IST-718 Homework/Lab-2 (Week6)

Prof. Jonathan M Fox
Student: Sharat Sripada (vssripad)

Introduction

The objective of this lab is to analyze a dataset *Zip_Zhvi_SingleFamilyResidence.csv* related to property prices across states in USA and make recommendations about THREE zip-codes where Syracuse Real Estate Investment Trust could possibly invest.

Here are some sought out objectives in achieving that goal:

- Data cleanup and EDA
- Data analysis to develop time series plots for Arkansas metro areas:
    - Hot Springs, Little Rock, Fayetteville, Searcy
    - Present all values from 1997 to present
    - Average at the metro area level
- Develop model(s) for forecasting average median housing value by zip code for 2020
- Use the historical data from 1996 through 2019 as training data

Exploratory Data Analysis

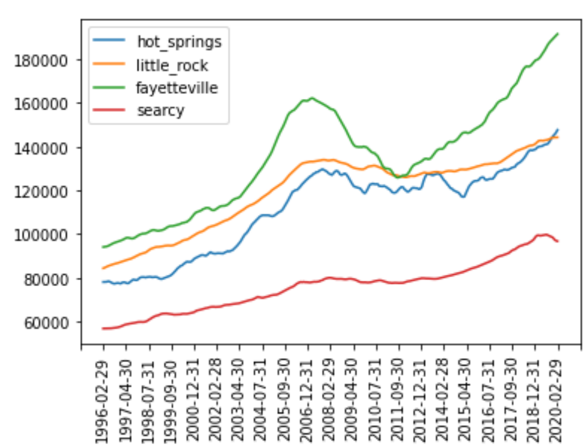
Initial EDA included data-cleanup, exploring dimensions/shape of data etc. Cleaning up or removing NaN rows showed over 60% reduction in data:

```
Original DF shape: (30464, 300)
Trimmed DF shape: (12033, 300)
Reduction/Loss percent:  60.50091911764706
```

For the purpose of experiments presented here, NaN values as smaller dataframes were derived and worked on.

Initial Analysis

Data analysis involved generating a time-series plot for Ankansas metro areas as seen below:



Fayetteville seems to have the maximum growth over this time:

```
Overall growth % from 1996-2020
------------------------------
(1) Hot Springs (Blue): 89.27933752000985
(2) Little Rock (Orange): 71.1098515591654
(3) Feyetteville (Green): 103.73990650233746
(4) Searcy: (Red):  70.54230049765292
```

NOTE

Variations or volatilities are important with investments. It also helps set exit/entry criteria. Couple of observations here:

- While Fayetteville has the highest growth, we should pay particular attention to the period between 2006-2011
- Compare that with Searcy that steadily delivery 70% ROI during this timeframe

ARIMA model on dataset

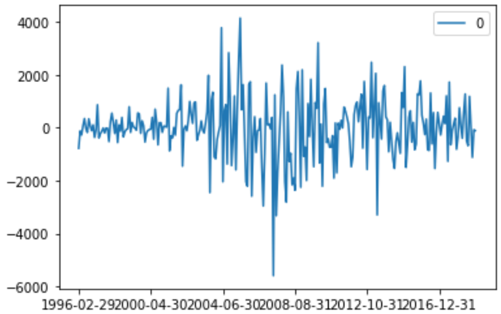
To run and generate an ARIMA model, data was split into train and test set as follows:

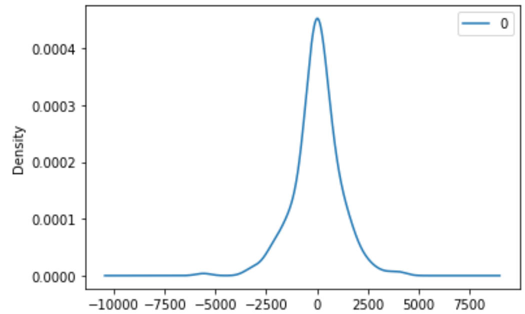
- data from 01-31-1996 to 02-28-2019 was used as train and
- data from 03-01-2019 – 2020 was used as test

Below is the summary of the ARIMA model:

```
                              ARIMA Model Results
==============================================================================
Dep. Variable:                    D.y   No. Observations:                  276
Model:                 ARIMA(5, 1, 0)   Log Likelihood               -2338.802
Method:                       css-mle   S.D. of innovations           1155.973
Date:                Sun, 21 Feb 2021   AIC                           4691.604
Time:                        05:15:44   BIC                           4716.947
Sample:                    02-29-1996   HQIC                          4701.773
                         - 01-31-2019
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const        672.2730    579.822      1.159      0.247    -464.158    1808.704
ar.L1.D.y      0.6086      0.059     10.246      0.000       0.492       0.725
ar.L2.D.y      0.0258      0.070      0.370      0.711      -0.111       0.162
ar.L3.D.y      0.0612      0.070      0.879      0.380      -0.075       0.198
ar.L4.D.y      0.0462      0.070      0.664      0.507      -0.090       0.182
ar.L5.D.y      0.1445      0.059      2.441      0.015       0.028       0.260
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.0607           -0.0000j            1.0607           -0.0000
AR.2            0.6229           -1.3547j            1.4910           -0.1814
AR.3            0.6229           +1.3547j            1.4910            0.1814
AR.4           -1.3130           -1.1008j            1.7134           -0.3890
AR.5           -1.3130           +1.1008j            1.7134            0.3890
-----------------------------------------------------------------------------
```

Residual plots and kernel density estimation plots were drawn using the train dataset:





And summary of descriptive stats:

```
count    276.000000
mean       5.899403
std     1158.685485
min    -5589.099356
25%     -519.020501
50%       16.428709
75%      581.065741
max     4141.559927
```

Finally, using the *Rolling Forecast ARIMA Model* function we make predictions and check for its accuracy on the train-data.

The code would help to:

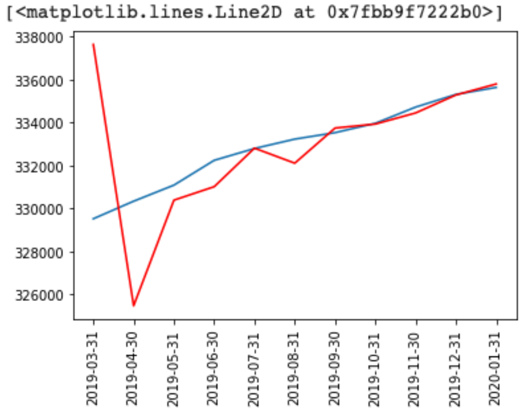
- Explore means to perform a one-step rolling forecast and re-create the ARIMA model after each new observation
- Manually keep track of all observations in a list that is seeded with the training data and to which new observations are appended in each iteration

```python
# Walk forward validation
for t in range(df_test.shape[0]):
    model = ARIMA(history, order=(5,1,0))
    model_fit = model.fit(trend='nc', disp=0)
    output = model_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = df_test[t]
    history.append(float(obs))
    print('predicted=%f, expected=%f' % (yhat, obs))

# Evaluate forecasts
rmse = sqrt(mean_squared_error(df_test, predictions))
print('Test RMSE: %.3f' % rmse)
```

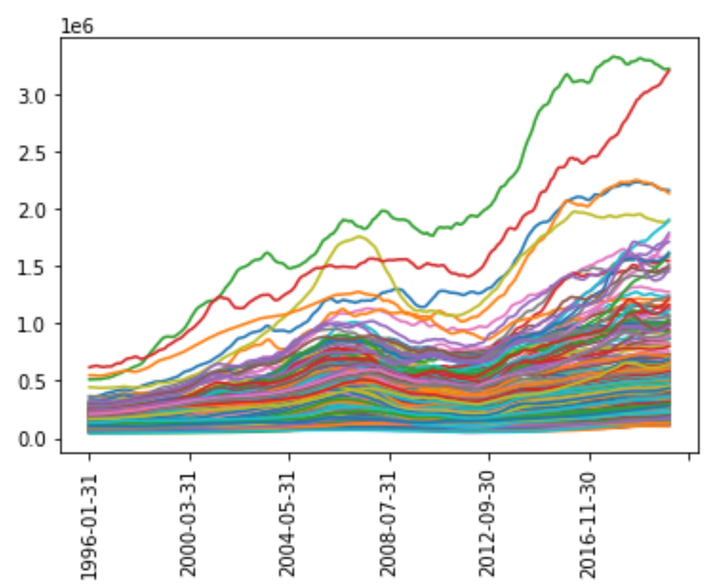
A simple line plot shows the prediction (in red) eventually close in with the actual value (Test RMSE was 2902.906):

```
predicted=337632.228243, expected=329529.000000
predicted=325482.303125, expected=330344.500000
predicted=330393.011877, expected=331091.500000
predicted=331019.454610, expected=332245.500000
predicted=332811.563191, expected=332795.500000
predicted=332112.255091, expected=333232.000000
predicted=333746.013489, expected=333531.500000
predicted=333932.637034, expected=333967.000000
predicted=334449.188744, expected=334721.000000
predicted=335284.521202, expected=335309.500000
predicted=335794.847722, expected=335635.500000
Test RMSE: 2902.906
[<matplotlib.lines.Line2D at 0x7fbb9f7222b0>]
```



The bigger question

Initial analysis with ARIMA model seems encouraging to extend and answer the questions regarding zip-code recommendations. We start with grouping the data-frame by 'RegionName' (seems to accurately indicate the state, county and can be safely assumed as the zip-code):



The plot shows similar trends as the Arkansas county plot but would need to be handled differently since we are now grouping it by 'RegionName'. Here's an excerpt of the data-frame:

RegionName	2148	2155	2169	2360	6010	7002	7030	7087	7093	7302
1996-01-31	143653.0	171541.0	141388.0	145894.0	114604.0	163031.0	293442.0	125382.0	155697.0	167873.0
1996-02-29	143399.0	170763.0	142114.0	145636.0	114350.0	162311.0	293668.0	125276.0	155753.0	167498.0
1996-03-31	143499.0	170914.0	142300.0	145584.0	113844.0	162009.0	294189.0	125029.0	155444.0	167428.0
1996-04-30	143469.0	170811.0	143007.0	145616.0	113400.0	161249.0	294911.0	125064.0	155088.0	167100.0
1996-05-31	143530.0	171332.0	143531.0	145804.0	113361.0	160706.0	296033.0	124710.0	155147.0	167312.0

Finally, to be able to split this into train/test data we transpose it, split it by similar dates as before and transpose it back as highlighted in the code:

```
for column in df_zipcode_clean.T[train_columns].T:
    history = [i for i in df_zipcode_clean[column]]
    predictions = []
    # Walk forward validation
    for t in range(df_zipcode_clean.T[test_columns].T.shape[0]):
        model = ARIMA(history, order=(5,1,0))
        model_fit = model.fit(trend='nc', disp=0)
        output = model_fit.forecast()
        yhat = output[0][0]
        # Evaluate forecasts
        predictions.append(yhat)
        obs = df_zipcode_clean.T[test_columns].T[column][t]
        history.append(float(obs))
        # Print for first & last iteration only
        if (column == 1001 or column == 99508) and t == 0:
            print('predicted=%f, expected=%f' % (yhat, obs))
```

We sort the dictionary *sorted_rmse_by_zipcode* using sorted() and show the zip-codes with the least RMSE. These likely are the best models and we can further detail analysis like drawing up the ARIMA summary, residual and kde plots if desired.

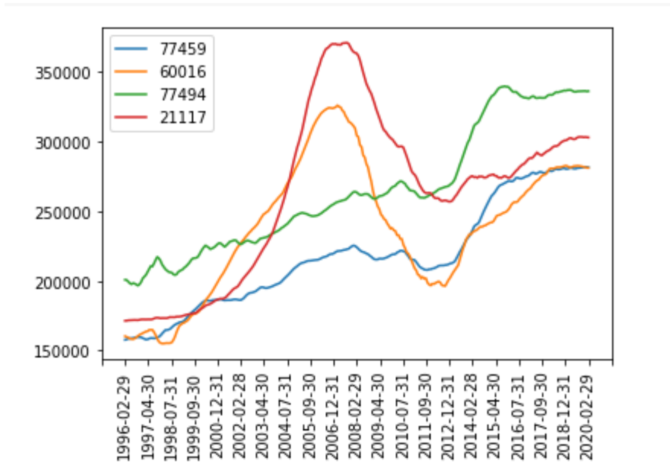
Conclusions

Based on the analysis presented thus far, the top FOUR zip-codes were identified as 77459, 60016, 77494, 21117. These correspondingly map to:

```
138    Houston-The Woodlands-Sugar Land
Name: Metro, dtype: object
228    Chicago-Naperville-Elgin
Name: Metro, dtype: object
3    Houston-The Woodlands-Sugar Land
```

```
Name: Metro, dtype: object
246    Baltimore-Columbia-Towson
Name: Metro, dtype: object
```

Plotting their investment returns between 1996-2020 shows returns in the range 67-78%:




```
Overall growth % from 1996-2020
------------------------------
(1) 77459 (Blue): 78.45621023653513
(2) 60016 (Orange): 75.36585518063417
(3) 77494 (Green): 67.41330091718439
(4) 21117 (Red):  76.75200504325291
```

Future Work

The model can further be strengthened by adding data from the Bureau of Labor Statistics and Census data which was not considered here.