

Predicting Housing Market – Milestone Report

Table of Contents

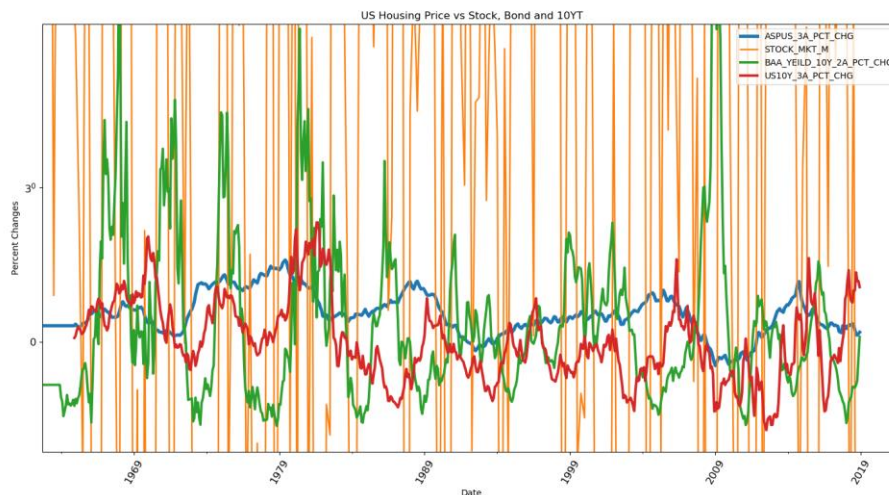
Predicting Housing Market – Milestone Report	1
Create Data sets and Data Exploration (Part 1)	2
Compare US average housing price with stock market, bonds and interest rates.....	3
Compare US average housing price with Consumer expenditures, Unemployment market and US GDP Growth	4
Compare US average housing price with Housing Supply Ratio, New 1F house sold in US and US Construction Permit	4
Machine Learning (Part 2).....	5
Linear Regression	5
K fold cross validation	5
Fitting Linear Regression using statsmodels¶	5
Correlation Matrix with Heatmap.....	6
Monthly housing DATA ERROR TESTING	6
Visualize the seasonality of multiple time series	6
Visualizing predicted values¶	8
Variance of the PCA features	8
Density plots¶.....	Error! Bookmark not defined.
Cross-validation with shuffling¶.....	9
Time-based cross-validation¶	9
Time Series (Part 3).....	10
Time series decomposition	10
FORECASTING ARIMA.....	10
Forecasting Housing Market using FBProphet (Part 4).....	10
Accurate and fast.	Error! Bookmark not defined.
Tunable forecasts.....	Error! Bookmark not defined.
Forecast using the ‘predict’ command:	11
Adjusting trend flexibility.....	Error! Bookmark not defined.

Visualizing Prophet models.....	11
Forecasting US housing market using Deep Learning (Part 5).....	12
Specifying a model	13
Compiling the model.....	13
Fitting the model.....	13
Classification models.....	13
Making predictions¶.....	13
Housing price Prediction_Using_CNN.....	13
Modeling	13
Define Network.....	13
Compile Network	13
Train Network¶.....	13
Evaluate Network.....	13
LSTM-RNN to forecast time-series.....	13
Visualising the results	13

Create Data sets and Data Exploration (Part 1)

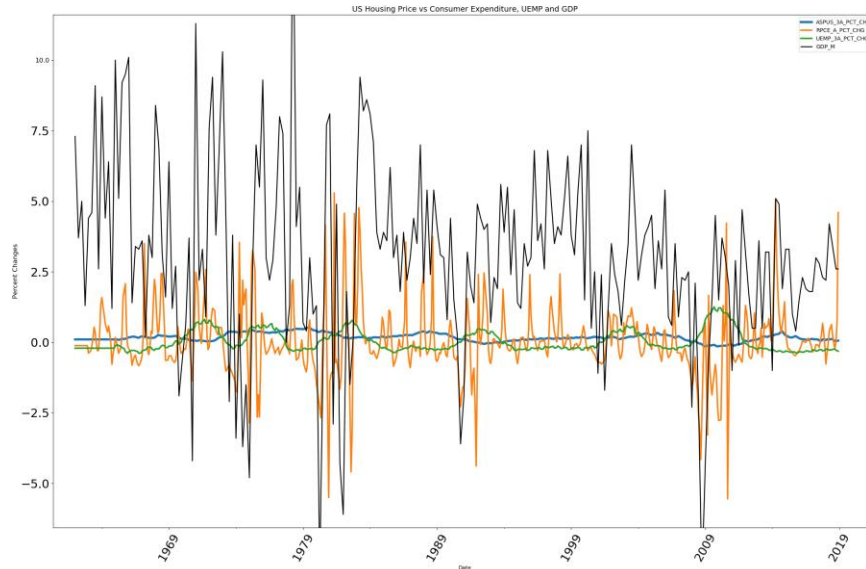
- During Housing market crash, average housing price drops 10% or more in 12 months
- This graph clearly indicate that during housing market crash (1981, 1992 & 2007) Personal Consumer Expenditures drops rapidly
- We can see positive growth in bond market during housing market crash (1981 & 2007)
- We can see that strong unemployment growth 1981 & 2007 or US Unemployment Growth is over 10% during housing market crash
- We can see that GDP growth is negative during 1981 & 2007 or GDP Growth is below - 5% during housing market crash
- Higher interest rate is one of the major factor for housing market crash
- Normally stock market goes down sharply along with housing market
- We can see that housing supply ratio jump quickly right before the housing market crash. In another words, rapid housing supply growth eventually brings the housing market down.
- During the Housing market (1981, 1992 & 2007), Number of new 1F house sold in US sharply goes down.
- During the Housing market crash (1981, 1992 & 2007), Number of US Construction PERMIT in US sharply goes down.

Compare US average housing price with stock market, bonds and interest rates



Before the housing market crash, generally interest rates are higher (RED) and bond markets are lower (GREEN). During Market Crash (1981, 1992 & 2007), stock market (ORANGE) goes down, bonds goes up and interest rates goes down. Recent Data (12/2018) shows strong housing market (BLUE), because strong stock market, low interest rate and healthy bond market.

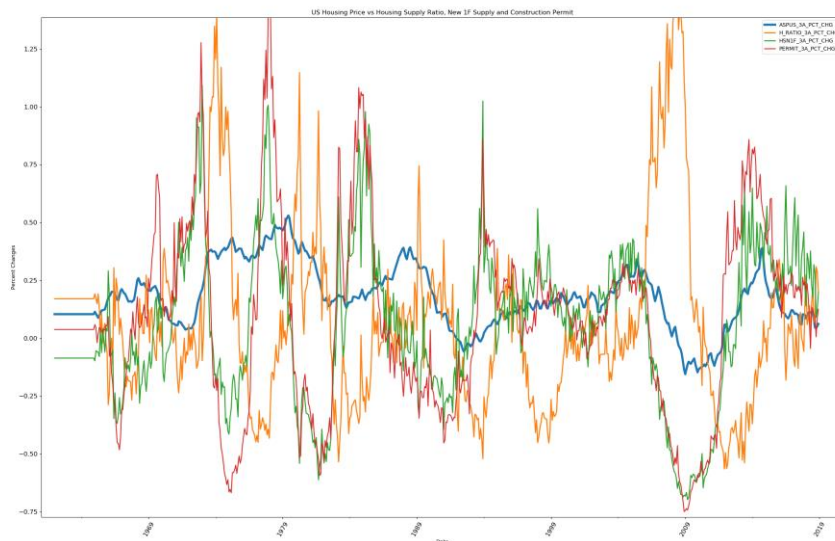
Compare US average housing price with Consumer expenditures, Unemployment market and US GDP Growth



During Market Crash (1981, 1992 & 2007), stock US GDP Growth drop sharply, US Unemployment Rate jump quickly and Personal Consumer Expenditures goes down.

If we look at the recent data, GDP (Black) is positive, Unemployment (Green) rate is very low, Personal Consumer expenditures (Orange) is positive. Based on these data housing market (Blue) still remain positive.

Compare US average housing price with Housing Supply Ratio, New 1F house sold in US and US Construction Permit



During Market Crash (1981, 1992 & 2007), Housing Supply Ratio (Orange) jump sharply, Both number of new 1F houses sold (Green) & number of construction permits (Red) drop rapidly. As Supply Ratio goes up, Housing Market goes down. On the other hand, if new 1F houses sold (Green) & number of construction permits (Red) goes up when market goes up and goes down when housing market goes down.

If we closely look at the graph for 2018 data, which shows Housing Supply Ratio is increasing slowly. Both number of new 1F houses sold & number of construction permits dropping slowly, which indicates downtrend market.

Machine Learning (Part 2)

Importing Housing Data

```
housing_df=pd.read_csv('C:/scripts/capstone2/housing_df2.csv', index_col=0)
```

ASPUS_3A_PCT_CHG US Average housing price movement (3 years % change)

H_RATIO_3A_PCT_CHG Housing Ratio: Number of house available vs. sold. (3 years % change)

HSN1F_3A_PCT_CHG New 1F House sold (3 years % change)

PERMIT_3A_PCT_CHG Current construction permits across USA (3 years % change)

STOCK_MKT_3A_PCT_CHG Stock market movement (3 years % change)

BAA_YEILD_10Y_2A_PCT_CHG Bond Market movement (2 years % change)

US10Y_3A_PCT_CHG US 10 Years Treasury Rate (3 years % change)

RPCE_A_PCT_CHG Personal consumer expenditures (1 years % change)

UEMP_3A_PCT_CHG Long Term Unemployment Rate (3 years % change)

RGDP_M_PCT_CHG Current US GDP (3 years % change)

Linear Regression

The R^2 in scikit learn is the coefficient of determination. It is $1 - \text{residual sum of square} / \text{total sum of squares}$.

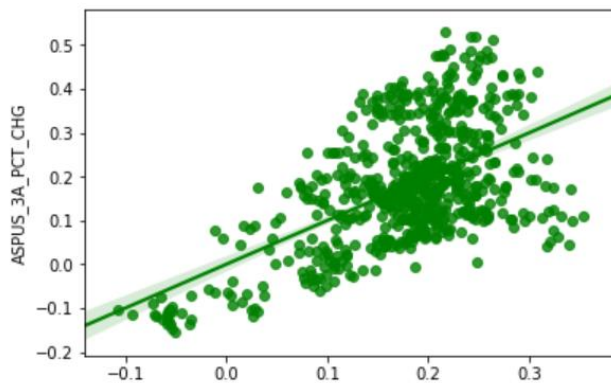
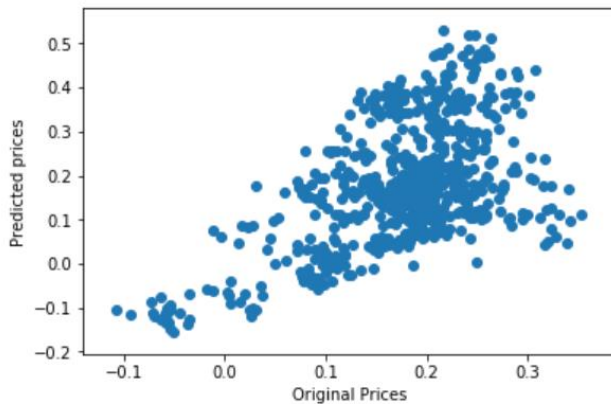
RMSE of the test data is closer to the training RMSE (and lower) if you have a well trained model. It will be higher if we have an overfitted model.

K fold cross validation

with Linear regressor we are able to predict the model with .114 RMSE and r squared 0.33 and cross validation root mean squared error is : 0.1507

Fitting Linear Regression using statsmodels

Statsmodels is a great Python library for a lot of basic and inferential statistics. It also provides basic regression functions using an R-like syntax, so it's commonly used by statisticians. While we don't cover statsmodels officially in the Data Science Intensive workshop, it's a good library to have in your toolbox. Here's a quick example of what you could do with it. The version of least-squares we will use in statsmodels is called ordinary least-squares (OLS). There are many other versions of least-squares such as partial least squares (PLS) and weighted least squares (WLS).



Correlation Matrix with Heatmap

Correlation states how the features are related to each other or the target variable. Correlation can be positive (increase in one value of feature increases the value of the target variable) or negative (increase in one value of feature decreases the value of the target variable)

Heatmap makes it easy to identify which features are most related to the target variable, we will plot heatmap of correlated features using the seaborn library.

Number of new 1F house sold is positively (0.87) correlated with construction permits

Number of new 1F house sold is negatively (-0.56) correlated with Housing supply ratio

US Ave Houe price is negatively (-0.35) correlated with US Unemployment rate

Number of new 1F house sold is positively (0.87) correlated with construction permit

Monthly housing DATA ERROR TESTING

The train root mean squared error is : 70267.9274598809

The test root mean squared error is : 77818.89478743549

The Linear Regression coefficient parameters are : [1788.79238781 216.56884311 -75.96446774

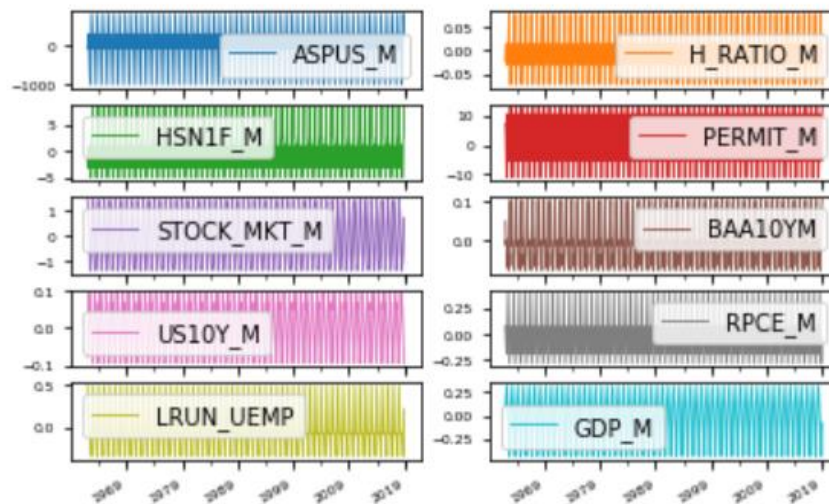
1773.71494551 56562.90186394 -16389.03294286 -9061.49679158 -5763.22028744 2896.18391227]

The Linear Regression intercept value is : 147549.50571956355

Visualize the seasonality of multiple time series

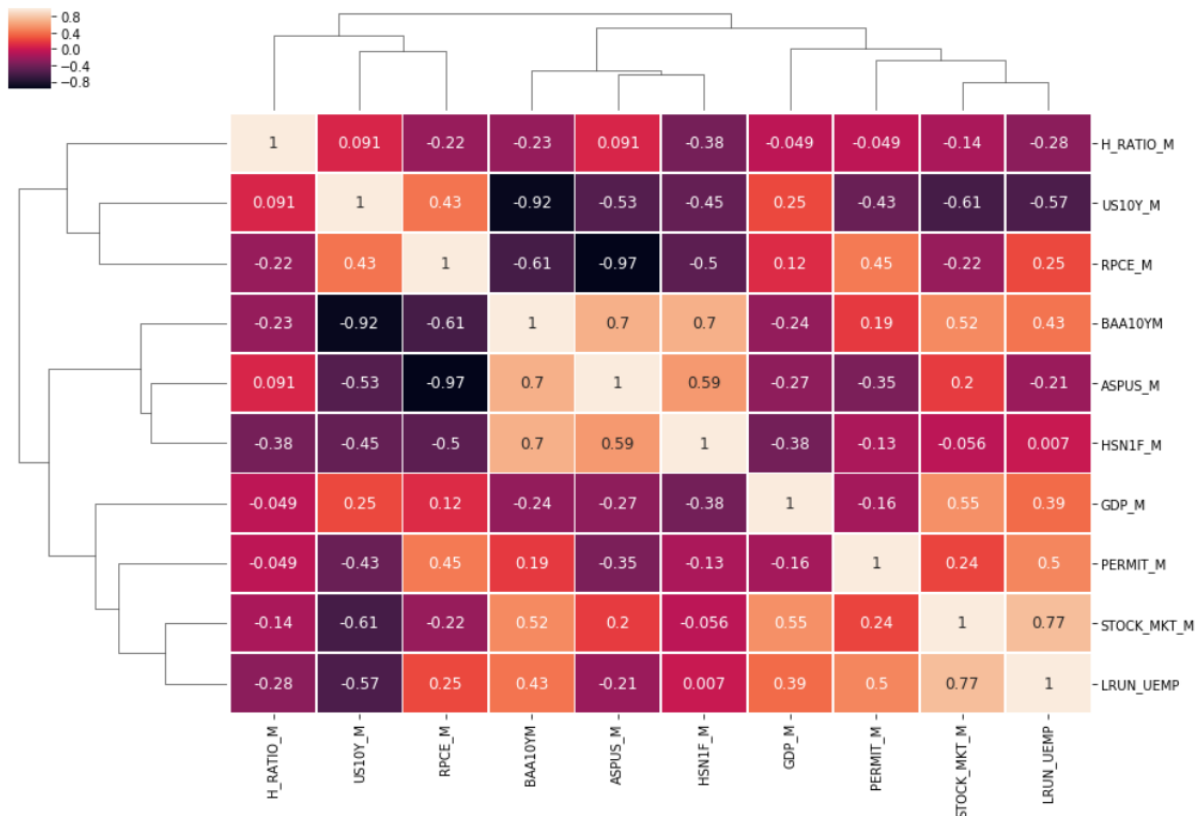
We will now extract the seasonality component of `h_m_df_decomp` to visualize the seasonality in these time series. Note that before plotting, you will have to convert the dictionary of seasonality components into a `DataFrame` using the `pd.DataFrame.from_dict()` function.

The seasonal component can be extracted using the `.seasonal` attribute. Use the `pd.DataFrame.from_dict()` to convert a dictionary to a `DataFrame`. Faceted plots of `DataFrame` `df` can be generated by setting the `subplots` argument to `True`



Compute the correlation between all columns in the `seasonality_df` `DataFrame` using the `spearman` method and assign the results to `seasonality_corr`. Create a new `clustermmap` of your correlation matrix.

Use the `.corr()` method along with the `method` argument to create a correlation matrix. To plot a `clustermmap`, use the `sns.clustermmap()` function.

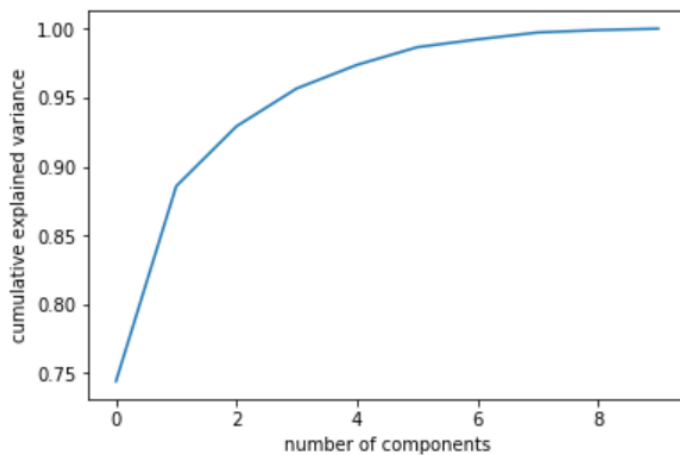


Visualizing predicted values

When dealing with time series data, it's useful to visualize model predictions on top of the "actual" values that are used to test the model.

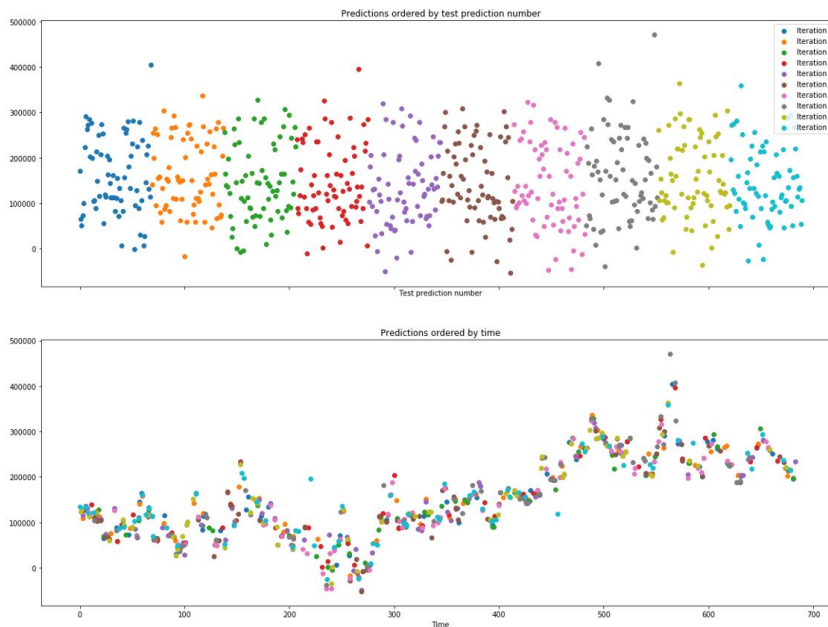
Variance of the PCA features

The dataset is 10-dimensional. But what is its intrinsic dimension? Make a plot of the variances of the PCA features to find out. As before, samples is a 2D array, where each row represents a fish. You'll need to standardize the features first.



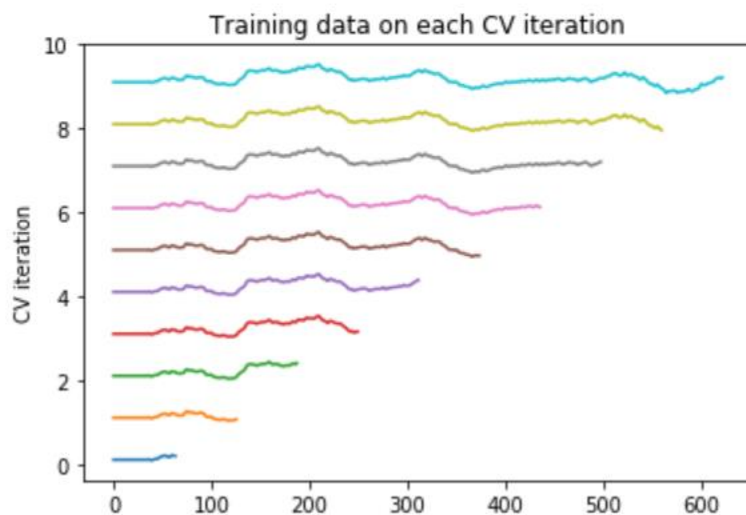
Cross-validation with shuffling

As you'll recall, cross-validation is the process of splitting your data into training and test sets multiple times. Each time you do this, you choose a different training and test set. In this exercise, you'll perform a traditional ShuffleSplit cross-validation on the company value data from earlier. Later we'll cover what changes need to be made for time series data. The data we'll use is the same historical price data for several large companies.



Time-based cross-validation

Finally, let's visualize the behavior of the time series cross-validation iterator in scikit-learn. Use this object to iterate through your data one last time, visualizing the training data used to fit the model on each iteration.



Note that the size of the training set grew each time when you used the time series cross-validation object. This way, the time points you predict are always after the timepoints we train on.

Time Series (Part 3)

Autocorrelation plots can be used to quickly discover patterns into your time series, so let's delve a little bit deeper into that!

Partial autocorrelation in time series data Like autocorrelation, the partial autocorrelation function (PACF) measures the correlation coefficient between a time-series and lagged versions of itself.

Time series decomposition

When visualizing time series data, we should look out for some distinguishable patterns:

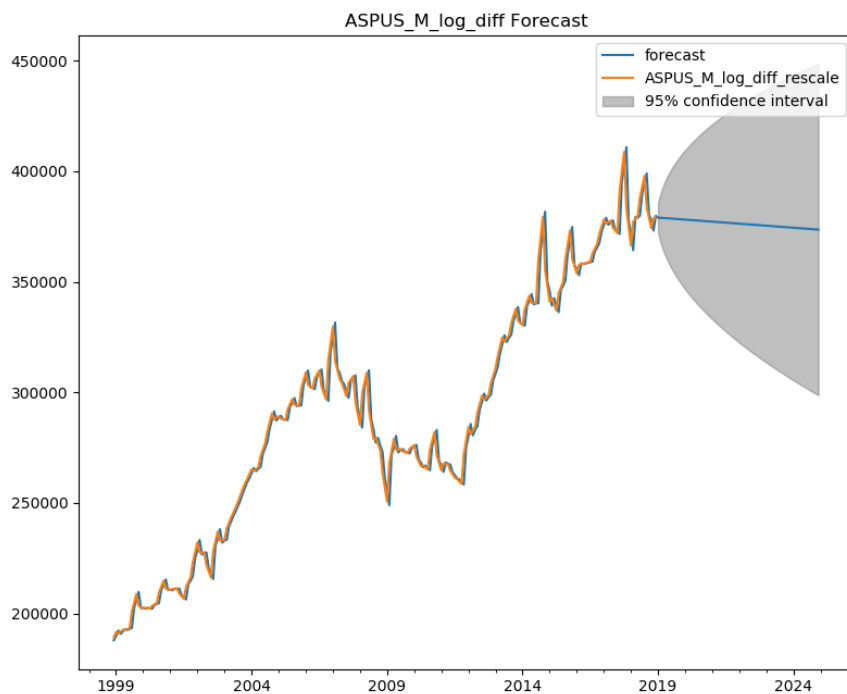
seasonality: does the data display a clear periodic pattern?

trend: does the data follow a consistent upwards or downward slope?

noise: are there any outlier points or missing values that are not consistent with the rest of the data?

FORECASTING ARIMA

Applying statistical modeling and machine learning to perform time-series forecasting.



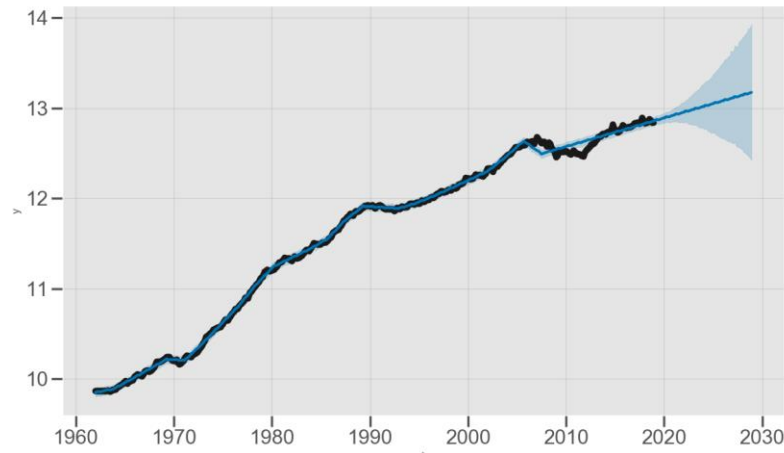
Forecasting Housing Market using FBProphet (Part 4)

Let us model some time-series data! using Facebook Prophet package.

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time

series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

Forecast using the 'predict' command:

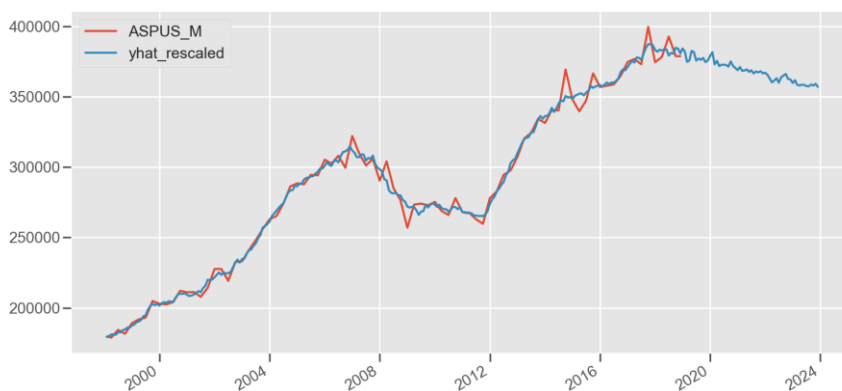


Visualizing Prophet models

In order to build a useful dataframe to visualize our model versus our original data, we need to combine the output of the Prophet model with our original data set, then we'll build a new chart manually using pandas and matplotlib. First, let's set our dataframes to have the same index of ds

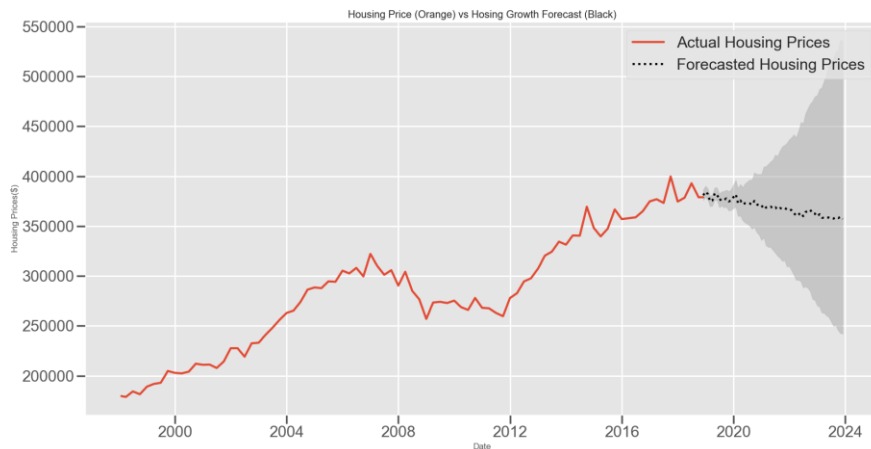
Let's take a look at the sales and yhat_rescaled data together in a chart.'

```
viz_df[['ASPUS_M', 'yhat_rescaled']].plot()
```



We can see downward trends in 60 months for AVE US HOUSING PRICE

Time to plot: let's plot everything to get the 'final' visualization of our monthly housing data and forecast with errors



This visualization is much better (in my opinion) than the default fbprophet plot. It is much easier to quickly understand and describe what's happening. The orange line is actual sales data and the black dotted line is the forecast. The gray shaded area is the uncertainty estimation of the forecast.

For next Five Years, we can see Ave Housing Price downward trend. Price range will remain between \$248000 and \$550000

Forecasting US housing market using Deep Learning (Part 5)

Time series forecasting is challenging, especially when working with long sequences, noisy data, multi-step forecasts and multiple input and output variables.

Deep learning methods offer a lot of promise for time series forecasting, such as the automatic learning of temporal dependence and the automatic handling of temporal structures like trends and seasonality.

Time series data can be phrased as supervised learning.

Given a sequence of numbers for a time series dataset.

- We will discover how to develop a Long Short-Term Memory Neural Network model or LSTM for univariate time series forecasting.
- We can define a simple univariate problem as a sequence of integers, fit the model on this sequence and have the model predict the next value in the sequence.
- The LSTM model expects three-dimensional input with the shape [samples, timesteps, features]. We will define the data in the form [samples, timesteps] and reshape it accordingly.
- We will use one LSTM layer to process each input sub-sequence of 3 time steps, followed by a Dense layer to interpret the summary of the input sequence. The model uses the efficient Adam version of stochastic gradient descent and optimizes the mean squared error ('mse' or 'rmse') loss function.
- Once the model is defined, it can be fit on the training data and the fit model can be used to make a prediction.

Import Monthly Housing Price Data

```
h_m_df = pd.read_csv('C:/scripts/capstone2/h_m_df.csv', index_col='DATE', parse_dates=True)
```

Specifying a model

Compiling the model

Fitting the model

Classification models

Making predictions

Housing price Prediction Using CNN

```
h_m_df = pd.read_csv('C:/scripts/capstone2/h_m_df.csv', index_col='DATE', parse_dates=True)
```

Modeling

Define Network

Compile Network

Train Network

Evaluate Network

LSTM-RNN to forecast time-series

RNN's (LSTM's) are pretty good at extracting patterns in input feature space, where the input data spans over long sequences. Given the gated architecture of LSTM's that has this ability to manipulate its memory state, they are ideal for such problems.

Visualising the results



R^2 is a statistic that will give some information about the [goodness of fit](#) of a model. In regression, the R^2 coefficient of determination is a statistical measure of how well the regression predictions approximate the real data points. An R^2 of 1 indicates that the regression predictions perfectly fit the data.

Values of R^2 outside the range 0 to 1 can occur when the model fits the data worse than a horizontal hyperplane. This would occur when the wrong model was chosen, or nonsensical constraints were applied by mistake. If equation 1 of Kvålseth^[1] is used (this is the equation used most often), R^2 can be less than zero.

Both Actual price and Predicted price is indicating that overheated housing market is slowing down. Pick Ave Housing Price was \$400000 and now down to \$378000. We can see 5.5% down from High Price.

