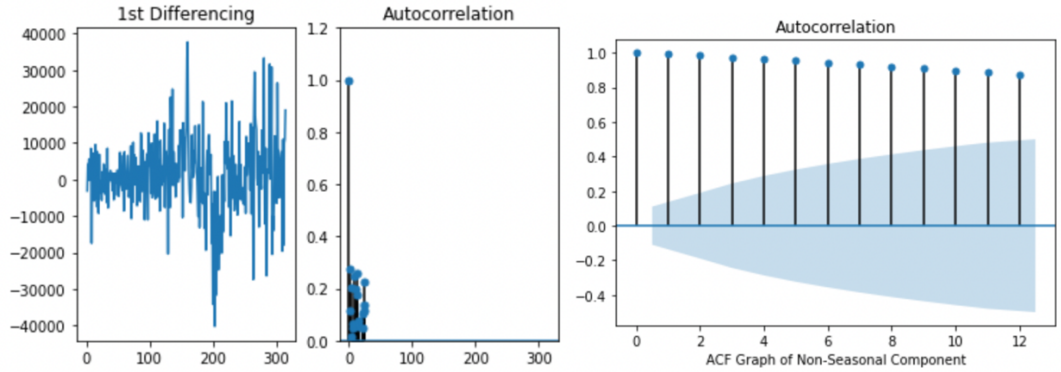

Figure 5: ACF plot of MA term

I see all 12 lags well above the significance level – so I can go with q = 12.

Section 5.2: Finding *P*, *D*, *Q*, *m*

In order to find the seasonal order, I must extract the seasonal component from the time series. On using the seasonal_decompose() function from the statsmodels library. This returns the seasonal components and we obtain the following graph:

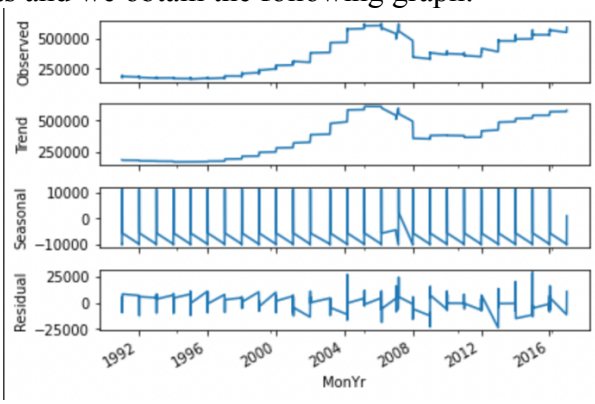

Figure 6: Seasonal decomposition of time series

Once we have the seasonal components, we can implement the same procedure as we did to the non-seasonal components in finding *P*, *D*, *Q* and *m*

m:
We know that m is simply the periodicity, or the number of periods in a season – this is 12 for my data since I am looking at monthly data (LoDuca).

Order of differencing (D):
First we check if the seasonal component even needs differencing or not by seeing if it is stationery. For this we implement the Adfuller test which is a statistical significance test which checks if a unit root is present in the time series. A unit root is said to exist in a time series if the value of alpha = 1 in the below equation (Prabhakaran, Augmented Dickey Fuller Test (ADF Test) – Must Read Guide):

$$Y_t = \alpha Y_{t-1} + \beta X_e + \varepsilon$$

Where Y_t is the value of the time series at time 't' and X_e is an exogenous variable.

If a unit root is present then it is not stationary. The result obtained was that indeed, the seasonal component was stationary so no differencing was required. Hence, $D = 0$.

Order of the AR term (P):

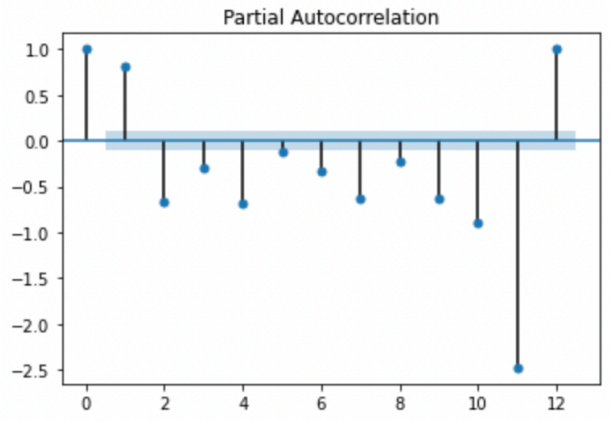

Figure 7: PACF plot seasonal AR term

Here, many lags appear to be over the confidence interval, so we take the largest ones out of that: $P = 4$

Order of the MA term (Q):

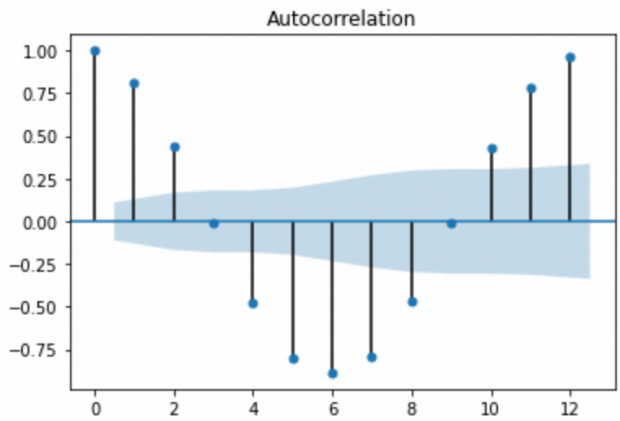

Figure 8: ACF plot seasonal MA term

Similar to finding the P, we take $Q = 7$.

Thus, we finally have our SARIMA model of SARIMA(2,2,12)(4,0,7,12). However, running this model was a challenge. The greater P and Q values, the longer it took to execute. In fact, at P = 4 and Q = 7, the model failed to execute. Since I did not have a high enough computing power, I was forced to reduce the P and Q values and obtain the best fit from that. This was the model SARIMA(2,2,12)(1,0,2,12). This model could now be implemented to get the forecast.

The model was forecasted with steps equal to the length of test set and the following prediction was obtained:

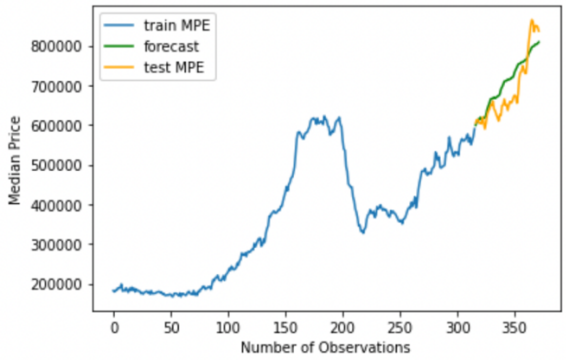

Figure 9: SARIMA forecast and test set

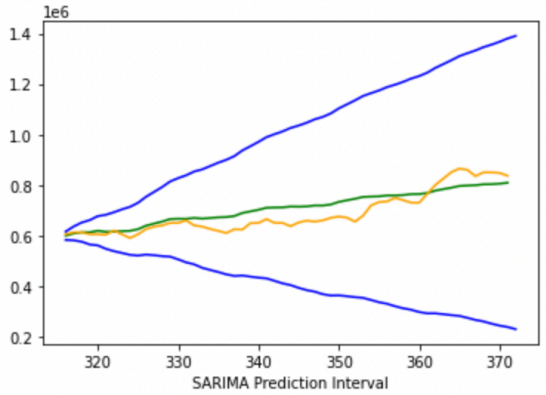

Figure 10: SARIMA Prediction Interval

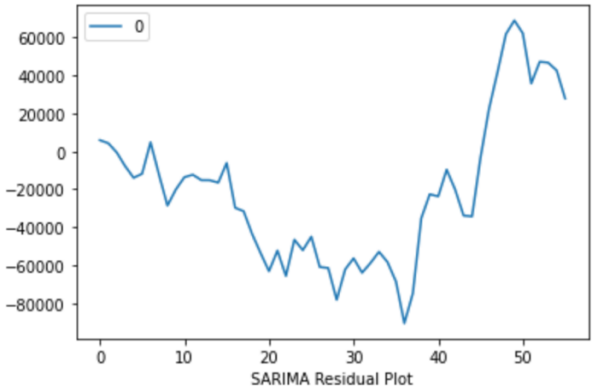

Figure 11: SARIMA residual plot

Clearly the forecast lies well within the 90% prediction interval which further supports the goodness of this model and although residual plot fluctuates quite a bit, the values appear to average out. The RMSE value of the SARIMA model on the test set was 0.1599.

SARIMAX

Next, we computed the SARIMAX model to evaluate how the introduction of exogenous variables would affect the prediction. In order to do this, we implemented a simple grid search algorithm that seeked to minimise the AICs. The Akaike Information Criterion or AIC is a mathematical method for evaluating how well a model fits the data it was generated from. It determines the relative information value of the model using the maximum likelihood estimate and the number of parameters in the model. The formula is given by (Bevans):

$$AIC = 2K - 2\ln(L)$$

Where K is the number of independent variables used and L is the log-likelihood estimate. This grid search was implemented in the interest of time and efficiency.
All of the variables were differenced and standardised and then ran against the auto.arima() function which found the best combination of p, d, q, P, D and Q with the exogenous factors MTM and PCI.

Thus we obtained the model SARIMAX(0,2,1)(2,0,1,12) and forecasted it with steps equal to the length of test set to get the graph:

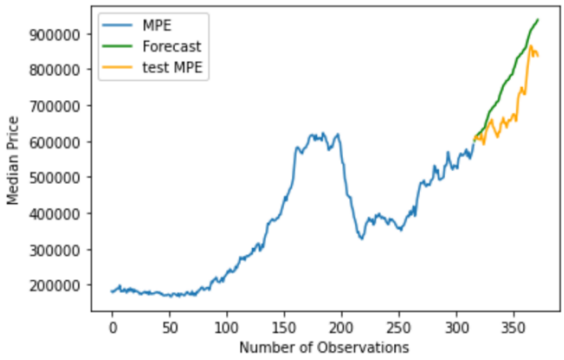

Figure 12: SARIMAX forecast and test set

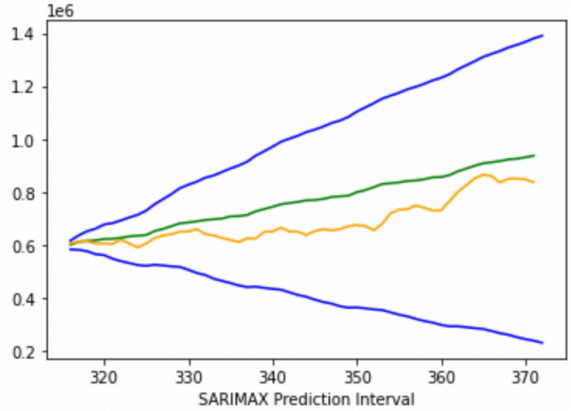

Figure 13: SARIMAX Prediction Interval

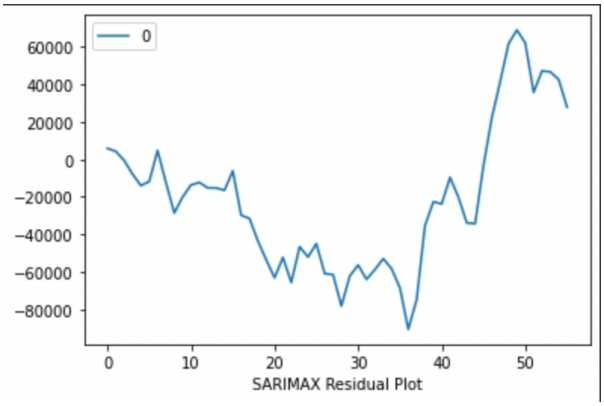

Figure 14: SARIMAX residual plot

The prediction interval against the number of observations shows that even this forecast is likely to fall in range with the actual values 90% of the time. The residual plot of this model also despite fluctuating quite a bit appears to still be relatively close to the actual values. The RMSE value of the SARIMAX model on the test set was 0.3169

Section 6: SARIMA and SARIMAX Comparison

On comparing the two RMSE values of the models, it was clear that SARIMA was predicting closer to the actual test values in the set. This presented a problem because the addition of exogenous factors was supposed to provide a better fit.
Consequently, we needed to determine if the SARIMAX was overfitting the data. This was done running the two models on the training set and comparing the RMSE values – if the RMSE of SARIMAX was lower than SARIMA's, then the model could be considered overfit (Dalpiaz). Which is what we obtain:

Train Set RMSE of SARIMA	Train Set RMSE of SARIMAX
0.034655	0.031277

Clearly, the train set RMSE of SARIMAX is lesser than that of SARIMA by about ~9.7% so we can consider the SARIMAX to be an overfit and the SARIMA model a better fit.

Consequently, I decided to use the SARIMA model to forecast the MPE till 2025 and obtained the following graphs:

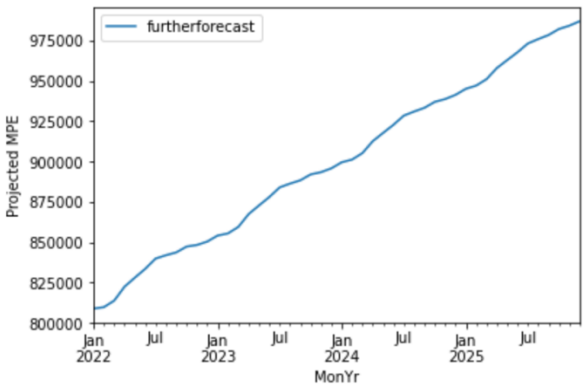

Figure 15: SARIMA further forecast

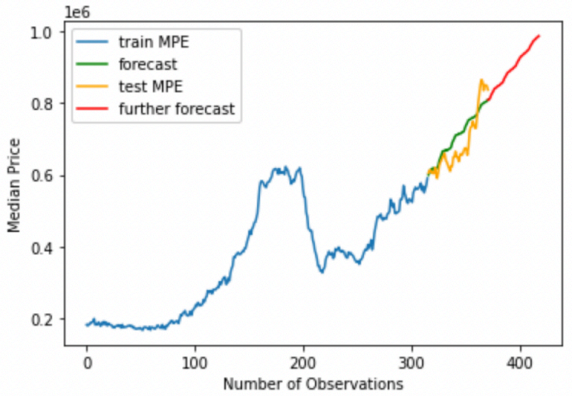

Figure 16: Forecast against test and train data

Thus, from our model, the median prices of homes is expected to reach almost a million dollars.

Section 7: Evaluation and Conclusion

Through this time series analysis project, our goal was to find the projected median prices of homes in San Diego using SARIMA and SARIMAX models. Both models provide valuable insight into the dataset and what it means for the future of housing prices in San Diego. While the SARIMA model performed better on the test set, the SARIMAX model was able to better account for the steep climb in housing prices from 2018 onwards since the SARIMA model chose average out the extremities. Furthermore, the SARIMA was able to better account for the seasonality of the data while the SARIMAX model produced a “smoother” line.

The main problem with SARIMAX was that it was overfitting. Usually this is addressed by constructing a validation set to test the hyperparameters against. However, our hyperparameters P, D, Q and m were obtained through a rigorous grid search that was anyway testing out all possible combinations to obtain the best fit parameters. The best possible way to reduce overfitting would have been to obtain more training data (Carremans). Time series analysis requires a lot of historical data. I had 19 years of data and although this is a decent amount, having more values would have improved the performance of both of my models. However, this was not possible as this was the only data that was available at the time and I did not want to reduce my test set any further as I wouldn’t be able to assess the quality of model without large enough data points. Another popular method to reduce overfitting is removing layers or extraneous variables. But this would just mean going back to my SARIMA model since I was anyway using only 2 extra variables (Carremans). A third way to address the overfitting would be to apply weight regularization to the model. This would add a cost to the loss function of the network for large weights (or parameter values). As a result, I would have simpler model which would focus only on the relevant patterns of the data (Carremans). However, in the interest of time, this did not seem feasible for the purpose of this project.

Another challenge was lack of computing power. As mentioned when computing the SARIMA model, the function would either take over a minute or simply fail to execute for high values of P and Q. Although this might pass for this project, it is imperative to have high functioning models in the real world. Thus, I was forced to compromise the fit of the model for a faster execute time which must have led to a higher RMSE.

Finally, neither of the models can account for "acts of God" or sudden changes in the economy. This is well depicted in the MPE graph where the median prices take a sharp dip in 2008 because of the financial crisis. Similarly, in San Diego, the Covid-19 Pandemic caused a massive housing crisis which caused the prices of houses to soar – this period just happens to be part of my test set. There was no way the SARIMA or SARIMAX models could have predicted that or have been able to account for that which also must have negatively contributed to the forecast of the models. Thus, from the above discussion it is clear to see that sometimes simplicity is key and that adding extra factors like I tried to do with MTM and PCS ended up negatively impacting my model rather than improving the fit.

Bibliography

Gomez, Joe. *8 critical factors that influence a home's value*. 27 March 2019. 11 February 2022. <https://www.opendoor.com/w/blog/factors-that-influence-home-value>.

Brownlee, Jason. *Gentle introduction to the Stacked LSTM with example code in Python.* 18 August 2017. 13 February 2022. <https://machinelearningmastery.com/stacked-long-short-term-memory-networks/>.

California Association of Realtors. *Historical Housing Data*. 18 January 2022. 13 February 2022. <https://www.car.org/en/marketdata/data/housingdata>.

Prabhakaran, Selva. *ARIMA Model – Complete Guide to Time Series Forecasting in Python*. 22 August 2021. 13 February 2022. <https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/>.

GitHub Pages. *From AR to SARIMAX: Mathematical Definitions of Time Series Models*. n.d. 13 February 2022. <https://phosgene89.github.io/sarima.html>.

LoDuca, Angelica. *Understanding the Seasonal Order of the SARIMA Model*. 12 July 2021. 13 February 2022. <https://towardsdatascience.com/understanding-the-seasonal-order-of-the-sarima-model-ebef613e40fa>.

Carremans, Bert. *Handling overfitting in deep learning model*. 23 August 2018. 3 March 2022. <https://towardsdatascience.com/handling-overfitting-in-deep-learning-models-c760ee047c6e>.

Kushi, Odeta. *Will Housing Market Seasonality Return to Normal?* 17 September 2021. 3 March 2022. <https://blog.firstam.com/economics/will-housing-market-seasonality-return-to-normal>.

Datascience George. *A Brief Introduction to ARIMA and SARIMAX Modeling in Python*. 9 April 2020. 3 March 2022. <https://medium.com/swlh/a-brief-introduction-to-arima-and-sarima-modeling-in-python-87a58d375def>.

Verma, Yugesh. *Complete Guide To SARIMAX in Python for Time Series Modeling*. 30 July 2021. 3 March 2022. <https://analyticsindiamag.com/complete-guide-to-sarimax-in-python-for-time-series-modeling/>.

Prabhakaran, Selva. *Augmented Dickey Fuller Test (ADF Test) – Must Read Guide*. 2 November 2019. 3 March 2022. <https://www.machinelearningplus.com/time-series/augmented-dickey-fuller-test/>.

Bevans, Rebecca. *Akaike Information Criterion | When & How to Use It*. 26 March 2020. 3 March 2022. <https://www.scribbr.com/statistics/akaike-information-criterion/#:~:text=The%20Akaike%20information%20criterion%20(AIC,best%20fit%20for%20the%20data.>.

Dalpiaz, David. *Chapter 4 Regression for Statistical Learning*. 28 October 2020. 3 March 2022. <https://daviddalpiaz.github.io/r4sl/regression-for-statistical-learning.html>.