Table of Contents

1 Final Project Submission

- Student name: Jim Petoskey

- Student pace: Self-paced
- Scheduled project review date/time: Thursday May 5, 2022 at 9:00 am EST
- Instructor name: Abhineet Kulkarni
- Blog post URL: Time Series Decomposition - Spotting Seasonality (https://dev.to/jpetoskey/time-series-decomposition-spotting-seasonality-4e71)
- Time Series Zillow Starter Repository (https://github.com/learn-co-curriculum/dsc-phase-4-choosing-a-dataset/tree/main/time-series)

1.1 Summary:

- Top Zip Codes to Invest in:
  - Oakland, CA 94606
    - High Return (68.1%), Low Risk
  - Lake Worth, FL 33460
    - Highest Return (93%), Modest Risk
  - Boise, ID 83703
    - Moderately high return (55.1%), Low Risk

- Process:

1. Measure rate of return for every zip code in US from April 2012 to April 2018
2. Choose top 15 zip codes by rate of return
3. Added top Mountain Towns and Mid-size Cities from Adding Breadth of Zip Codes Section
4. Model price predictions, measure accuracy with RMSE
5. Choose zip codes which are modeled most accurately
6. Re-model those zip codes, forecast rate of return, plot predictions and confidence interval
7. Propose investment in zip codes with lower risk, as measured by the confidence interval and RMSE, and high predicted returns over 5 year investment timeline

2 Import Libraries

In [4]:

```
# Import necessary libraries
import warnings
warnings.filterwarnings('ignore')
import itertools
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
from matplotlib.pylab import rcParams
plt.style.use('default')
from statsmodels.tsa.seasonal import seasonal_decompose
```

executed in 7ms, finished 10:45:47 2022-05-03

3 Define Functions

3.1 Melt

In [5]:

```
def melt_data(df):
    """
    Takes the zillow_data dataset in wide form or a subset of the zillow_datas
    Returns a long-form datetime dataframe
    with the datetime column names as the index and the values as the 'values'

    If more than one row is passes in the wide-form dataset, the values column
    will be the mean of the values from the datetime columns in all of the row
    """

    melted = pd.melt(df, id_vars=['ZipCode','SizeRank', 'City', 'State', 'Metr
    melted['time'] = pd.to_datetime(melted['time'], infer_datetime_format=True
    melted = melted.dropna(subset=['value'])
    return melted.groupby('time').aggregate({'value':'mean'})
```

executed in 7ms, finished 10:45:49 2022-05-03

3.2 Stationarity Check

In [6]:

```
def stationarity_check(TS):

    # Import adfuller
    from statsmodels.tsa.stattools import adfuller

    # Calculate rolling statistics
    roll_mean = TS.rolling(window=8, center=False).mean()
    roll_std = TS.rolling(window=8, center=False).std()

    # Perform the Dickey Fuller test
    dftest = adfuller(TS)

    # Plot rolling statistics:
    fig = plt.figure(figsize=(12,6))
    orig = plt.plot(TS, color='blue',label='Original')
    mean = plt.plot(roll_mean, color='red', label='Rolling Mean')
    std = plt.plot(roll_std, color='black', label = 'Rolling Std')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.show(block=False)

    # Print Dickey-Fuller test results
    print('Results of Dickey-Fuller Test: \n')

    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic', 'p-value',
                                             '#Lags Used', 'Number of Observat
    for key, value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print(dfoutput)

    return None
```

executed in 8ms, finished 10:45:50 2022-05-03

4 Choose Zip Codes by Percent Increase in Median Price

- Time Period: April 2012 to April 2018

4.1 Create Melted Dataframe of ZipCodes

In [7]:

```
# Create dataframe from zillow median home price data
df = pd.read_csv('zillow_data.csv')

# make copy of df
df1 = df.copy()

# Rename RegionName as ZipCode
df1 = df1.rename(columns={"RegionName": "ZipCode"})

# add column with percent change between 2017 and 2012 median value
df1['Percent Change'] = (df['2018-04'] / df['2012-04']) * 100

# create new dataframe with top 10 zip codes for Percent Change
df_percent = df1.sort_values('Percent Change',ascending = False)

# new df_percent with only top ten results
df_percent = df_percent.head(15)

# add mountain towns and mid-size cities
mount_mid = df1.loc[[9058,4502,4293,4442,6810,5764]]
df_percent = pd.concat([df_percent,mount_mid])

# Remove Raw Difference and Percent Change for melting
prospects = df_percent.drop(columns=['Percent Change', 'RegionID'])

# Remove Column with missing data
prospects = prospects.drop([1946])
```

executed in 967ms, finished 10:45:52 2022-05-03

In [8]:

```
1 prospects
```

executed in 70ms, finished 10:45:53 2022-05-03

Out[8]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05	
1477	94601	Oakland	CA	San Francisco	Alameda	1478	114600.0	114500.0	
1239	94590	Vallejo	CA	Vallejo	Solano	1240	108200.0	107800.0	
2627	33460	Lake Worth	FL	Miami-Fort Lauderdale	Palm Beach	2628	59800.0	59900.0	
1681	94606	Oakland	CA	San Francisco	Alameda	1682	120400.0	120300.0	
1853	94804	Richmond	CA	San Francisco	Contra Costa	1854	171300.0	170300.0	
1960	89104	Las Vegas	NV	Las Vegas	Clark	1961	94800.0	94700.0	
474	85008	Phoenix	AZ	Phoenix	Maricopa	475	61600.0	61900.0	
4298	95824	Sacramento	CA	Sacramento	Sacramento	4299	73800.0	73400.0	
2409	33404	Riviera Beach	FL	Miami-Fort Lauderdale	Palm Beach	2410	56800.0	56700.0	
2661	33705	Saint Petersburg	FL	Tampa	Pinellas	2662	54600.0	54800.0	
2934	89107	Las Vegas	NV	Las Vegas	Clark	2935	92300.0	92400.0	
815	89115	Las Vegas	NV	Las Vegas	Clark	816	92500.0	92500.0	
2775	95820	Sacramento	CA	Sacramento	Sacramento	2776	71800.0	71800.0	
5830	48240	Redford	MI	Detroit	Wayne	5831	67800.0	68200.0	
9058	84101	Salt Lake City	UT	Salt Lake City	Salt Lake	9059	85100.0	85400.0	
4502	48009	Birmingham	MI	Detroit	Oakland	4503	242700.0	242600.0	2
4293	66104	Kansas City	KS	Kansas City	Wyandotte	4294	41300.0	41200.0	
4442	84606	Provo	UT	Provo	Utah	4443	142700.0	141400.0	
6810	87506	Santa Fe	NM	Santa Fe	Santa Fe	6811	227000.0	227400.0	2
5764	83703	Boise	ID	Boise City	Ada	5765	107500.0	107500.0	

20 rows × 271 columns

In [9]:

```
# Check for missing data
print(prospects.isnull().sum().sum())
```

executed in 12ms, finished 10:45:53 2022-05-03

0

4.2 Print Decomposition Chart for each Zip Code

In [10]:

```
# Melt and decompose each zip from prospects df
for zipcode in prospects['ZipCode']:
    zips = prospects.loc[prospects['ZipCode'] == zipcode]
    m1 = melt_data(zips)
    m1[zips.iloc[0,1],zips.iloc[0,2],str(zipcode)] = m1['value']
    m1 = m1.drop('value', axis=1)

    #Apply seasonal_decompose()
    decomposition = seasonal_decompose(np.log(m1))

    # Gather the trend, seasonality, and residuals
    trend = decomposition.trend
    seasonal = decomposition.seasonal
    residual = decomposition.resid

    # Plot gathered statistics
    plt.figure(figsize=(12,8))
    plt.subplot(411)
    plt.title(str(m1.columns.values))
    plt.plot(np.log(m1), label='Original', color='blue')
    plt.legend(loc='best')
    plt.subplot(412)
    plt.plot(trend, label='Trend', color='blue')
    plt.legend(loc='best')
    plt.subplot(413)
    plt.plot(seasonal,label='Seasonality', color='blue')
    plt.legend(loc='best')
    plt.subplot(414)
    plt.plot(residual, label='Residuals', color='blue')
    plt.legend(loc='best')
    plt.tight_layout()
```

executed in 17.0s, finished 10:46:10 2022-05-03

4.3 Seasonality Trends Fascinating

- Seasonality present in every zip code, but how the trend occurs is different for each zip code.
- Certainly justifies using SARIMA instead of ARIMA due to seasonal pattern.

5 Adding Breadth of Zip Codes

- Mountain Towns and Mid-Size Cities

5.1 Mountain Towns

- Boise, ID
- Santa Fe, NM
- Boulder, CO
- Provo, UT
- Bend, OR

5.1.1 Create Mountain Towns df

In [26]:

```
mountain_list = []
mountain_list.append(['Boise', 'Boulder', 'Provo', 'Bend'])

mountain_towns = pd.DataFrame()

mt = pd.DataFrame(mountain_list)
mt = mt.T

for i in mt[0]:
    # Create df from new city
    ok = df1[df1['City'] == str(i)]

    # add column with percent change between 2017 and 2012 median value
    ok['Percent Change'] = (ok['2018-04'] / ok['2012-04']) * 100

    # create new dataframe with top 10 zip codes for Percent Change
    ok = ok.sort_values('Percent Change',ascending = False)

    # new df_percent with only top ten results
    ok = ok.head(2)

    # concatenate to mountain towns df
    mountain_towns = pd.concat([ok,mountain_towns])

# Create df from new city
santa_fe = df1[df1['City'] == 'Santa Fe']

# Choose top zip code and my zip code from Santa Fe, NM
santa_fe = santa_fe[(santa_fe['ZipCode'] == 87505) | (santa_fe['ZipCode'] == 8

# concatenate to mountain towns df
mountain_towns = pd.concat([santa_fe,mountain_towns])

mountain_towns = mountain_towns.drop(['RegionID','Percent Change'], axis=1)

mountain_towns
```

executed in 110ms, finished 11:50:25 2022-05-03

Out[26]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05	1996-
1776	87505	Santa Fe	NM	Santa Fe	Santa Fe	1777	188200.0	188400.0	18850(
6810	87506	Santa Fe	NM	Santa Fe	Santa Fe	6811	227000.0	227400.0	22790(
1446	97701	Bend	OR	Bend	Deschutes	1447	136300.0	136100.0	13590(
706	97702	Bend	OR	Bend	Deschutes	707	160300.0	160300.0	16030(

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05	1996-
4081	84601	Provo	UT	Provo	Utah	4082	142700.0	140500.0	13840(
4442	84606	Provo	UT	Provo	Utah	4443	142700.0	141400.0	14010(
6115	80305	Boulder	CO	Boulder	Boulder	6116	187900.0	187700.0	18750(
3363	80303	Boulder	CO	Boulder	Boulder	3364	231100.0	230300.0	22950(
5764	83703	Boise	ID	Boise City	Ada	5765	107500.0	107500.0	10750(
2962	83705	Boise	ID	Boise City	Ada	2963	83000.0	83000.0	8300(

10 rows × 271 columns

In [27]:

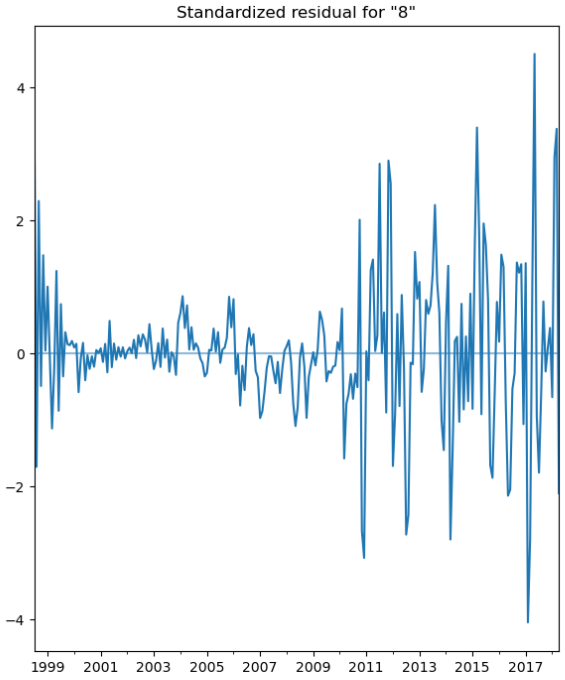
```
# Create list for final predicted value
pred_list_2 = []

# Construct model from entirety of data - no train/test
# iterate over prospects to extract optimal SARIMA values,
# instantiate SARIMAX model, make predictions
# and produce forecast with confidence intervals
for zipcode in mountain_towns['ZipCode']:
    zips = mountain_towns.loc[mountain_towns['ZipCode'] == zipcode]
    m1 = melt_data(zips)
    m1[str(zipcode)] = m1['value']
    m1 = m1.drop('value', axis=1)

    # do NOT create train and test sets
    #train = m1.iloc[:238]
    #test = m1.iloc[238:]

    # Run a grid with pdq and seasonal pdq parameters calculated above and ge
    ans = []
    outputs = []
    comb_list = []
    combs_list = []

    for comb in pdq:
        for combs in pdqs:
            try:
                mod = sm.tsa.statespace.SARIMAX(m1,
                                                order=comb,
                                                seasonal_order=combs,
                                                enforce_stationarity=False,
                                                enforce_invertibility=False)

                output = mod.fit()
                ans.append([comb, combs, output.aic])
                outputs.append([output.aic])
                comb_list.append([comb])
                combs_list.append([combs])
                AIC_df = pd.DataFrame(list(zip(outputs, comb_list, combs_list
                #print('ARIMA {} x {}12 : AIC Calculated ={}'.format(comb, co
            except:
                continue

    # Print the lowest AIC and it's parameters for SARIMA
    AIC_df[0] = AIC_df[0].str.get(0)
    best_AIC = AIC_df.copy()
    lowest_AIC = min(best_AIC[0])
    lowest_AIC_row = AIC_df.loc[AIC_df[0] == lowest_AIC]
    #print(lowest_AIC_row)

    # call value for comb and combs for model input
    lowest_AIC_row[1] = lowest_AIC_row[1].str.get(0)
    lowest_AIC_row[2] = lowest_AIC_row[2].str.get(0)
```

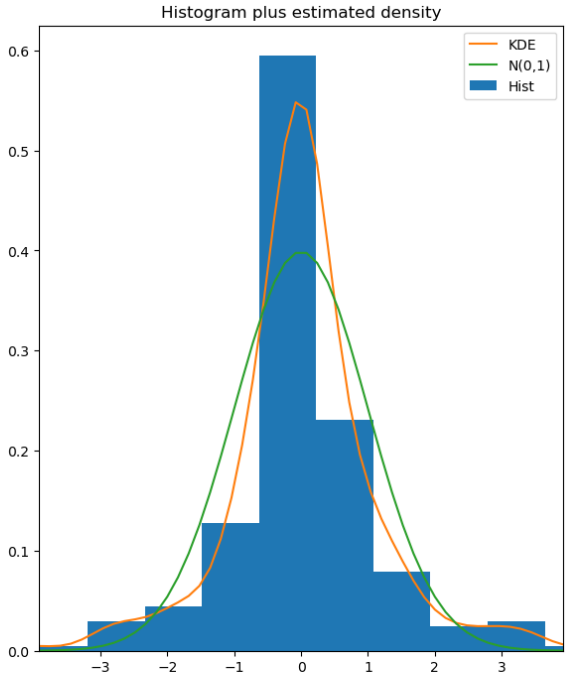
```
    arima = lowest_AIC_row.iat[0,1]
    s = lowest_AIC_row.iat[0,2]
    print(str(m1.columns.values))
    print(arima, s)

    # Plug the optimal parameter values into a new SARIMAX model
    ARIMA_MODEL = sm.tsa.statespace.SARIMAX(m1,
                                            order=arima,
                                            seasonal_order=s,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)

    # Fit the model and print results
    output = ARIMA_MODEL.fit()

    print(output.summary().tables[1])

    # Call plot_diagnostics() on the results calculated above
    output.plot_diagnostics(figsize=(15, 18))
    plt.show()

    # Get actual results in dataframe
    price_truth = m1['2018-04-01':]

    # Get forecast 500 steps ahead in future
    prediction = output.get_forecast(steps=60)

    # Create predictions df
    pred = prediction.predicted_mean

    # Add prediction for 5 year forecasted value to list
    pred_list_2.append(pred['2023-04-01'])

    # Get confidence intervals of forecasts
    pred_conf = prediction.conf_int()

    # Only use data from April 2010 to improve visual of forecast
    plot_vals = m1.loc['2010-04-01':]

    # Plot future predictions with confidence intervals
    ax = plot_vals.plot(label='observed', figsize=(20, 15), color='black')
    prediction.predicted_mean.plot(ax=ax, label='Forecast', color='green')
    ax.fill_between(pred_conf.index,
                    pred_conf.iloc[:, 0],
                    pred_conf.iloc[:, 1], color='k', alpha=0.25)
    ax.set_xlabel('Date').set_fontsize(30)
    ax.set_ylabel('Median House Price ($)').set_fontsize(30)
    ax.set_title('Five Year Median House Price Forecast').set_fontsize(40)

    for item in (ax.get_xticklabels() + ax.get_yticklabels()):
        item.set_fontsize(20)

    right_side = ax.spines["right"]
    right_side.set_visible(False)
```

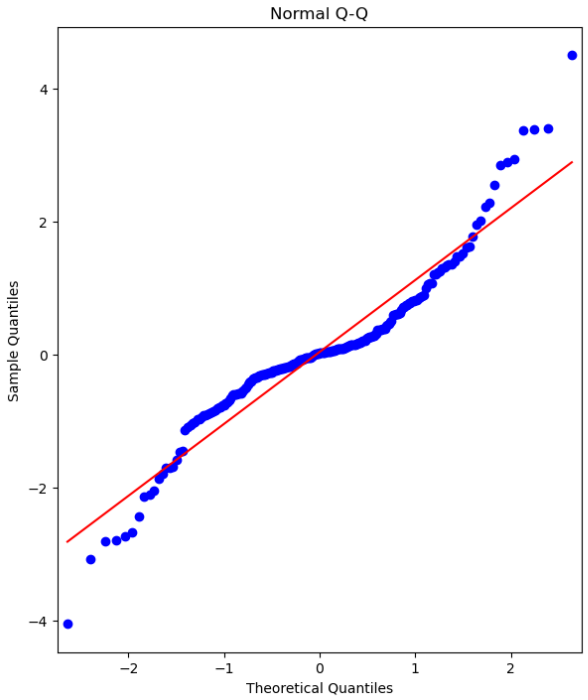
```
107
108     top_side = ax.spines["top"]
109     top_side.set_visible(False)
110
111     mylabels = 'Actual', 'Forecast', 'Confidence Interval'
112     plt.legend(fontsize=30,labels=mylabels,bbox_to_anchor=(0.6,0.85))
113     plt.show()
114     print()
115     print()
116
117
```

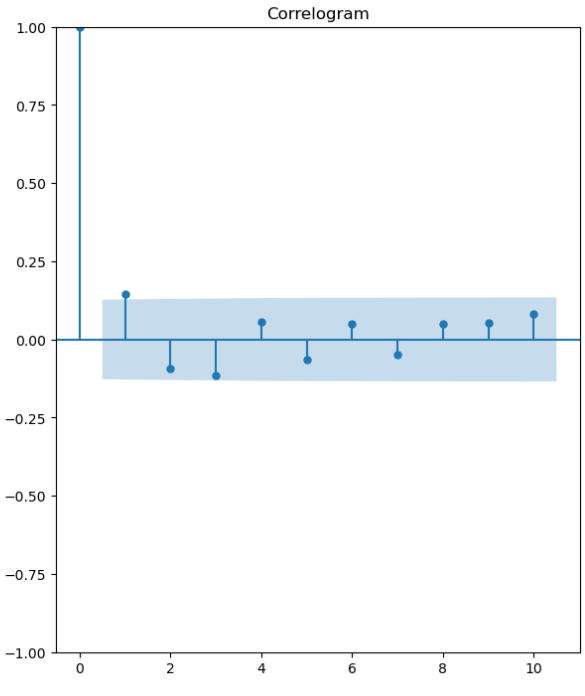
executed in 4m 43s, finished 11:55:23 2022-05-03

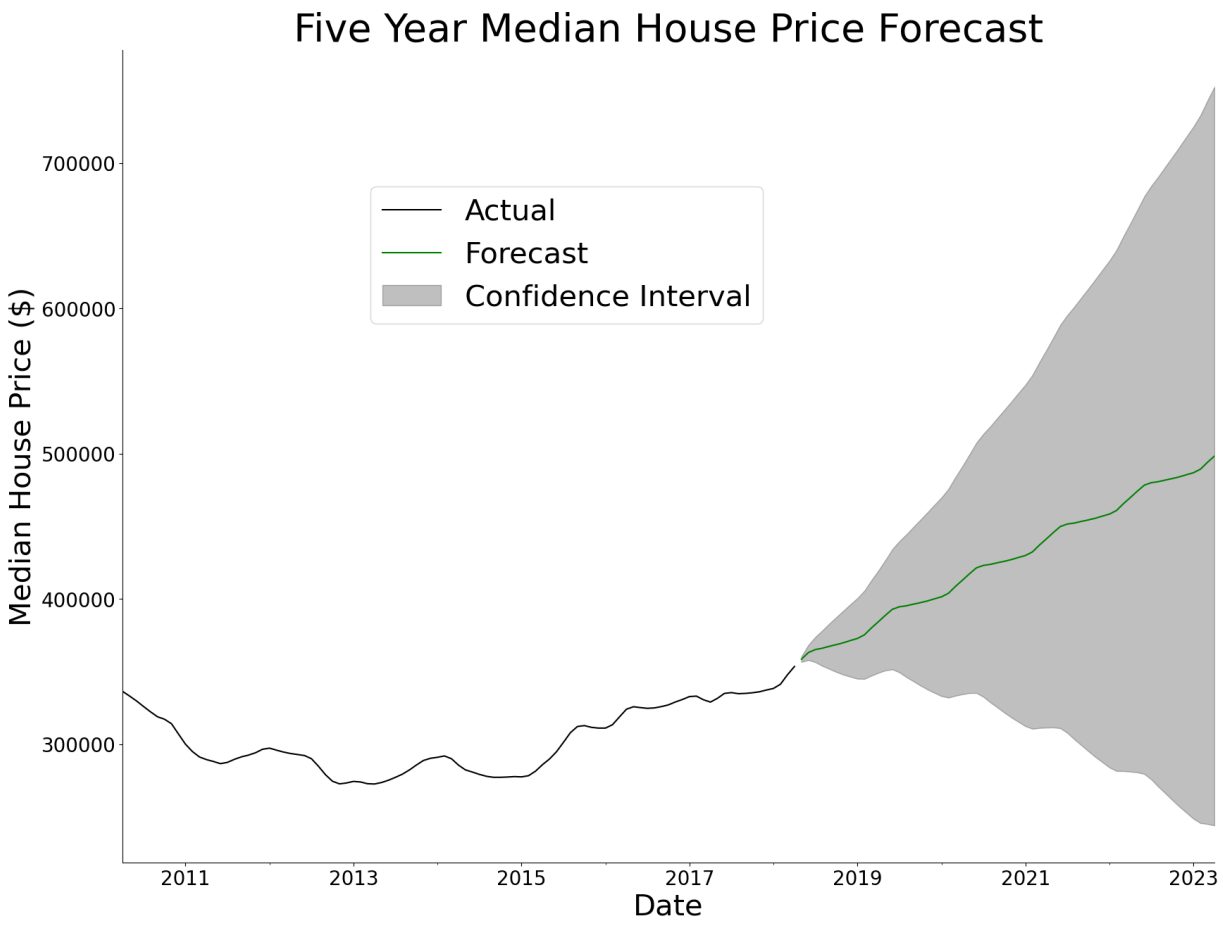
```
['87505']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.7393      0.033     22.200      0.000       0.674
0.805
ma.L1          0.7373      0.028     26.530      0.000       0.683
0.792
ma.S.L12      -0.2625      0.027     -9.647      0.000      -0.316
-0.209
sigma2      1.005e+06   6.32e+04     15.899      0.000    8.81e+05
1.13e+06
==============================================================================
```


Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

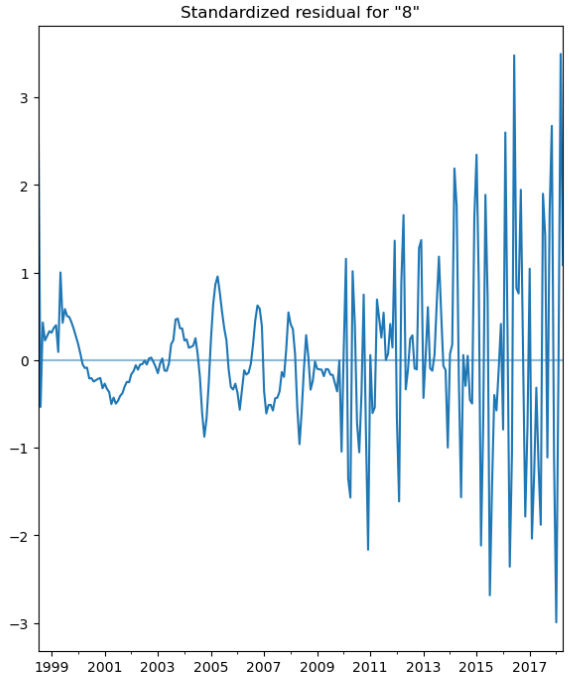

Histogram plus estimated density
KDE
N(0,1)
Hist
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

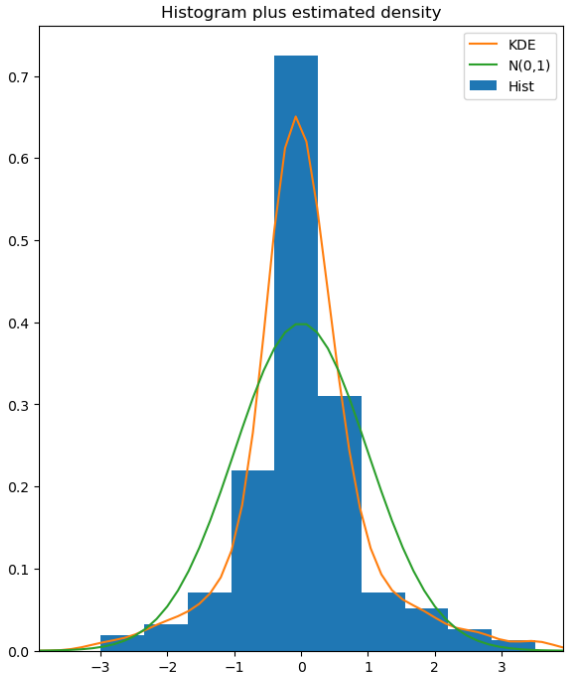

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2

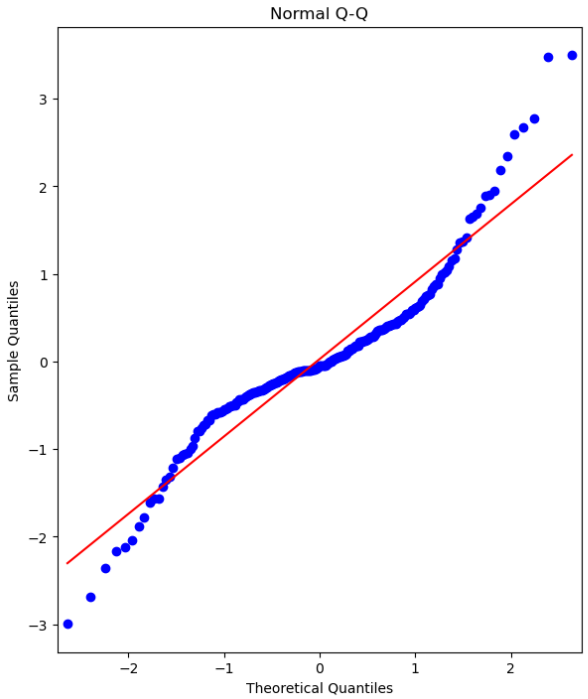

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

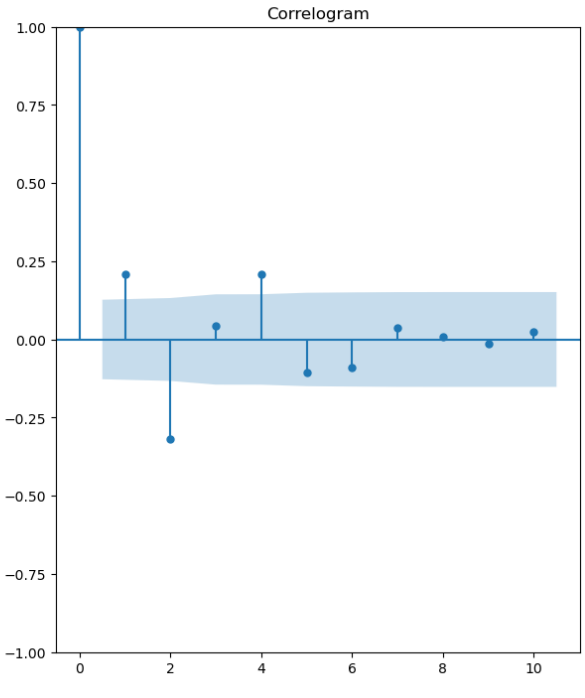


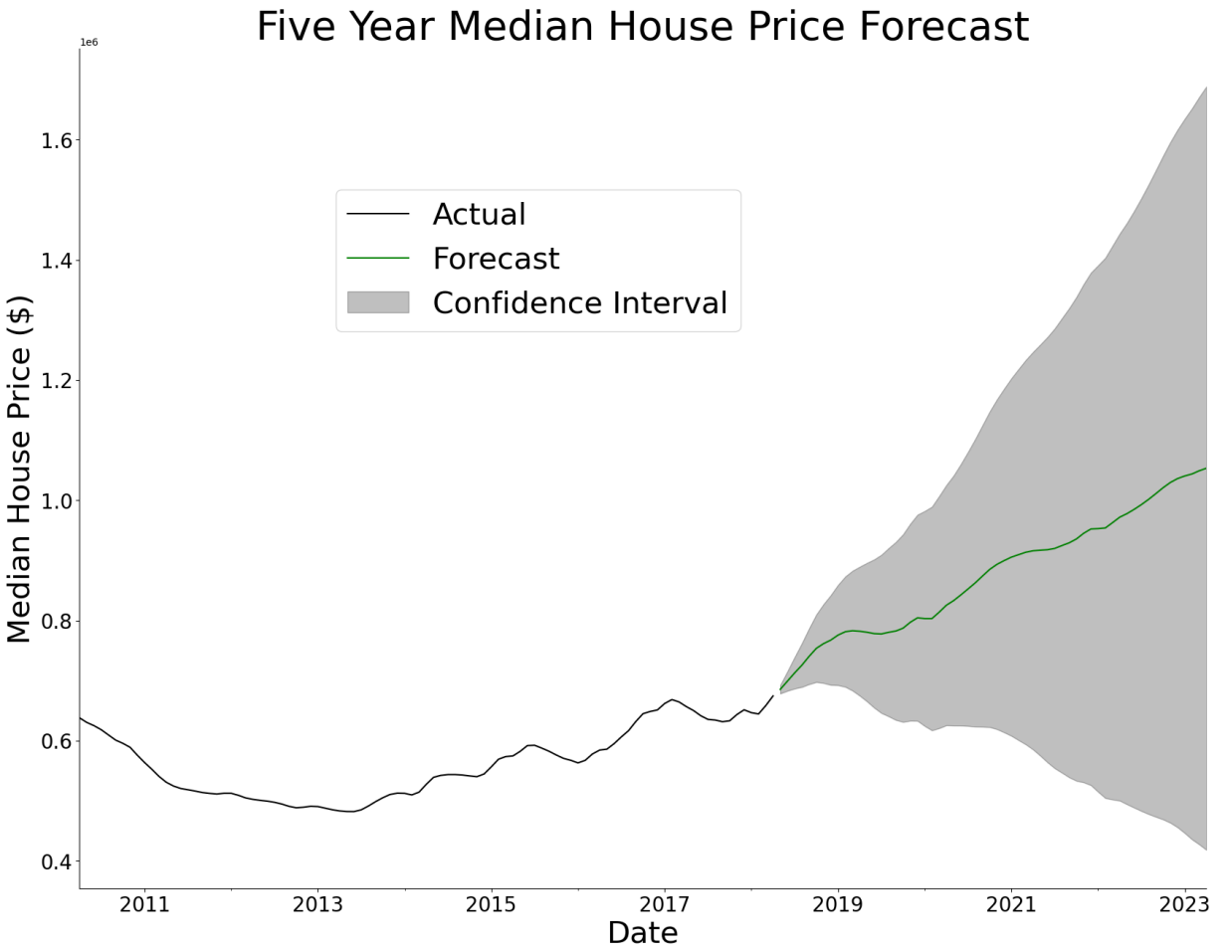

```
['87506']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7830      0.044     17.772      0.000       0.697       0.869
ma.L1          0.2943      0.021     14.125      0.000       0.253       0.335
ar.S.L12      -0.7339      0.050    -14.795      0.000      -0.831      -0.637
ma.S.L12      -0.0188      0.032     -0.590      0.555      -0.081       0.044
sigma2      1.292e+07   9.44e-10   1.37e+16      0.000    1.29e+07    1.29e+07
==============================================================================
```


Standardized residual for "8"
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017


Histogram plus estimated density
KDE
N(0,1)
Hist
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

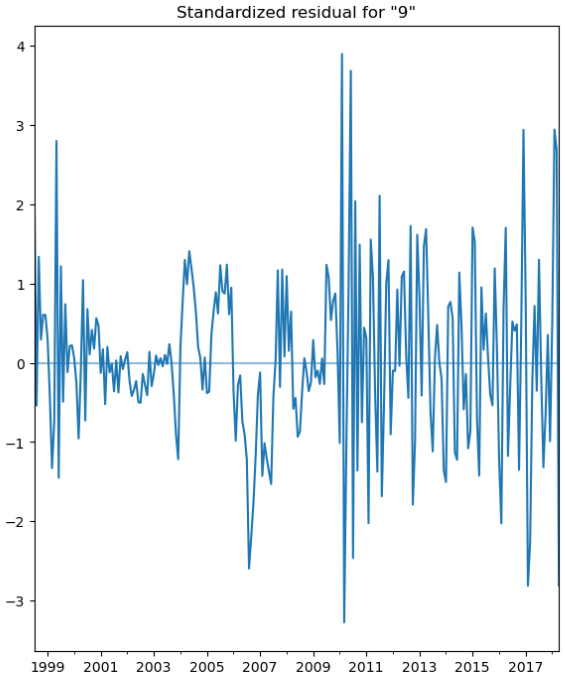

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

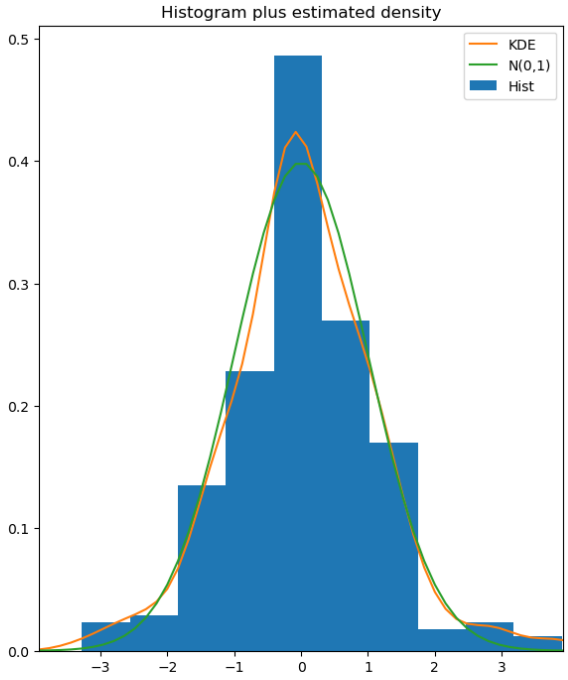

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

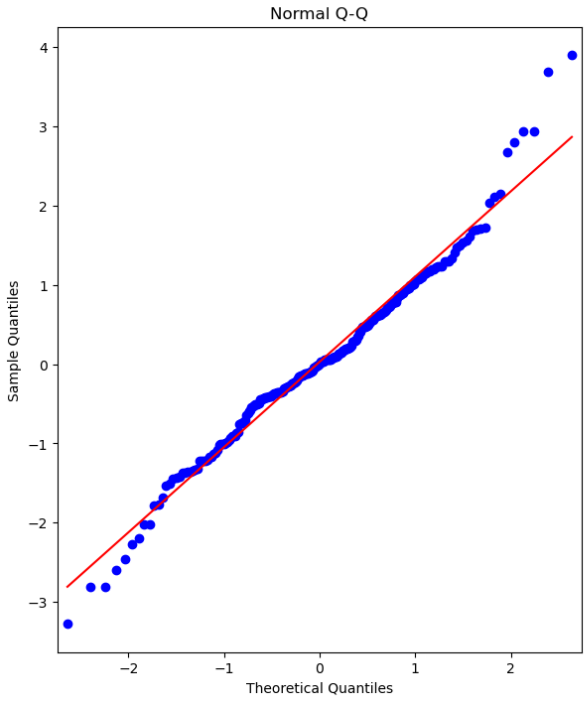


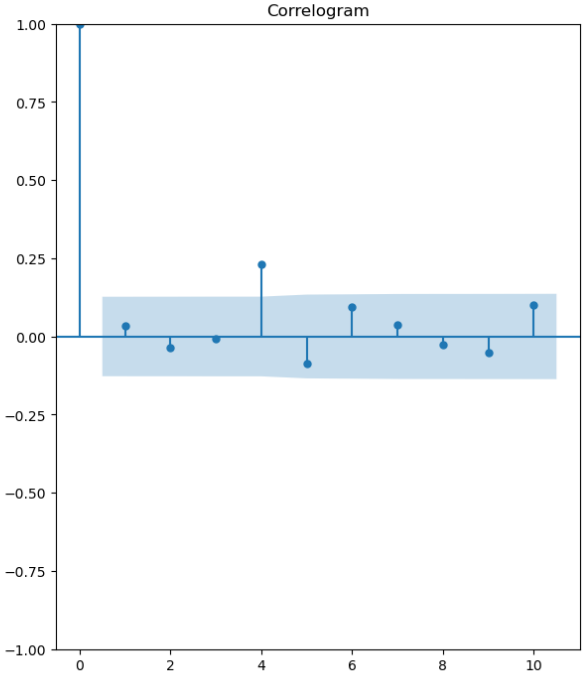

```
['97701']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
```

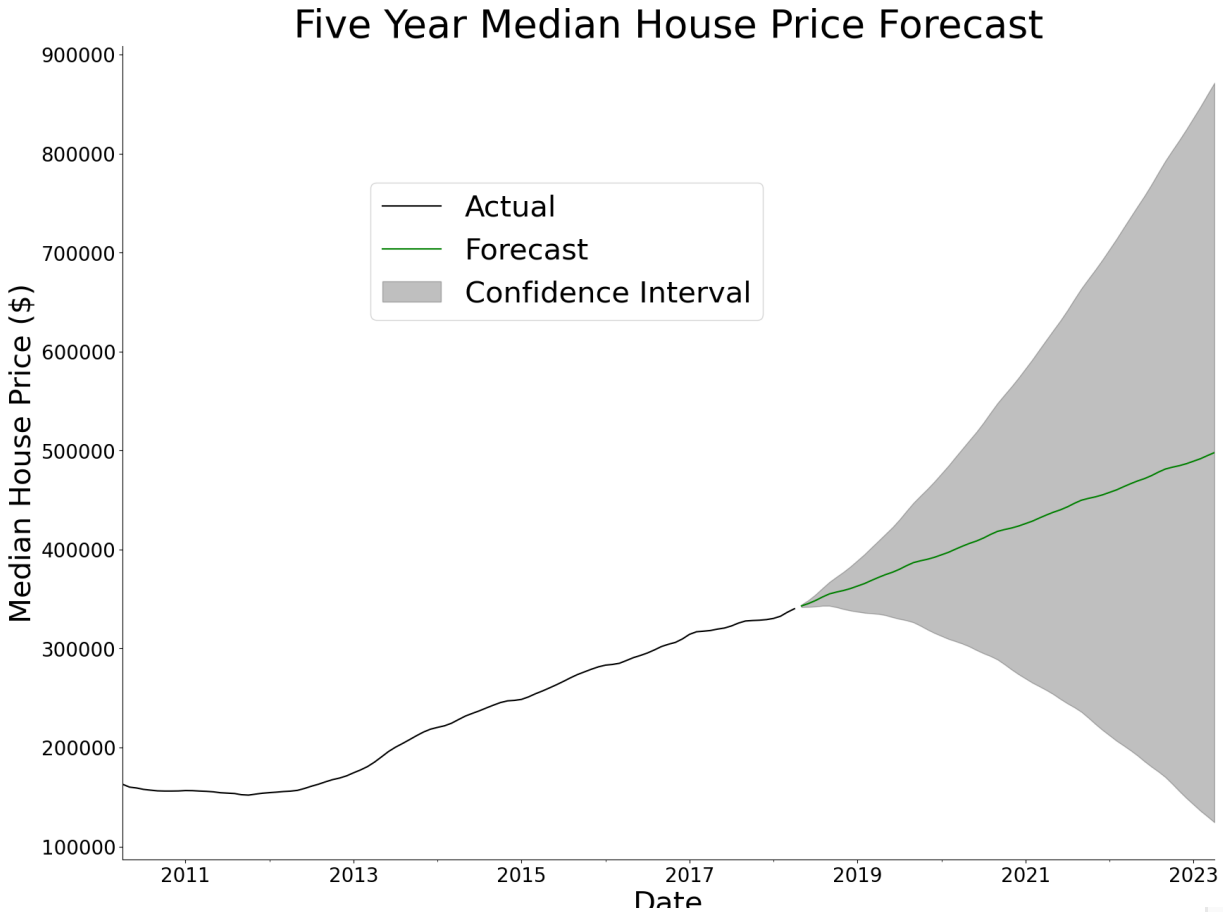
```
------------------------------------------------------------------------------
           coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9299      0.022     43.130      0.000       0.888
0.972
ma.L1          0.5517      0.034     16.282      0.000       0.485
0.618
ma.S.L12      -0.4288      0.030    -14.182      0.000      -0.488
-0.370
sigma2      4.249e+05   2.95e+04     14.381      0.000    3.67e+05
4.83e+05
==============================================================================
===========
```



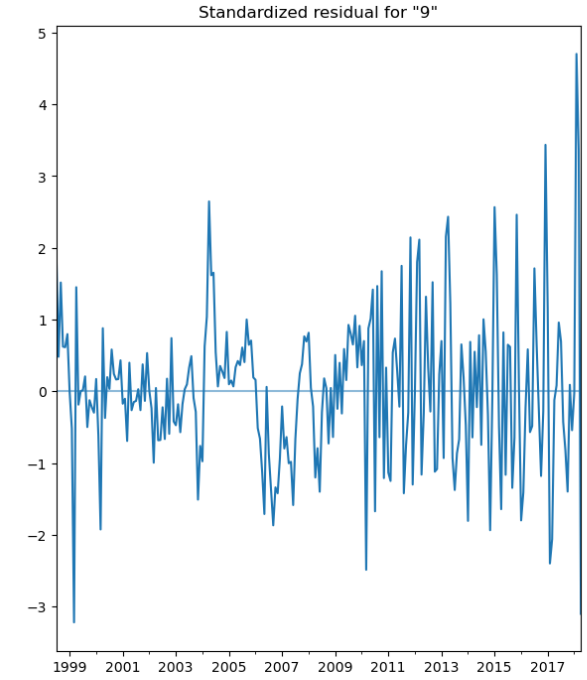


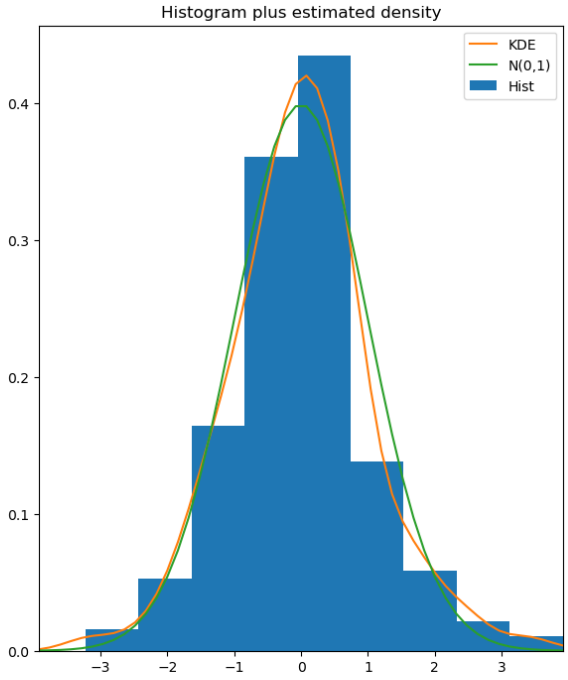


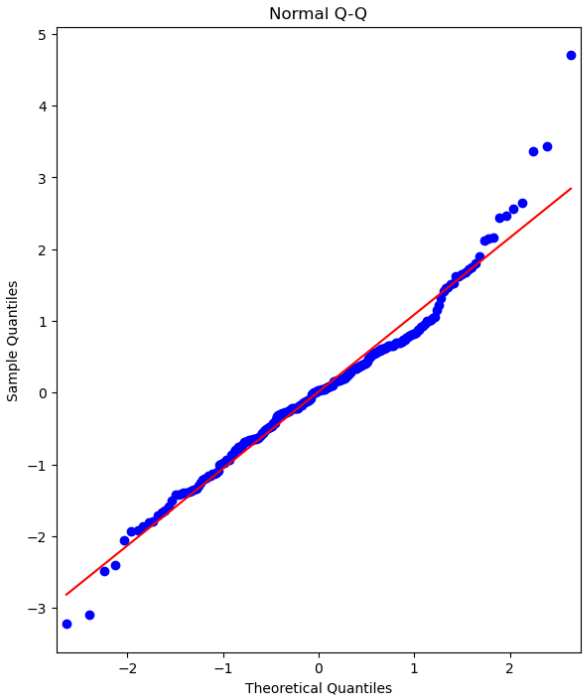


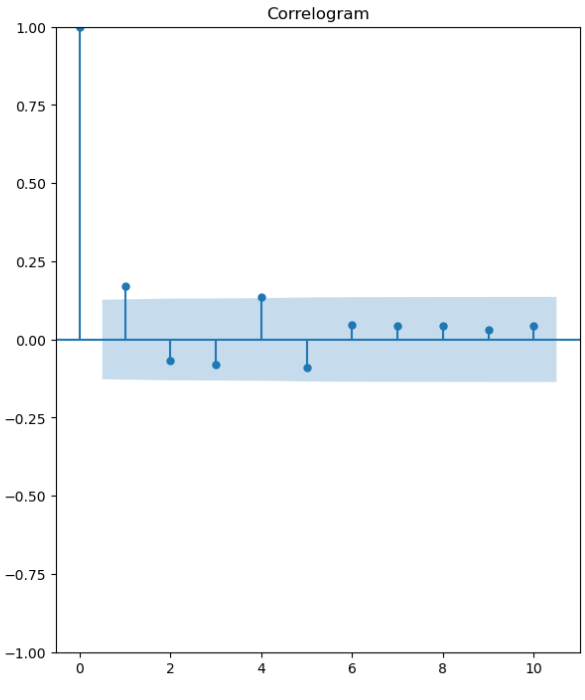


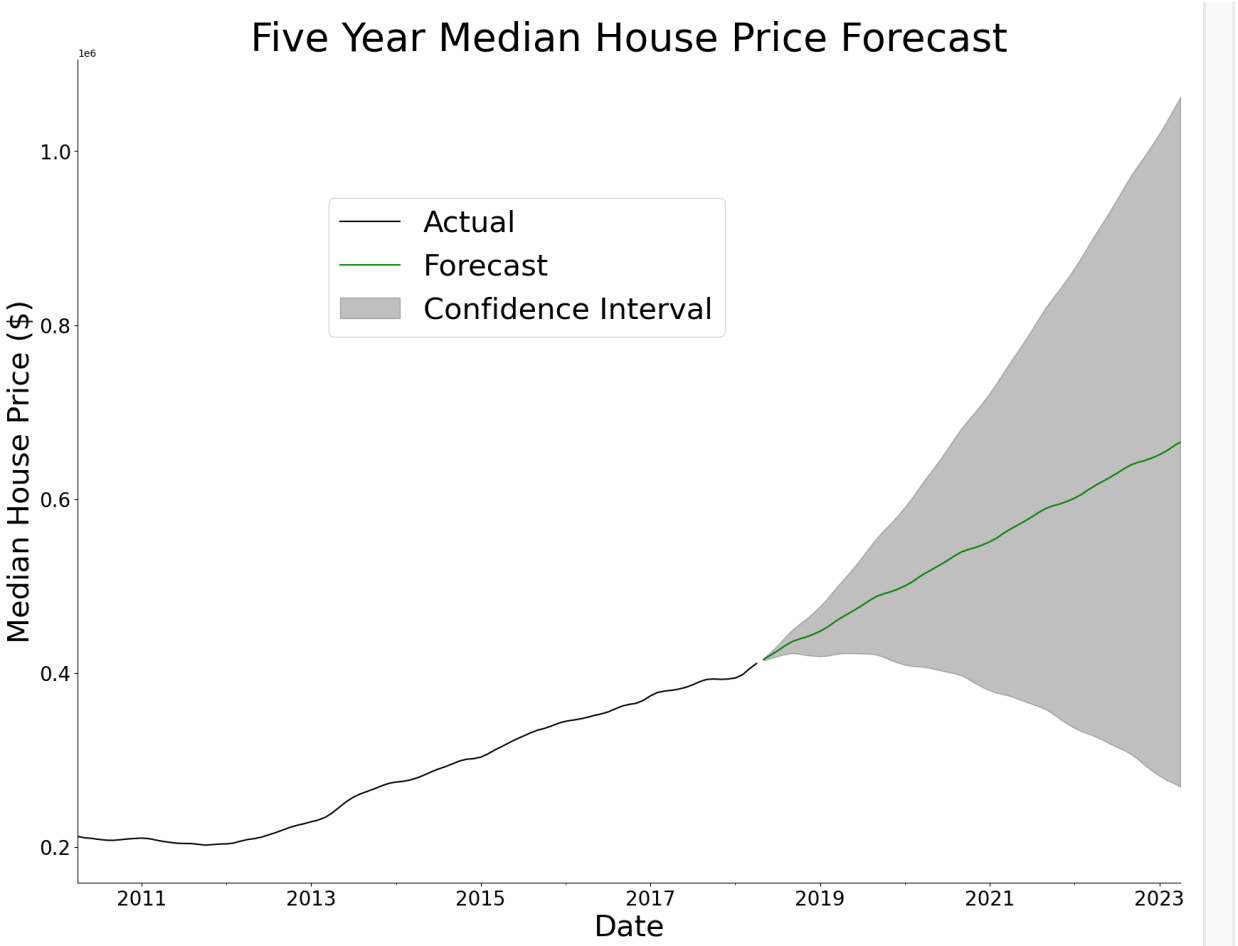

```
['97702']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9307      0.024     39.489      0.000       0.885      0.977
ma.L1          0.7650      0.039     19.513      0.000       0.688      0.842
ar.S.L12       0.0607      0.011      5.459      0.000       0.039      0.082
ma.S.L12      -0.5339      0.029    -18.252      0.000      -0.591     -0.477
sigma2      4.122e+05   2.93e+04     14.086      0.000    3.55e+05    4.7e+05
==============================================================================
```


Standardized residual for "9"
5
4
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

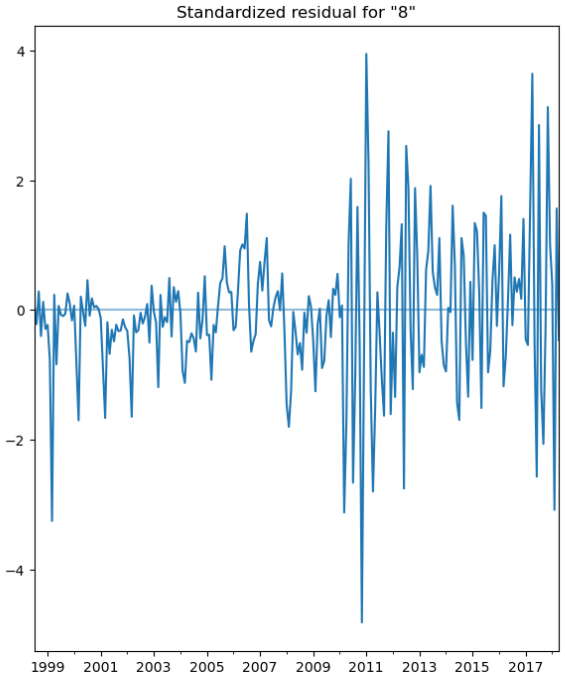

Histogram plus estimated density
KDE
N(0,1)
Hist
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

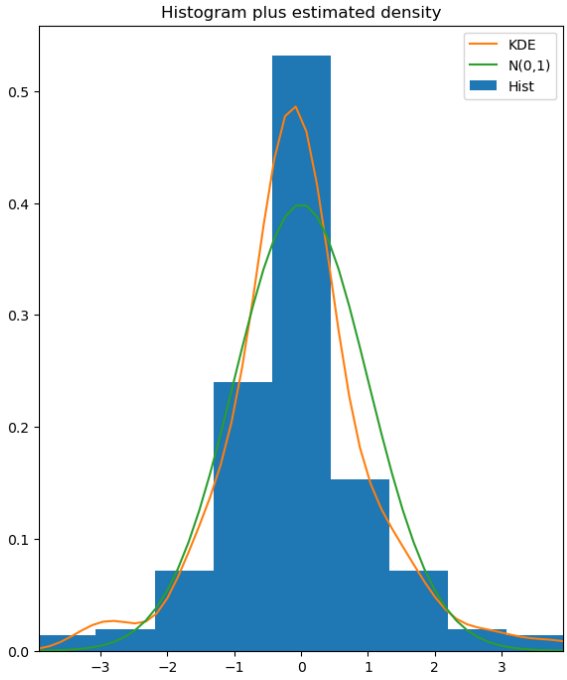

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
5
4
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

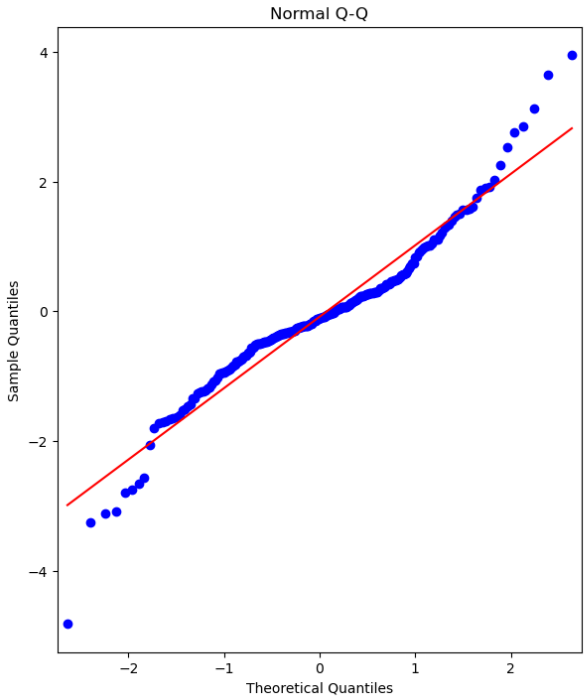

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

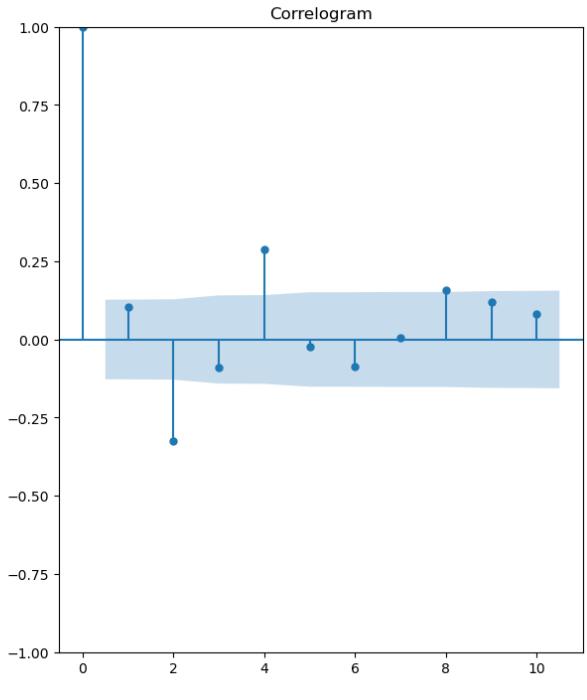

Five Year Median House Price Forecast
1e6
1.0
0.8
0.6
0.4
0.2
Median House Price ($)
Actual
Forecast
Confidence Interval
2011
2013
2015
2017
2019
2021
2023
Date

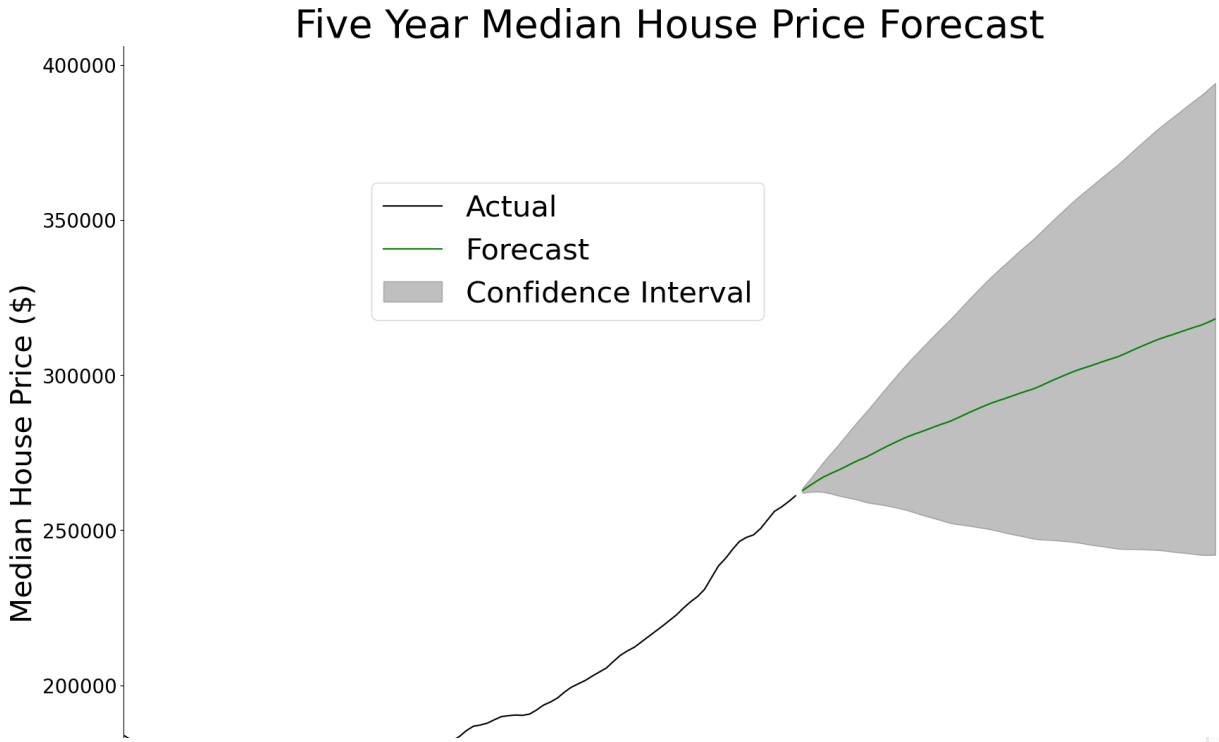
```
['84601']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8761      0.024     36.982      0.000       0.830
0.923
ma.L1          0.4131      0.036     11.379      0.000       0.342
0.484
ar.S.L12       0.0880      0.014      6.510      0.000       0.061
0.114
ma.S.L12      -0.9493      0.085    -11.121      0.000      -1.117
-0.782
sigma2      1.636e+05   1.29e+04     12.725      0.000    1.38e+05
1.89e+05
==============================================================================
```


Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

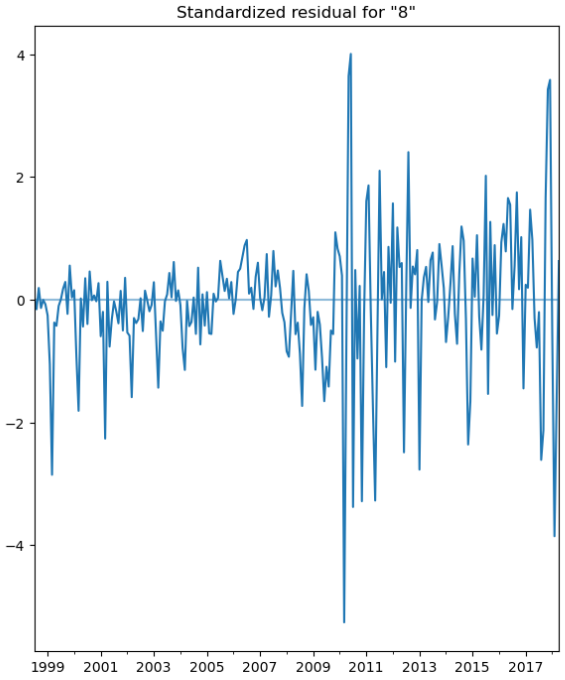

Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

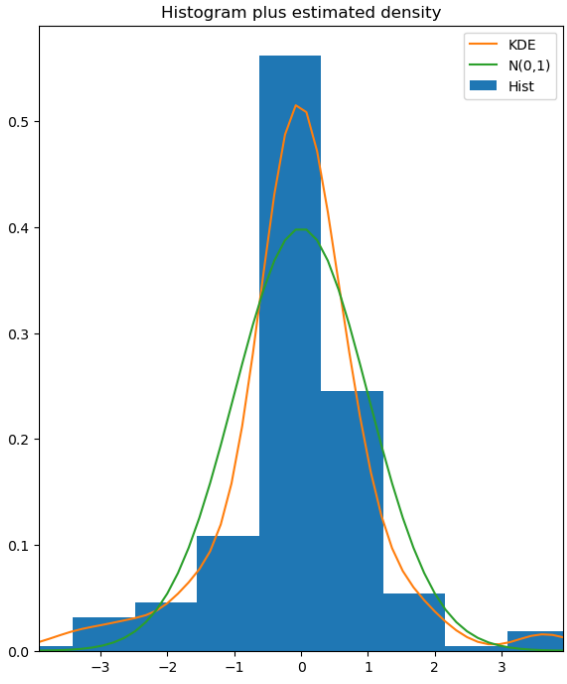

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2

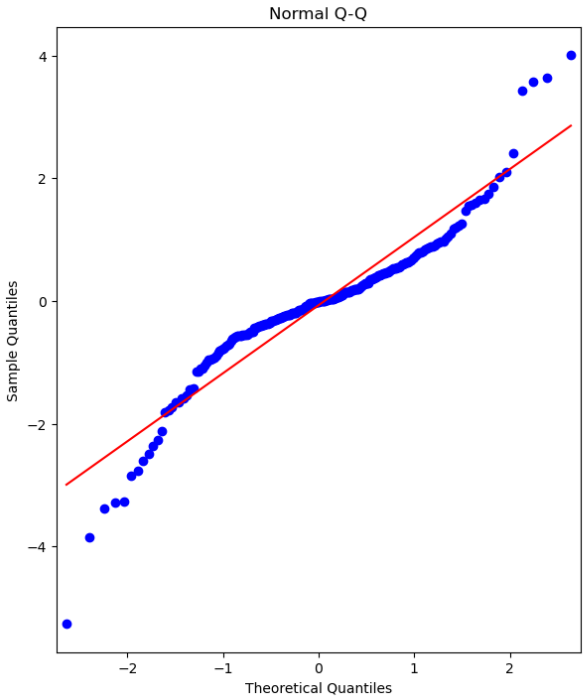

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

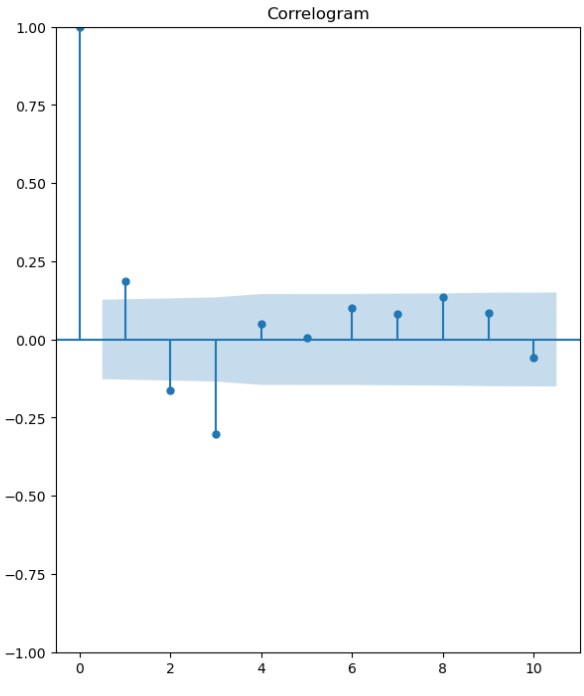


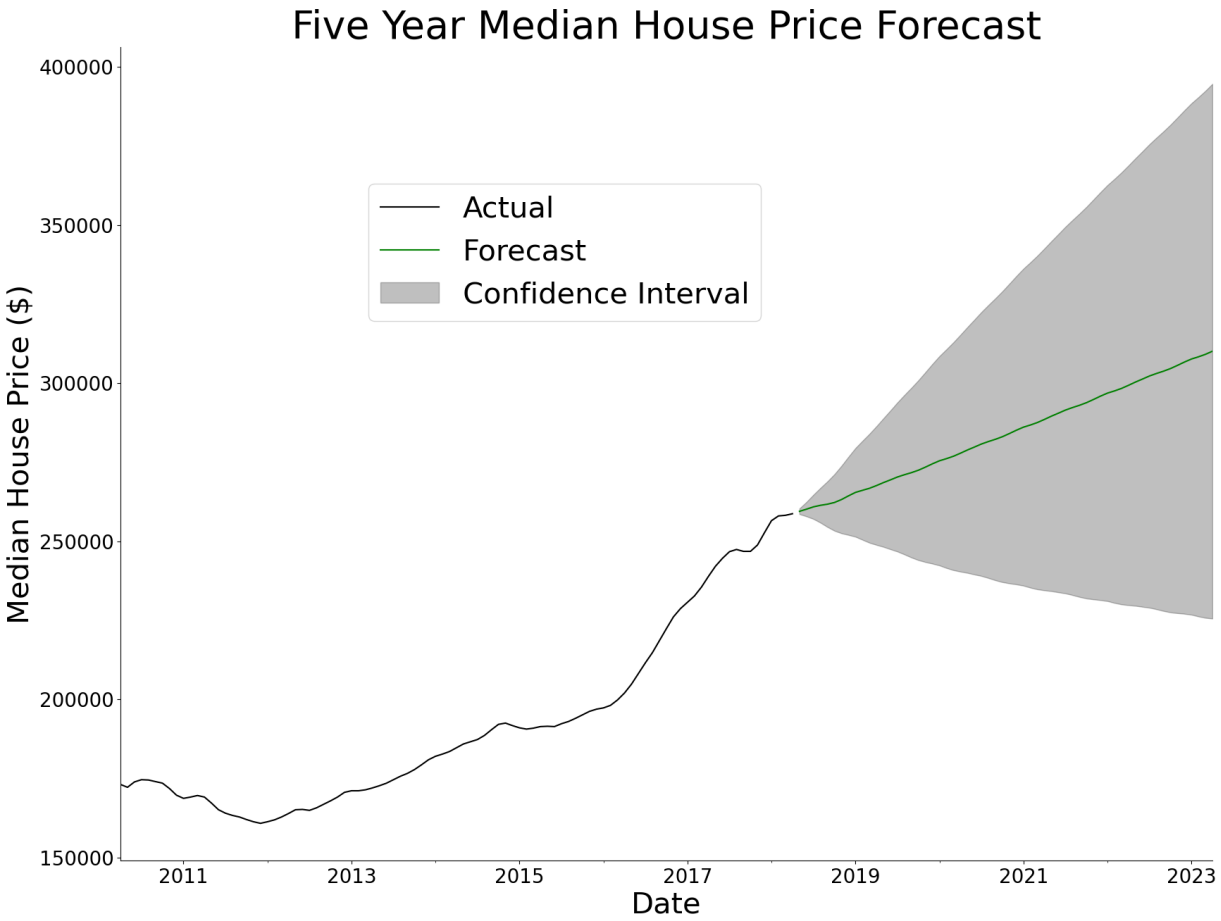

```
['84606']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8480      0.028     30.792      0.000       0.794       0.902
ma.L1          0.4739      0.041     11.471      0.000       0.393       0.555
ar.S.L12       0.0933      0.018      5.135      0.000       0.058       0.129
ma.S.L12      -0.8693      0.048    -18.100      0.000      -0.963      -0.775
sigma2      2.099e+05   1.09e+04     19.290      0.000    1.89e+05    2.31e+05
==============================================================================
```


Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017


Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

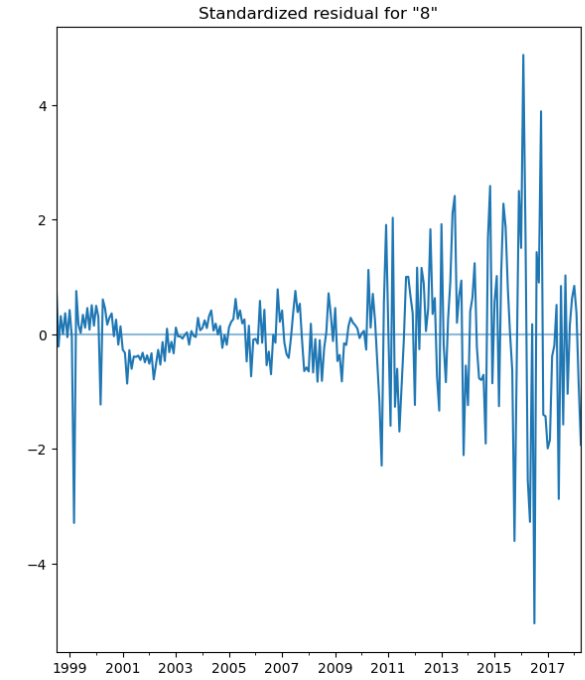

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2

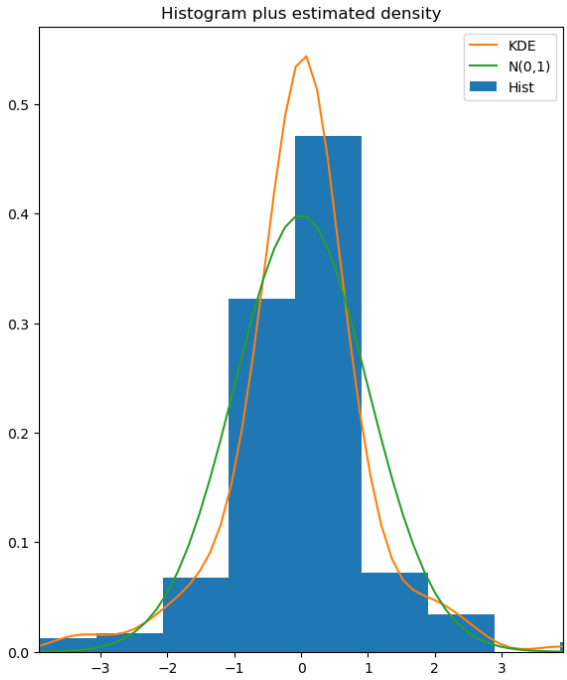

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

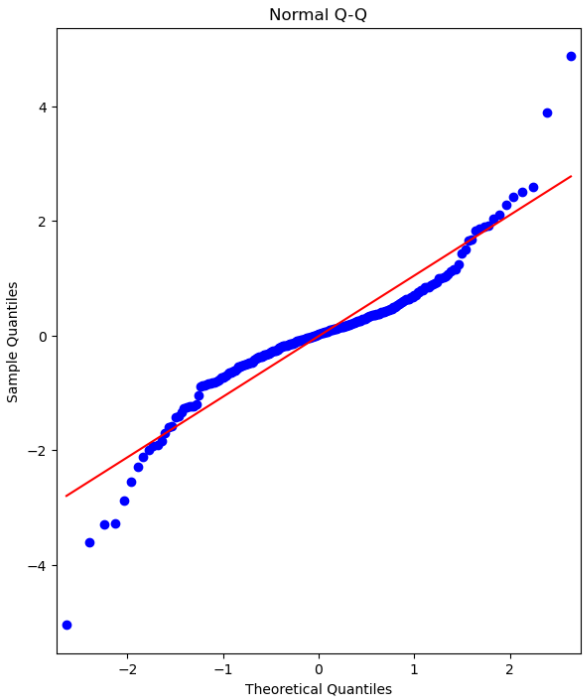


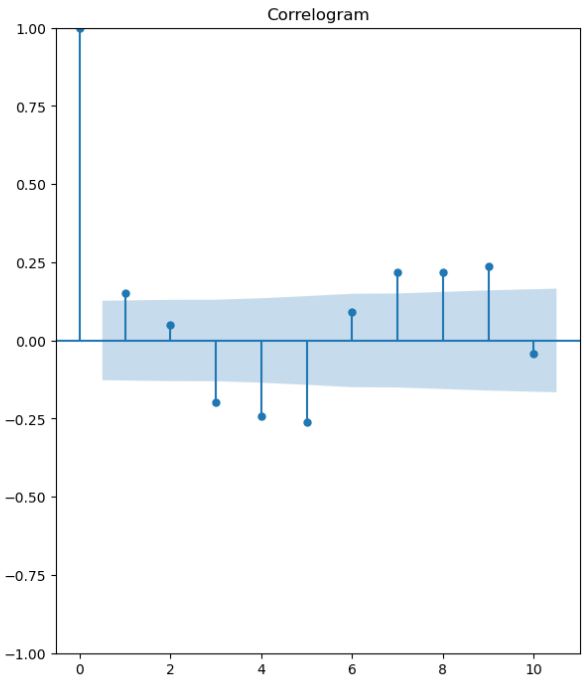

```
['80305']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.7820      0.029     26.897      0.000       0.725
0.839
ma.L1          0.7818      0.028     27.511      0.000       0.726
0.838
ar.S.L12       0.0834      0.023      3.676      0.000       0.039
```

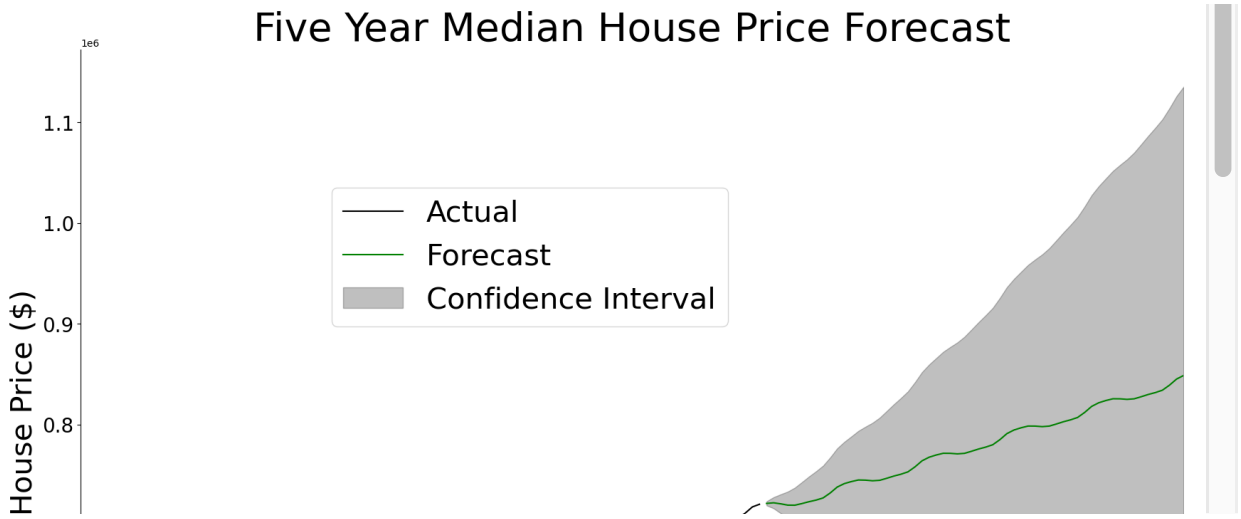
```
0.128
ma.S.L12       -0.4268       0.021    -20.510       0.000       -0.468
-0.386
sigma2      1.052e+06    5.02e+04      20.971       0.000     9.54e+05
1.15e+06
==============================================================================
===========
```



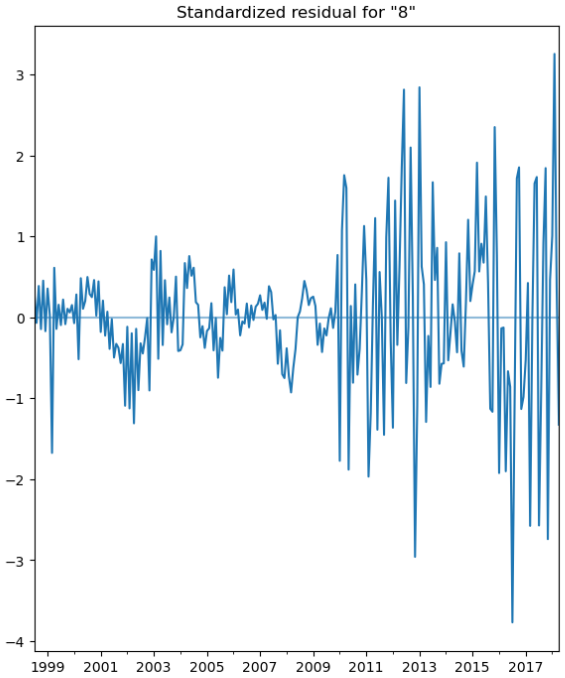


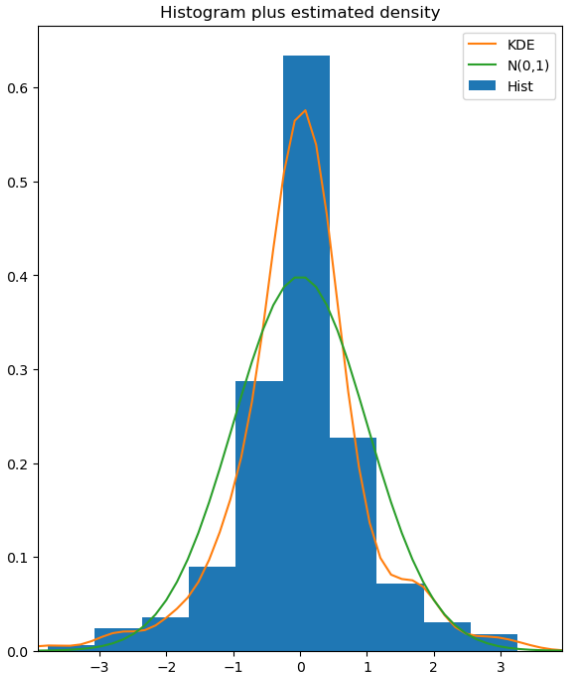


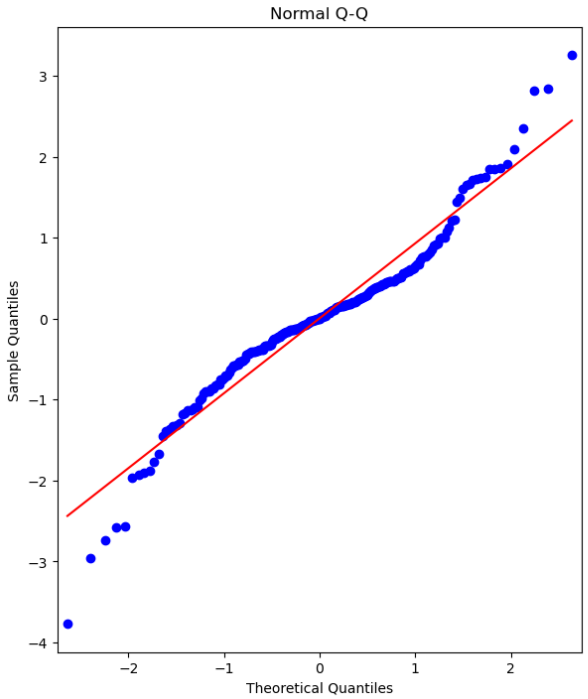


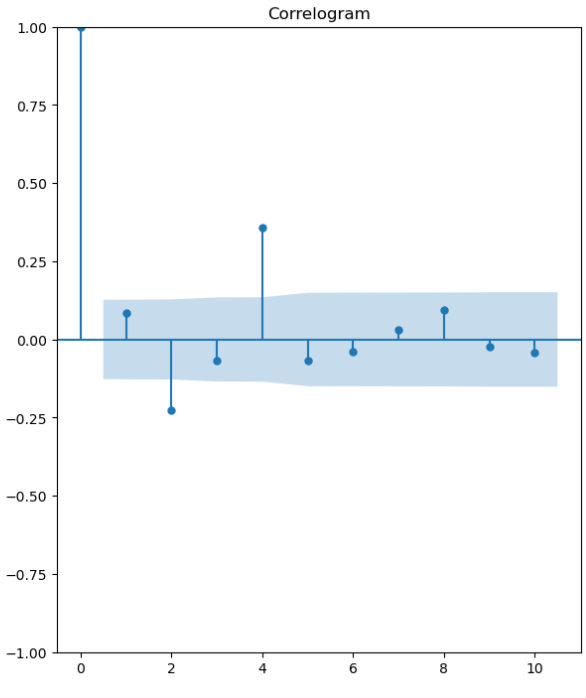


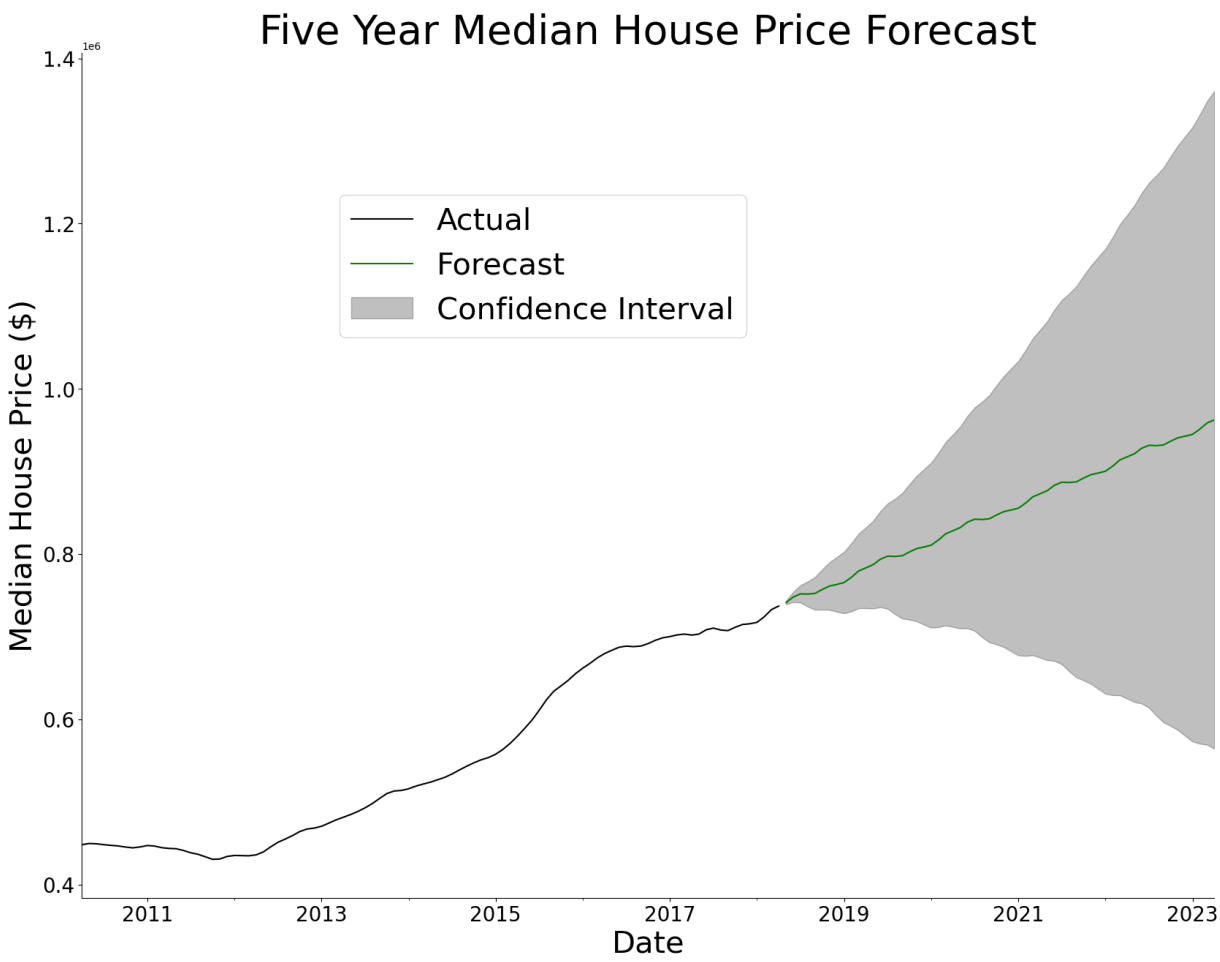

```
['80303']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8025      0.035     22.671      0.000       0.733
0.872
ma.L1          0.7827      0.046     16.927      0.000       0.692
0.873
ar.S.L12       0.0784      0.036      2.178      0.029       0.008
0.149
ma.S.L12      -0.2799      0.042     -6.647      0.000      -0.362
-0.197
sigma2      1.304e+06   9.64e+04     13.530      0.000    1.11e+06
1.49e+06
==============================================================================
```


Standardized residual for "8"
3
2
1
0
−1
−2
−3
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

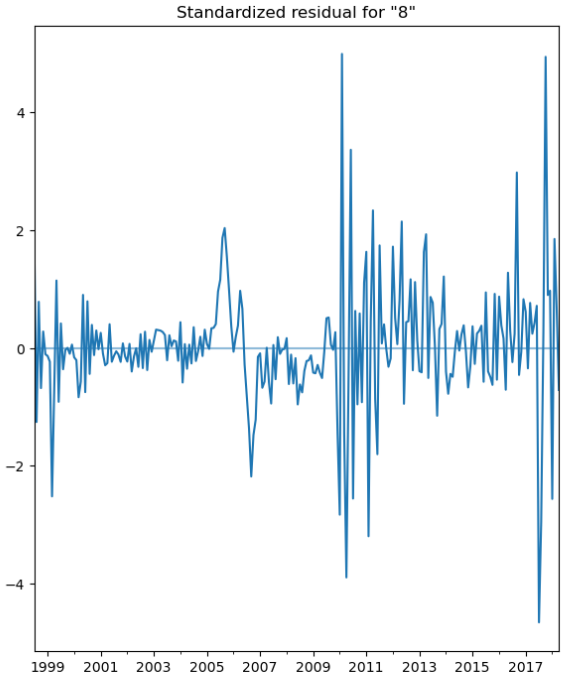

Histogram plus estimated density
KDE
N(0,1)
Hist
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

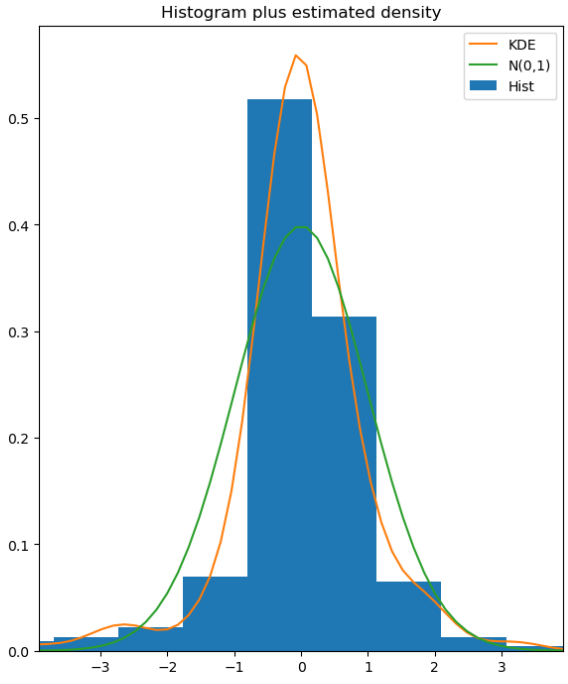

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
3
2
1
0
−1
−2
−3
−4
−2
−1
0
1
2

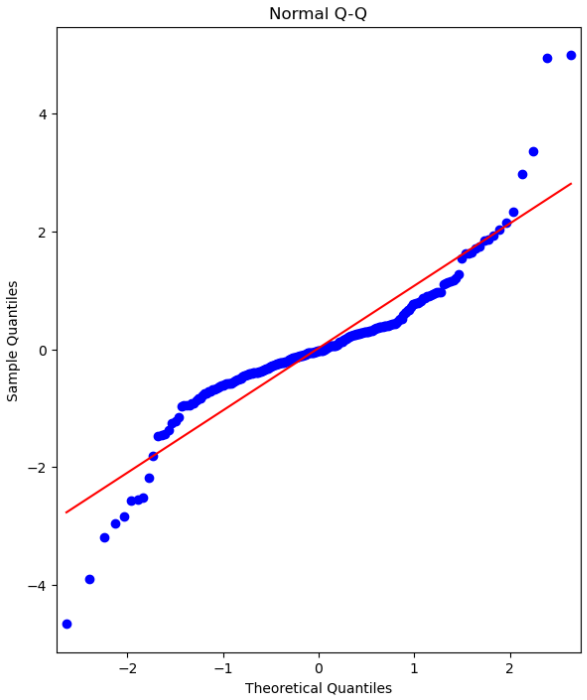

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

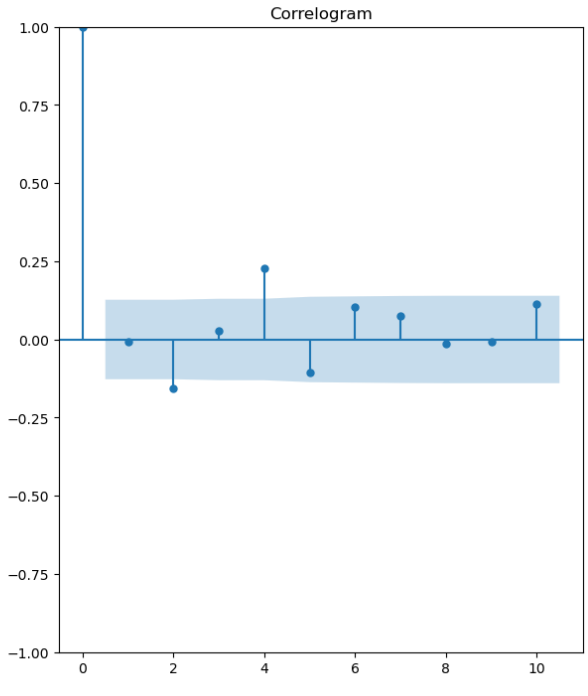


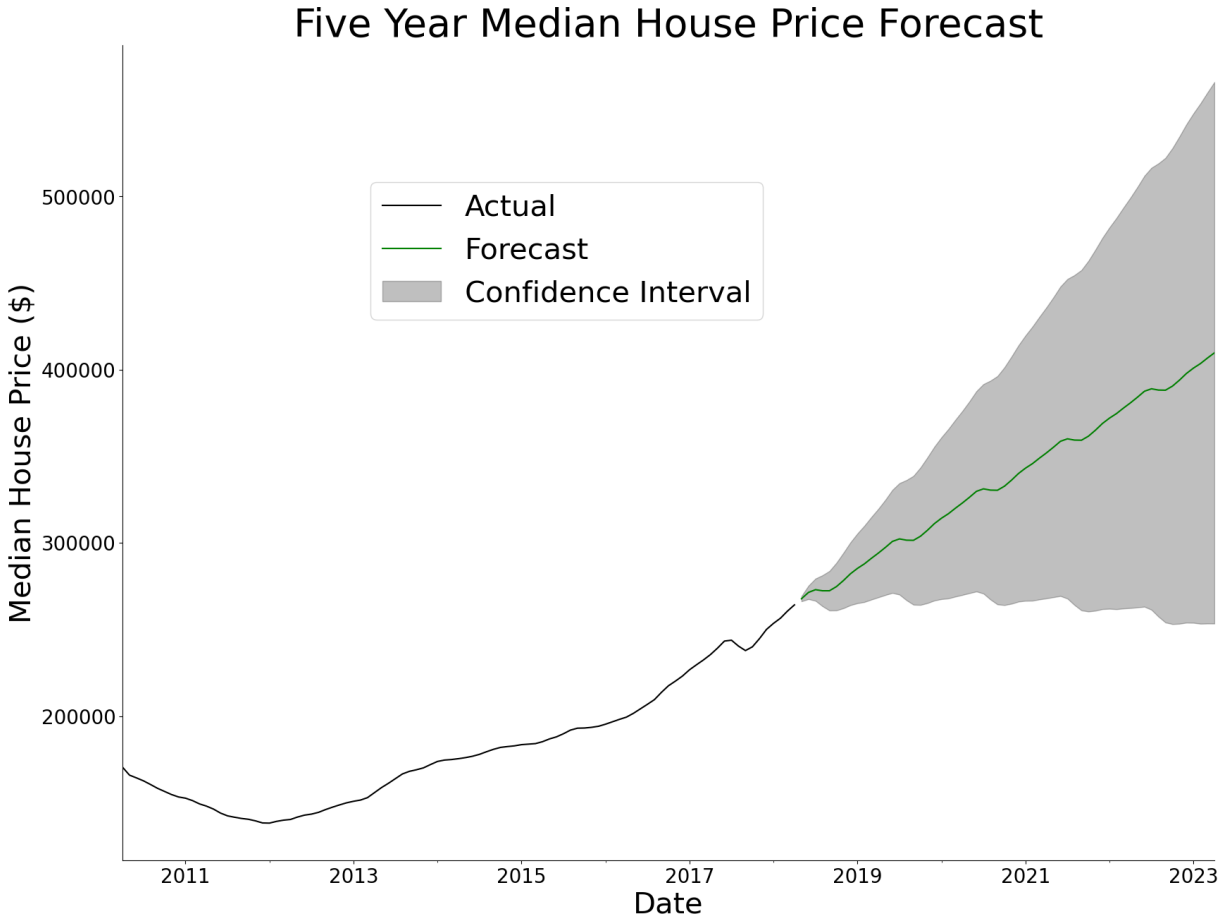

```
['83703']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7428      0.038     19.304      0.000       0.667       0.818
ma.L1          0.6268      0.033     18.860      0.000       0.562       0.692
ma.S.L12      -0.4391      0.032    -13.887      0.000      -0.501      -0.377
sigma2      5.895e+05   3.31e+04     17.792      0.000    5.25e+05    6.54e+05
==============================================================================
```


Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017


Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

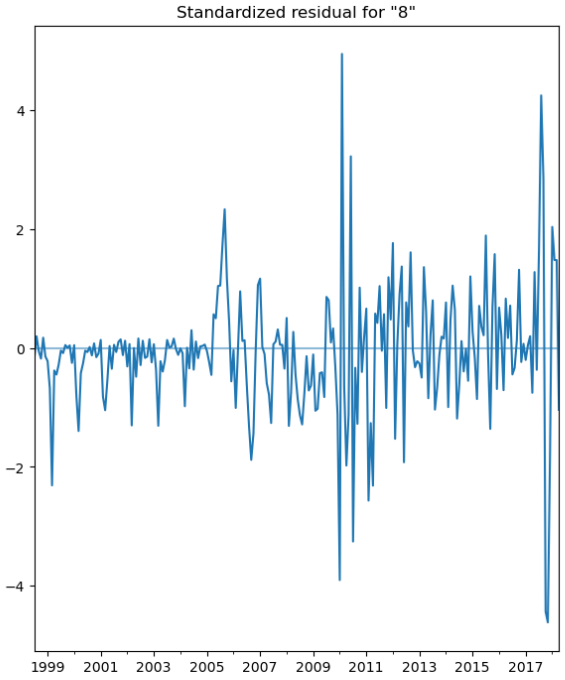

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2

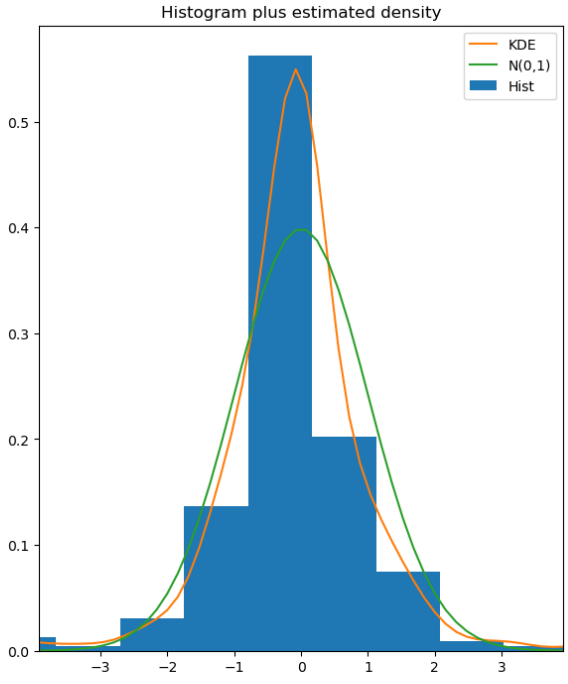

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

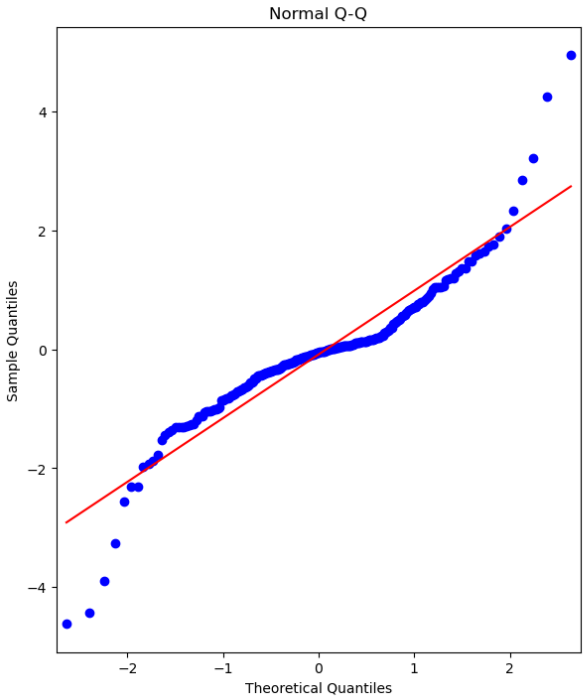


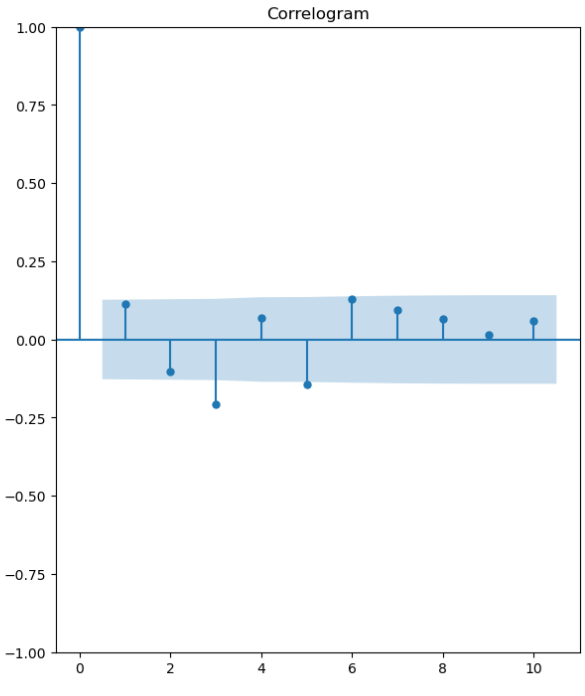

```
['83705']
(1, 1, 1) (1, 1, 1, 12)
```

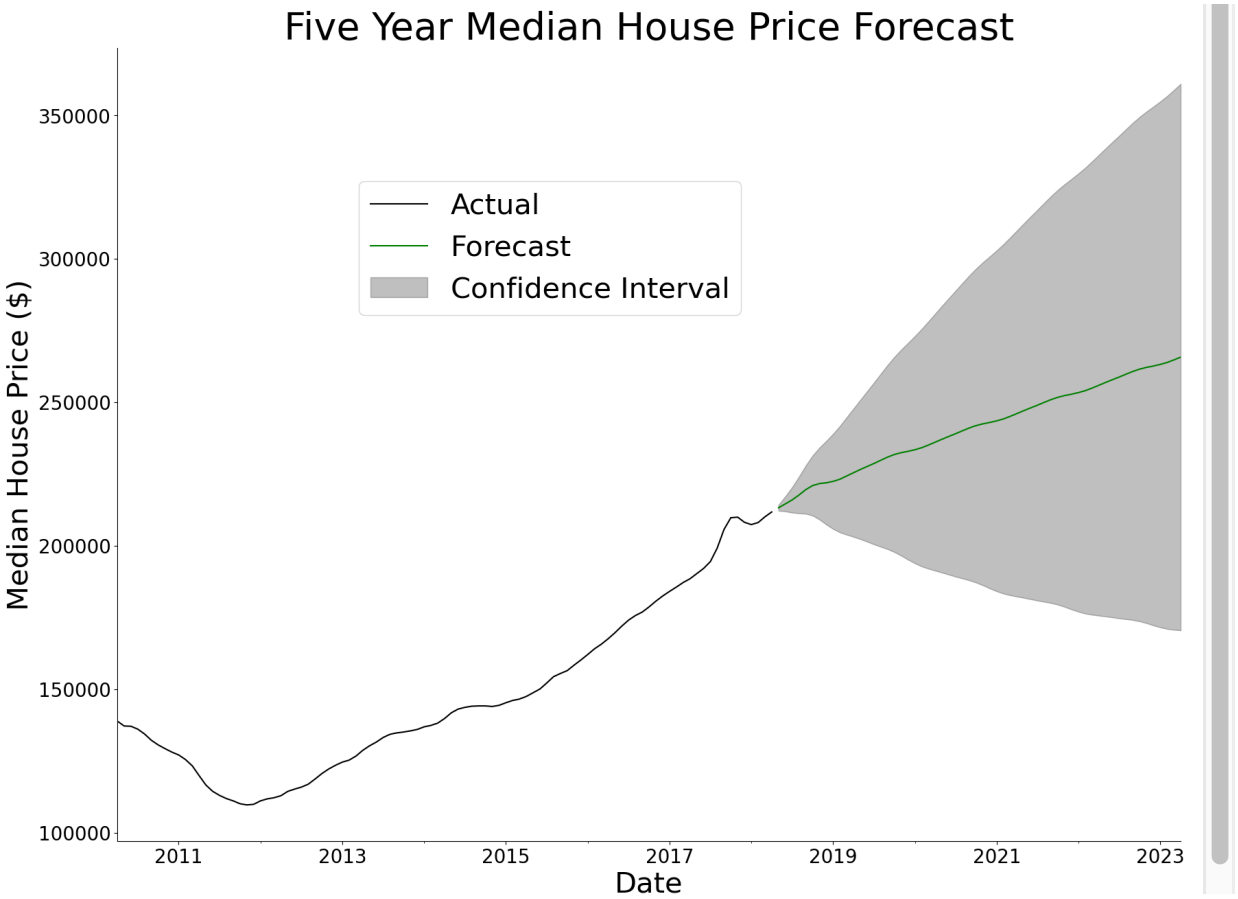
```
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8608      0.021     41.553      0.000       0.820       0.901
ma.L1          0.4534      0.029     15.615      0.000       0.396       0.510
ar.S.L12       0.1321      0.032      4.097      0.000       0.069       0.195
ma.S.L12      -0.9555      0.113     -8.433      0.000      -1.178      -0.733
sigma2      2.787e+05   2.93e+04      9.506      0.000    2.21e+05    3.36e+05
==============================================================================
```


Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017


Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3


Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2


Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10


Five Year Median House Price Forecast
Actual
Forecast
Confidence Interval
Median House Price ($)
350000
300000
250000
200000
150000
100000
2011
2013
2015
2017
2019
2021
2023
Date

In [28]:

```
# Create column for predicted price after 5 years
mountain_towns['5_Year_Price'] = pred_list_2

# Calculate Percent Return from time of investment
mountain_towns['Rate_Return'] = ((
    mountain_towns['5_Year_Price']-
    mountain_towns['2018-04'])/
    mountain_towns['2018-04'])*100

mountain_towns
```

executed in 62ms, finished 11:55:23 2022-05-03

Out[28]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05	1996-06
1776	87505	Santa Fe	NM	Santa Fe	Santa Fe	1777	188200.0	188400.0	188500.0
6810	87506	Santa Fe	NM	Santa Fe	Santa Fe	6811	227000.0	227400.0	227900.0
1446	97701	Bend	OR	Bend	Deschutes	1447	136300.0	136100.0	135900.0
706	97702	Bend	OR	Bend	Deschutes	707	160300.0	160300.0	160300.0
4081	84601	Provo	UT	Provo	Utah	4082	142700.0	140500.0	138400.0
4442	84606	Provo	UT	Provo	Utah	4443	142700.0	141400.0	140100.0
6115	80305	Boulder	CO	Boulder	Boulder	6116	187900.0	187700.0	187500.0
3363	80303	Boulder	CO	Boulder	Boulder	3364	231100.0	230300.0	229500.0
5764	83703	Boise	ID	Boise City	Ada	5765	107500.0	107500.0	107500.0
2962	83705	Boise	ID	Boise City	Ada	2963	83000.0	83000.0	83000.0

10 rows × 273 columns

In [29]:

```
rate = mountain_towns[['Rate_Return','City']]
rate
```

executed in 23ms, finished 11:55:23 2022-05-03

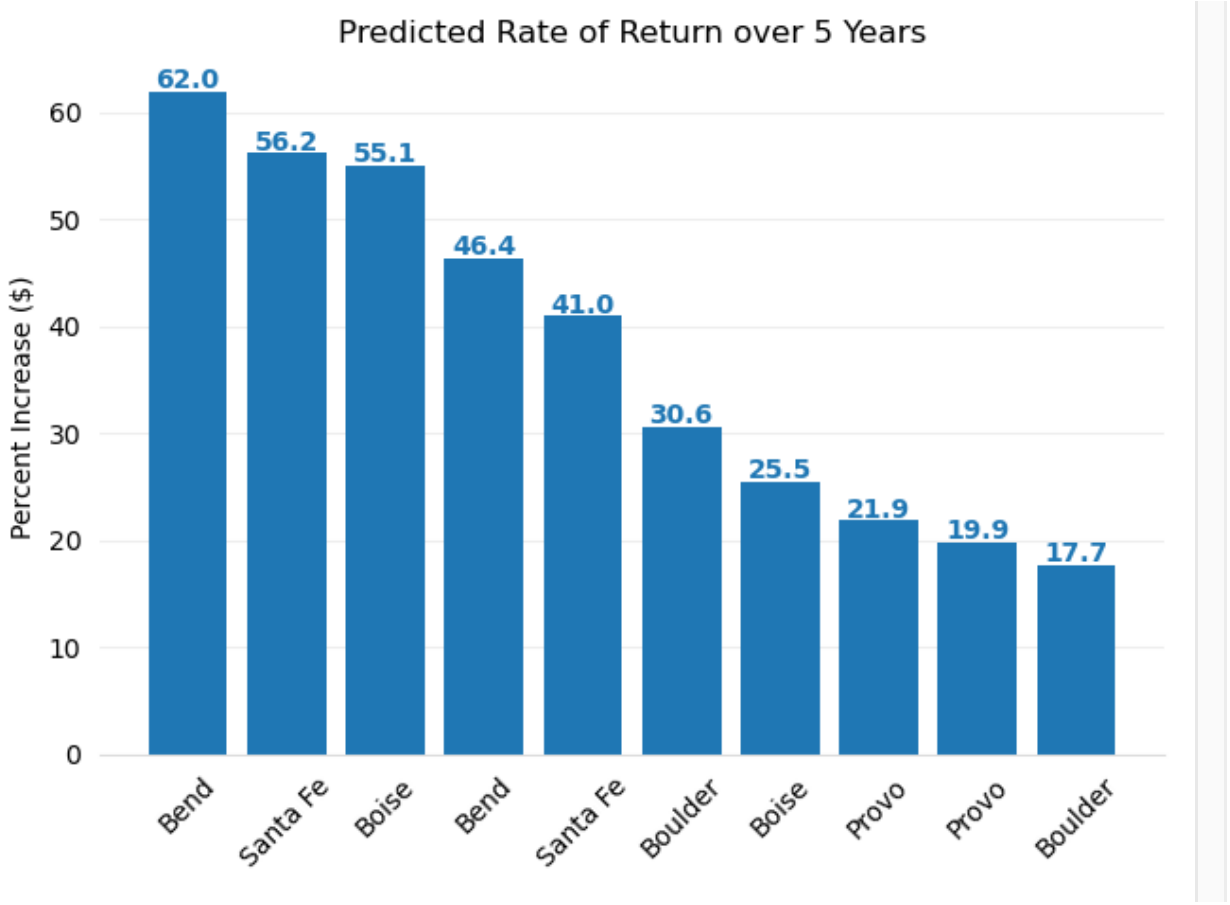
Out[29]:

	Rate_Return	City
1776	40.979317	Santa Fe
6810	56.185596	Santa Fe
1446	46.366086	Bend
706	61.959046	Bend
4081	21.852131	Provo
4442	19.863232	Provo
6115	17.734878	Boulder
3363	30.563084	Boulder
5764	55.076165	Boise
2962	25.482573	Boise

In [69]:

```
# create df and sort by Rate_Return in Descending Order
rate_2 = mountain_towns[['Rate_Return','City']]
#df.sort_values('importances',inplace=True)
rate_2.sort_values('Rate_Return',inplace=True,ascending=False)

fig, ax = plt.subplots()

# Save the chart so we can loop through the bars below.
bars = ax.bar(
    x=np.arange(rate_2['Rate_Return'].size),
    height=rate_2['Rate_Return'],
    tick_label=rate_2['City']
)

# Axis formatting.
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.spines['bottom'].set_color('#DDDDDD')
ax.tick_params(bottom=False, left=False)
ax.set_axisbelow(True)
ax.yaxis.grid(True, color='#EEEEEE')
ax.xaxis.grid(False)

# Grab the color of the bars so we can make the
# text the same color.
bar_color = bars[0].get_facecolor()

# Add text annotations to the top of the bars.
# Note, you'll have to adjust this slightly (the 0.3)
# with different data.
for bar in bars:
  ax.text(
      bar.get_x() + bar.get_width() / 2,
      bar.get_height() + 0.3,
      round(bar.get_height(), 1),
      horizontalalignment='center',
      color=bar_color,
      weight='bold'
  )

plt.xticks(rotation = 45)
plt.title('Predicted Rate of Return over 5 Years')
plt.ylabel('Percent Increase ($)')

fig.tight_layout()
```

executed in 459ms, finished 13:56:16 2022-05-03



5.1.2 Summary

- Due to the work-from-home movement following the covid pandemic, these zip codes are currently ~50% above the predicted price, which represents a rate of return well over 100% in most zip codes.

5.1.2.1 Provo, Utah 84601 and 84606

- Both of the Provo zip codes had good rates of return (21.9% and 19.9%) and had the least potential downside.

5.1.2.2 Santa Fe, New Mexico 87506

- This zip code had the highest rate of return (56.2%), but did have a greater potential downside than Provo, UT.

5.1.2.3 Boise, Idaho 83703

- Predicted increase of 55.1% with very little risk.
- Probably the best investment potential for mountain towns.

5.2 Mid-Size Cities

- Jacksonville, FL
- Birmingham, AL
- Kansas City, MO
- Oklahoma City, OK
- Salt Lake City, UT

In [30]:

```
midsize_list = []
midsize_list.append(['Salt Lake City', 'Oklahoma City', 'Jacksonville', 'Birmi

midsize_cities = pd.DataFrame()

mid = pd.DataFrame(midsize_list)
mid = mid.T

for i in mid[0]:
    # Create df from new city
    ok = df1[df1['City'] == str(i)]

    # add column with percent change between 2017 and 2012 median value
    ok['Percent Change'] = (ok['2018-04'] / ok['2012-04']) * 100

    # create new dataframe with top 10 zip codes for Percent Change
    ok = ok.sort_values('Percent Change',ascending = False)

    # new df_percent with only top ten results
    ok = ok.head(2)

    # concatenate to mountain towns df
    midsize_cities = pd.concat([ok,midsize_cities])

midsize_cities = midsize_cities.drop(['RegionID','Percent Change'], axis=1)

midsize_cities
```

executed in 204ms, finished 11:55:24 2022-05-03

Out[30]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05
4121	66102	Kansas City	KS	Kansas City	Wyandotte	4122	38800.0	38800.0
4293	66104	Kansas City	KS	Kansas City	Wyandotte	4294	41300.0	41200.0
4502	48009	Birmingham	MI	Detroit	Oakland	4503	242700.0	242600.0
8459	35222	Birmingham	AL	Birmingham	Jefferson	8460	103400.0	104200.0
8259	32204	Jacksonville	FL	Jacksonville	Duval	8260	41400.0	41400.0
2596	32211	Jacksonville	FL	Jacksonville	Duval	2597	62000.0	61700.0
9540	73103	Oklahoma City	OK	Oklahoma City	Oklahoma	9541	46800.0	47000.0
5702	73114	Oklahoma City	OK	Oklahoma City	Oklahoma	5703	NaN	NaN
9058	84101	Salt Lake City	UT	Salt Lake City	Salt Lake	9059	85100.0	85400.0

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05
6557	84111	Salt Lake City	UT	Salt Lake City	Salt Lake	6558	116700.0	116900.0

10 rows × 271 columns

5.2.1 Predicted Values

In [31]:

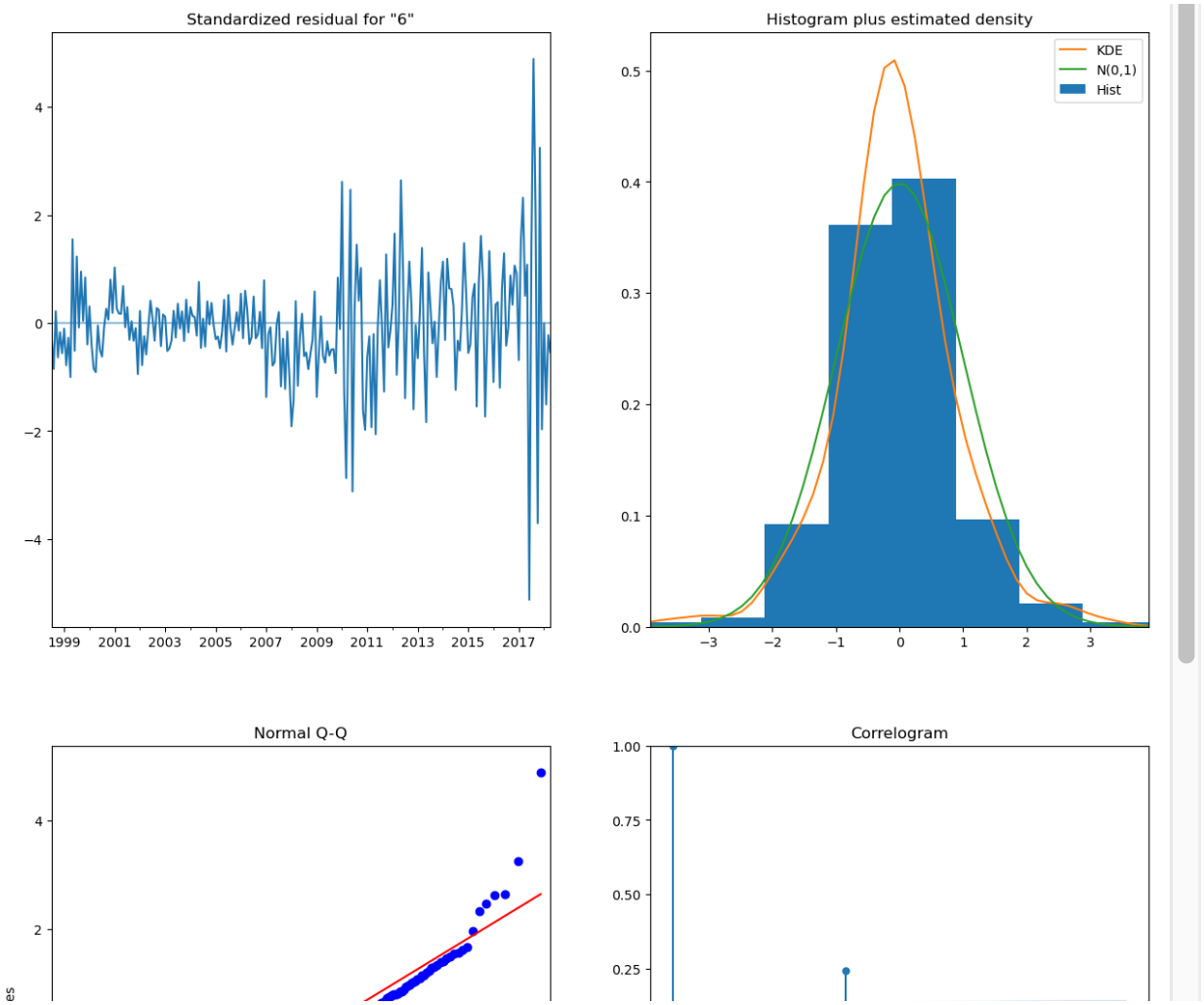
```
# Create list for final predicted value
pred_list_3 = []

# Construct model from entirety of data - no train/test
# iterate over prospects to extract optimal SARIMA values,
# instantiate SARIMAX model, make predictions
# and produce forecast with confidence intervals
for zipcode in midsize_cities['ZipCode']:
    zips = midsize_cities.loc[midsize_cities['ZipCode'] == zipcode]
    m1 = melt_data(zips)
    m1[str(zipcode)] = m1['value']
    m1 = m1.drop('value', axis=1)

    # do NOT create train and test sets
    #train = m1.iloc[:238]
    #test = m1.iloc[238:]

    # Run a grid with pdq and seasonal pdq parameters calculated above and ge
    ans = []
    outputs = []
    comb_list = []
    combs_list = []

    for comb in pdq:
        for combs in pdqs:
            try:
                mod = sm.tsa.statespace.SARIMAX(m1,
                                                order=comb,
                                                seasonal_order=combs,
                                                enforce_stationarity=False,
                                                enforce_invertibility=False)

                output = mod.fit()
                ans.append([comb, combs, output.aic])
                outputs.append([output.aic])
                comb_list.append([comb])
                combs_list.append([combs])
                AIC_df = pd.DataFrame(list(zip(outputs, comb_list, combs_list
                #print('ARIMA {} x {}12 : AIC Calculated ={}'.format(comb, co
            except:
                continue

    # Print the lowest AIC and it's parameters for SARIMA
    AIC_df[0] = AIC_df[0].str.get(0)
    best_AIC = AIC_df.copy()
    lowest_AIC = min(best_AIC[0])
    lowest_AIC_row = AIC_df.loc[AIC_df[0] == lowest_AIC]
    #print(lowest_AIC_row)

    # call value for comb and combs for model input
    lowest_AIC_row[1] = lowest_AIC_row[1].str.get(0)
    lowest_AIC_row[2] = lowest_AIC_row[2].str.get(0)
```

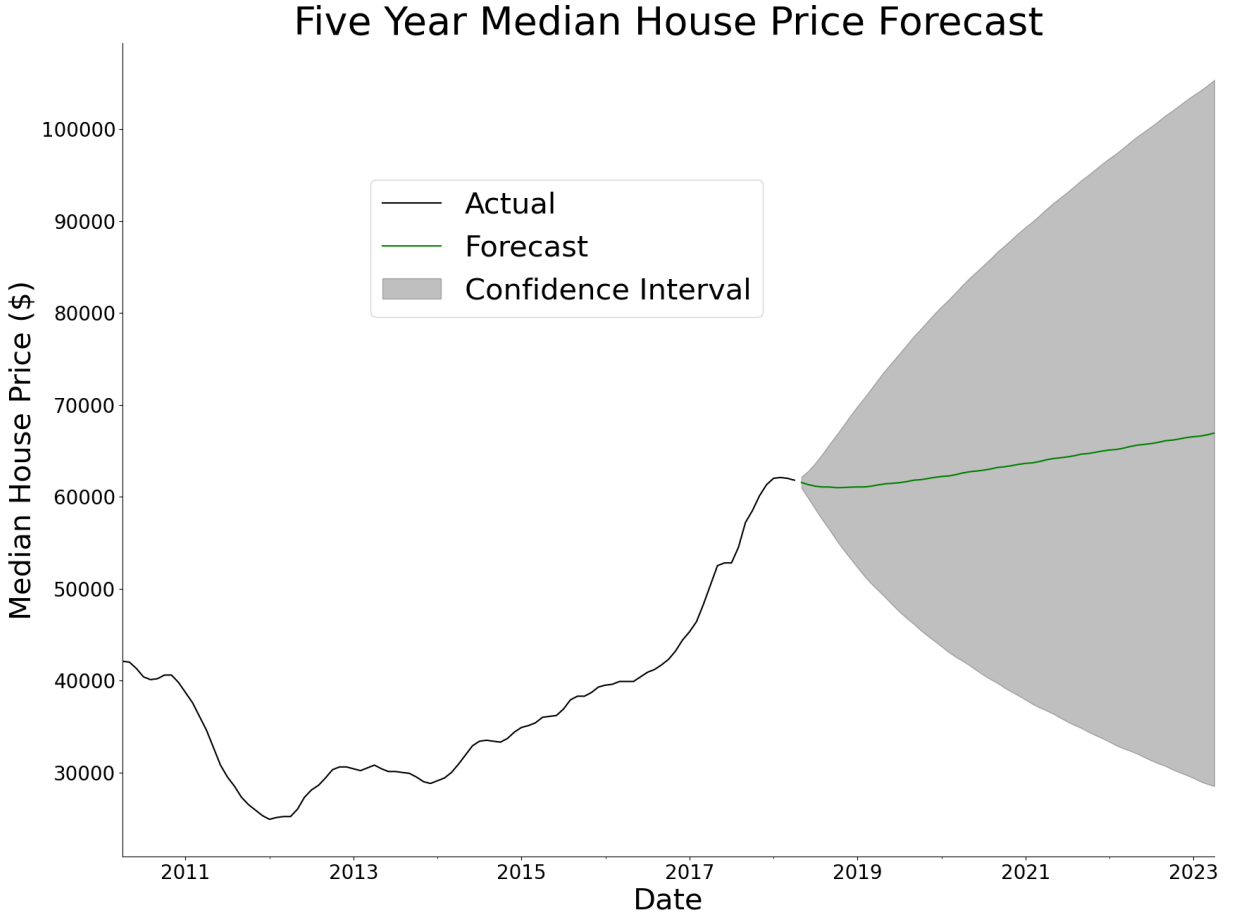
```
    arima = lowest_AIC_row.iat[0,1]
    s = lowest_AIC_row.iat[0,2]
    print(str(m1.columns.values))
    print(arima, s)

    # Plug the optimal parameter values into a new SARIMAX model
    ARIMA_MODEL = sm.tsa.statespace.SARIMAX(m1,
                                            order=arima,
                                            seasonal_order=s,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)

    # Fit the model and print results
    output = ARIMA_MODEL.fit()

    print(output.summary().tables[1])

    # Call plot_diagnostics() on the results calculated above
    output.plot_diagnostics(figsize=(15, 18))
    plt.show()

    # Get actual results in dataframe
    price_truth = m1['2018-04-01':]

    # Get forecast 500 steps ahead in future
    prediction = output.get_forecast(steps=60)

    # Create predictions df
    pred = prediction.predicted_mean

    # Add prediction for 5 year forecasted value to list
    pred_list_3.append(pred['2023-04-01'])

    # Get confidence intervals of forecasts
    pred_conf = prediction.conf_int()

    # Only use data from April 2010 to improve visual of forecast
    plot_vals = m1.loc['2010-04-01':]

    # Plot future predictions with confidence intervals
    ax = plot_vals.plot(label='observed', figsize=(20, 15), color='black')
    prediction.predicted_mean.plot(ax=ax, label='Forecast', color='green')
    ax.fill_between(pred_conf.index,
                    pred_conf.iloc[:, 0],
                    pred_conf.iloc[:, 1], color='k', alpha=0.25)
    ax.set_xlabel('Date').set_fontsize(30)
    ax.set_ylabel('Median House Price ($)').set_fontsize(30)
    ax.set_title('Five Year Median House Price Forecast').set_fontsize(40)

    for item in (ax.get_xticklabels() + ax.get_yticklabels()):
        item.set_fontsize(20)

    right_side = ax.spines["right"]
    right_side.set_visible(False)
```

```
107
108     top_side = ax.spines["top"]
109     top_side.set_visible(False)
110
111     mylabels = 'Actual', 'Forecast', 'Confidence Interval'
112     plt.legend(fontsize=30,labels=mylabels,bbox_to_anchor=(0.6,0.85))
113     plt.show()
114     print()
115     print()
```

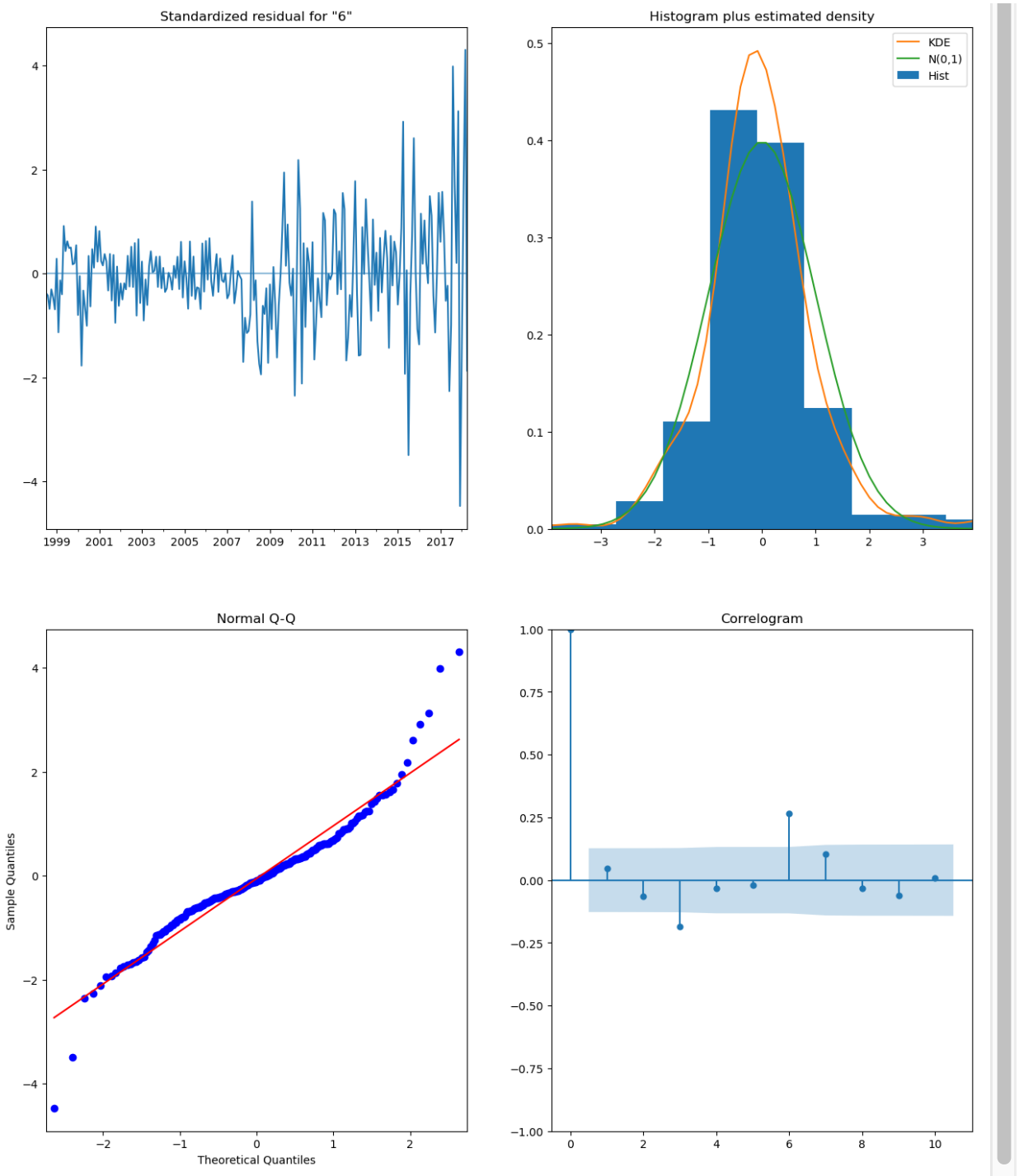
executed in 4m 5s, finished 11:59:28 2022-05-03

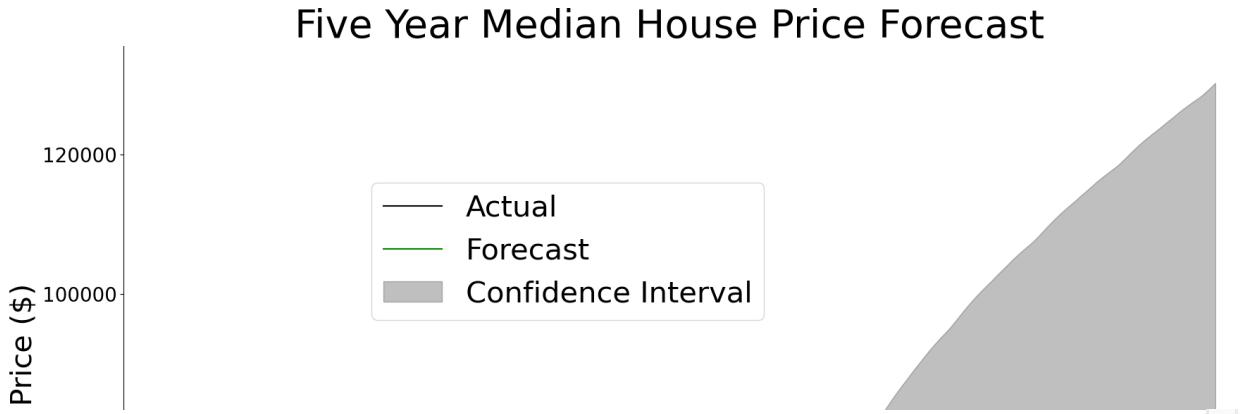
```
['66102']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8185      0.029     28.179      0.000       0.762
0.875
ma.L1          0.4528      0.035     12.856      0.000       0.384
0.522
ma.S.L12      -0.9918      0.559     -1.775      0.076      -2.087
0.103
sigma2      9.407e+04   4.85e+04      1.939      0.052   -1012.390
1.89e+05
==============================================================================
```


Standardized residual for "6"
Histogram plus estimated density
KDE
N(0,1)
Hist
Normal Q-Q
Correlogram

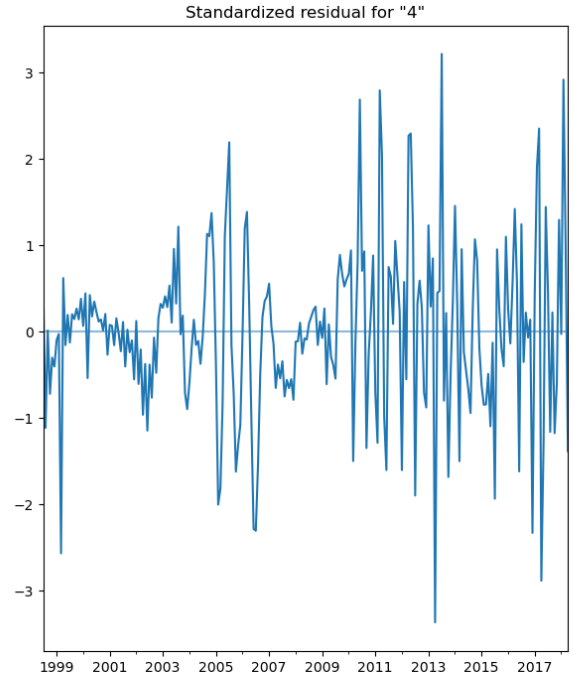

Five Year Median House Price Forecast
Actual
Forecast
Confidence Interval
Median House Price ($)
Date

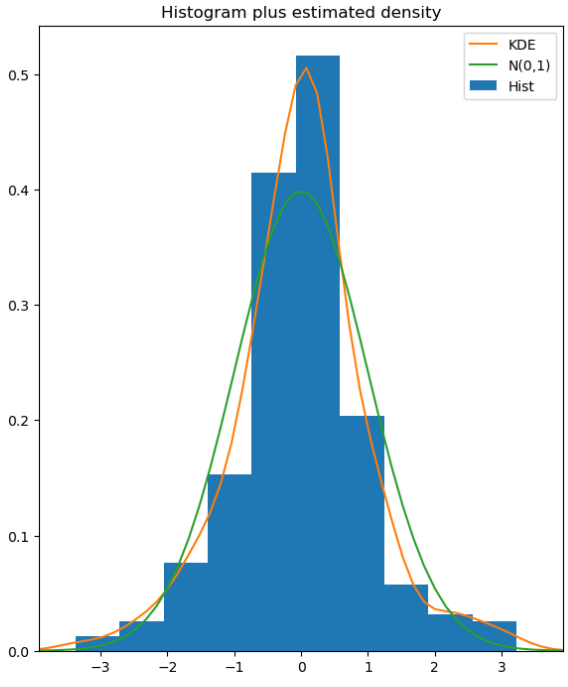
```
['66104']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8532      0.028     29.979      0.000       0.797
0.909
ma.L1          0.6579      0.048     13.745      0.000       0.564
0.752
ma.S.L12      -0.9592      0.115     -8.349      0.000      -1.184
-0.734
sigma2      8.856e+04   8088.236     10.949      0.000    7.27e+04
1.04e+05
==============================================================================
```


Standardized residual for "6"
Histogram plus estimated density
KDE
N(0,1)
Hist
Normal Q-Q
Sample Quantiles
Theoretical Quantiles
Correlogram



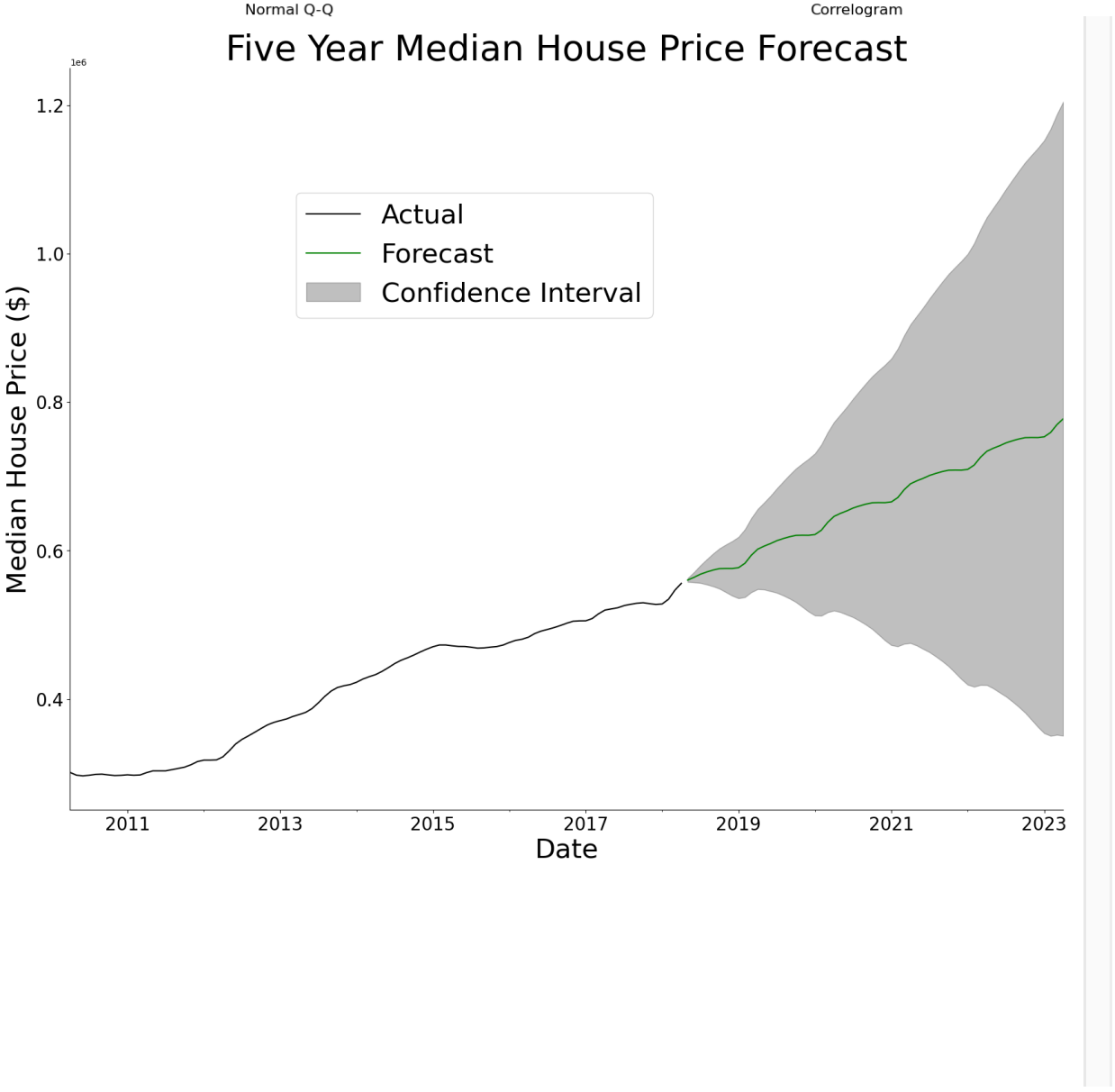

```
['48009']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7974      0.041     19.250      0.000       0.716      0.879
ma.L1          0.8200      0.039     21.040      0.000       0.744      0.896
ar.S.L12       0.0829      0.017      4.856      0.000       0.049      0.116
ma.S.L12      -0.3089      0.024    -12.926      0.000      -0.356     -0.262
sigma2      1.574e+06   1.25e+05     12.583      0.000    1.33e+06   1.82e+06
==============================================================================
```



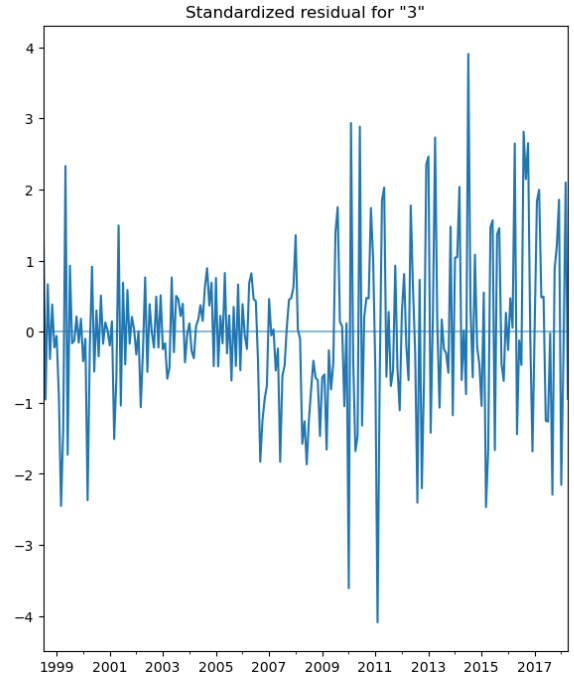


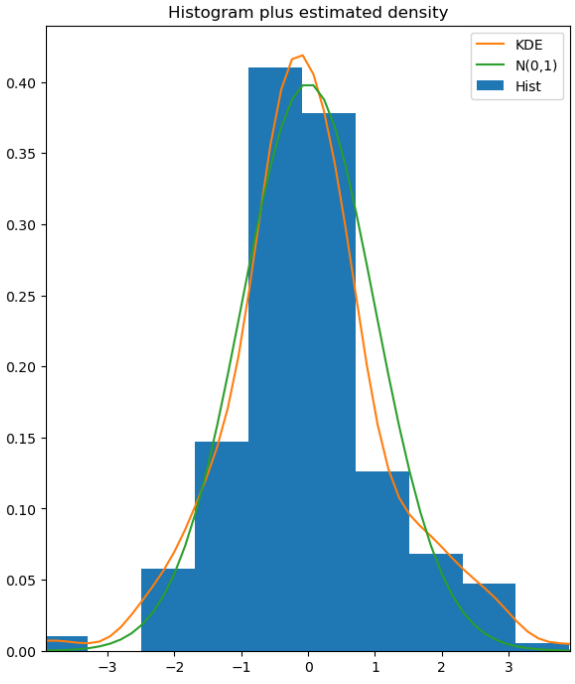
Normal Q-Q

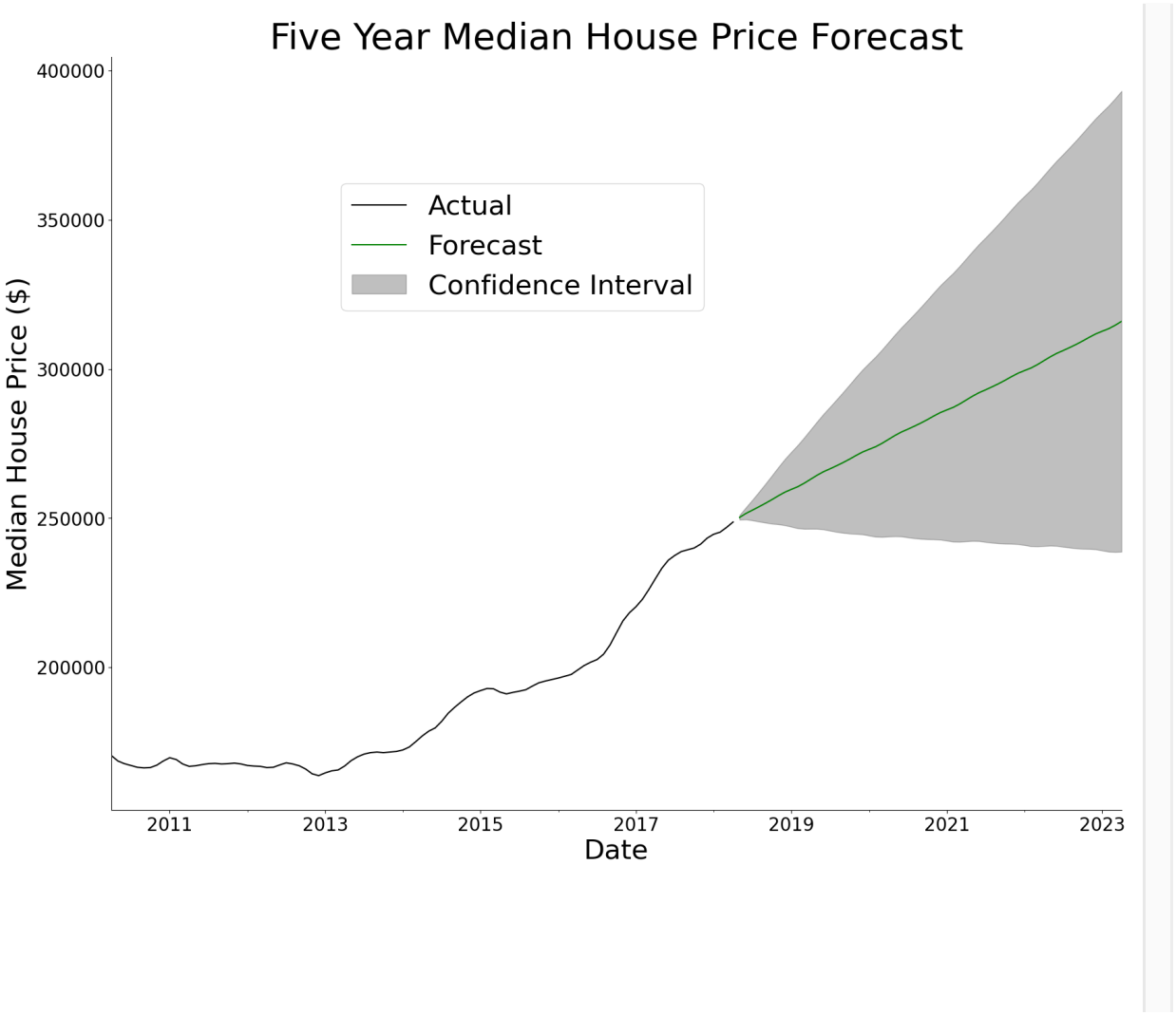
Correlogram



```
['35222']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8322      0.029     28.728      0.000       0.775      0.889
ma.L1          0.6149      0.033     18.823      0.000       0.551      0.679
ma.S.L12      -0.7776      0.045    -17.446      0.000      -0.865     -0.690
sigma2      1.534e+05   1.06e+04     14.487      0.000    1.33e+05   1.74e+05
==============================================================================
```

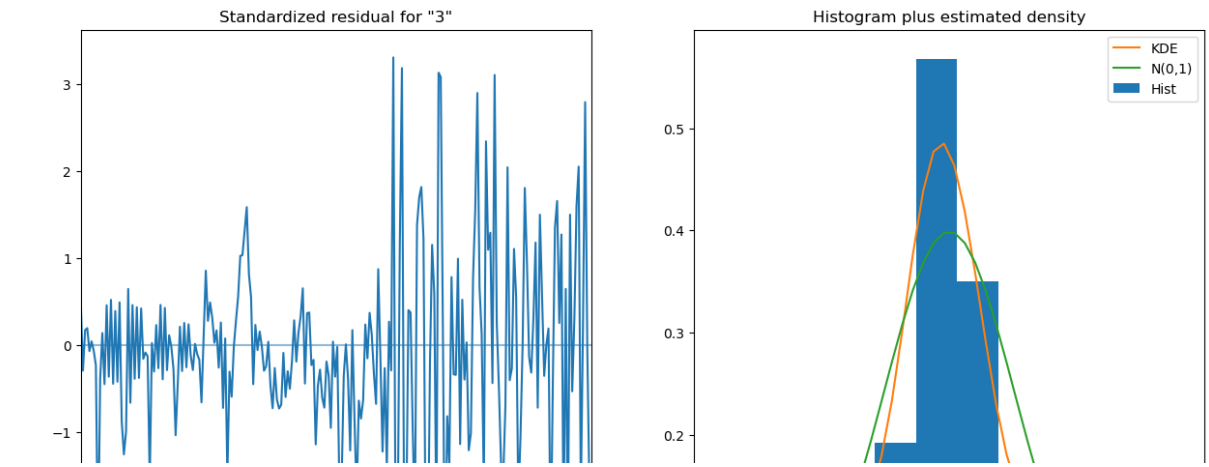

Standardized residual for "3"
4
3
2
1
0
−1
−2
−3
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

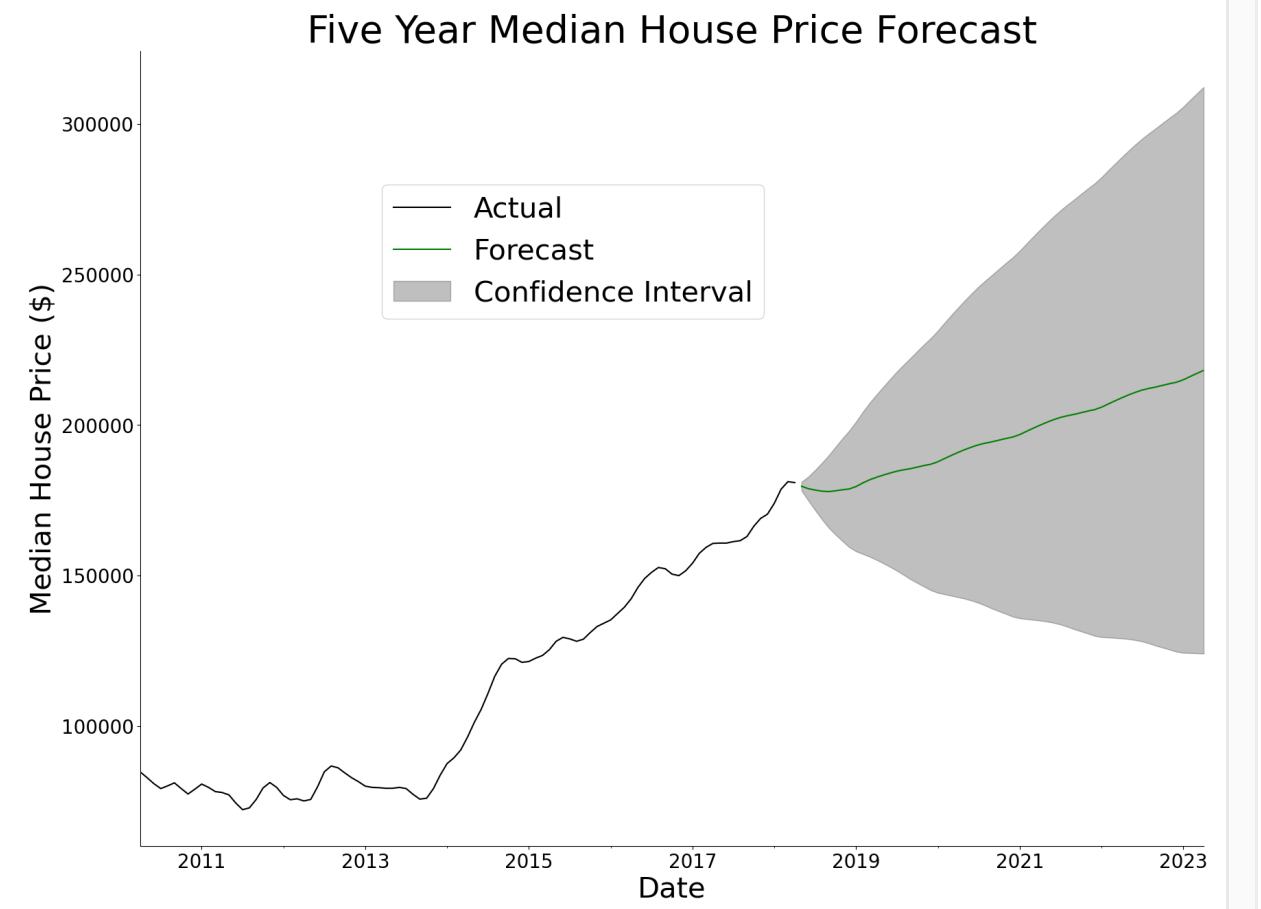

Histogram plus estimated density
KDE
N(0,1)
Hist
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−3
−2
−1
0
1
2
3


Five Year Median House Price Forecast
Actual
Forecast
Confidence Interval
Median House Price ($)
400000
350000
300000
250000
200000
2011
2013
2015
2017
2019
2021
2023
Date

```
['32204']
```

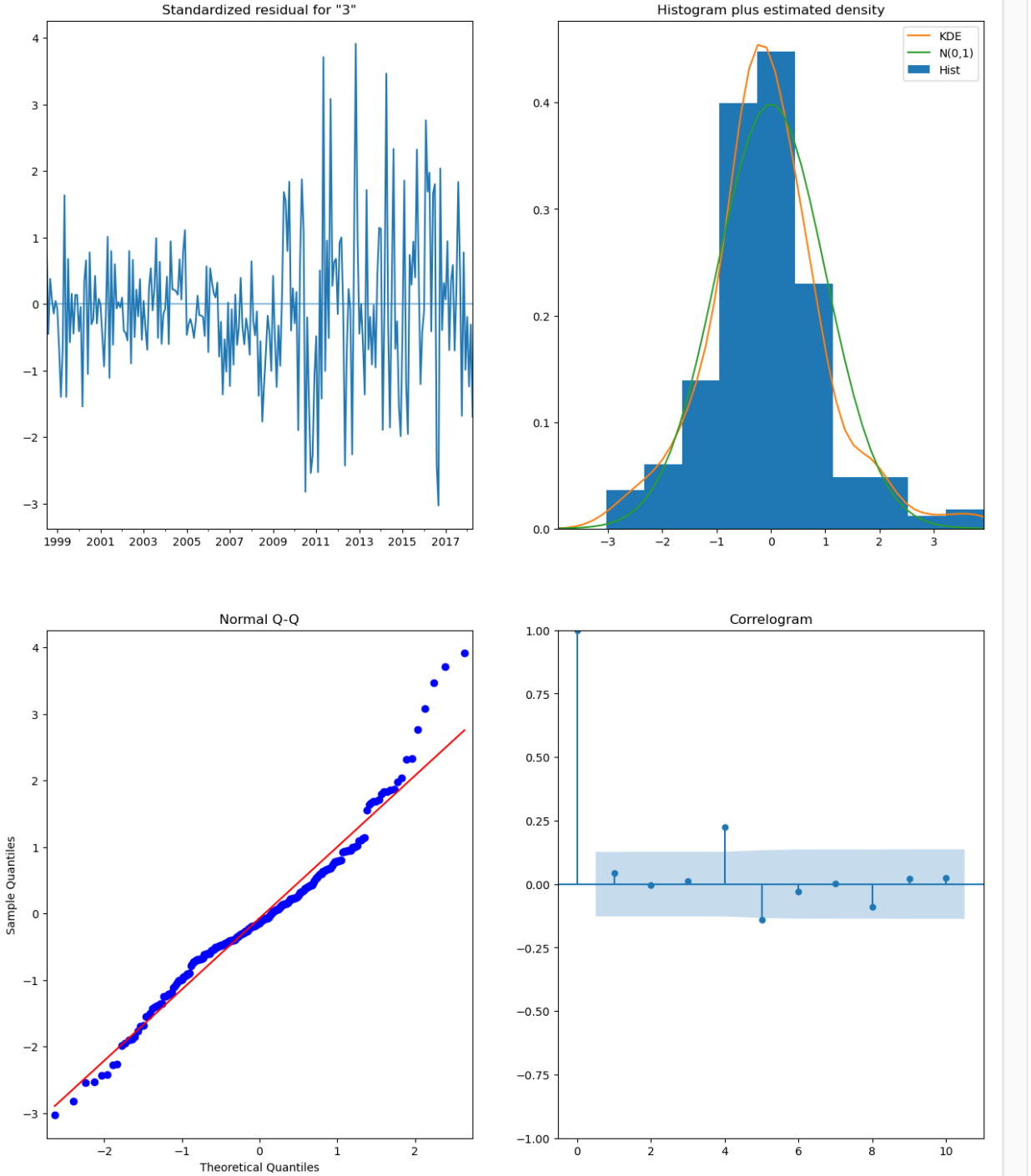
```
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.7482      0.039     19.425      0.000       0.673
0.824
ma.L1          0.7681      0.029     26.890      0.000       0.712
0.824
ar.S.L12       0.0770      0.048      1.616      0.106      -0.016
0.170
ma.S.L12      -0.9257      0.056    -16.407      0.000      -1.036
-0.815
sigma2      5.603e+05   4.88e+04     11.494      0.000    4.65e+05
6.56e+05
==============================================================================
```

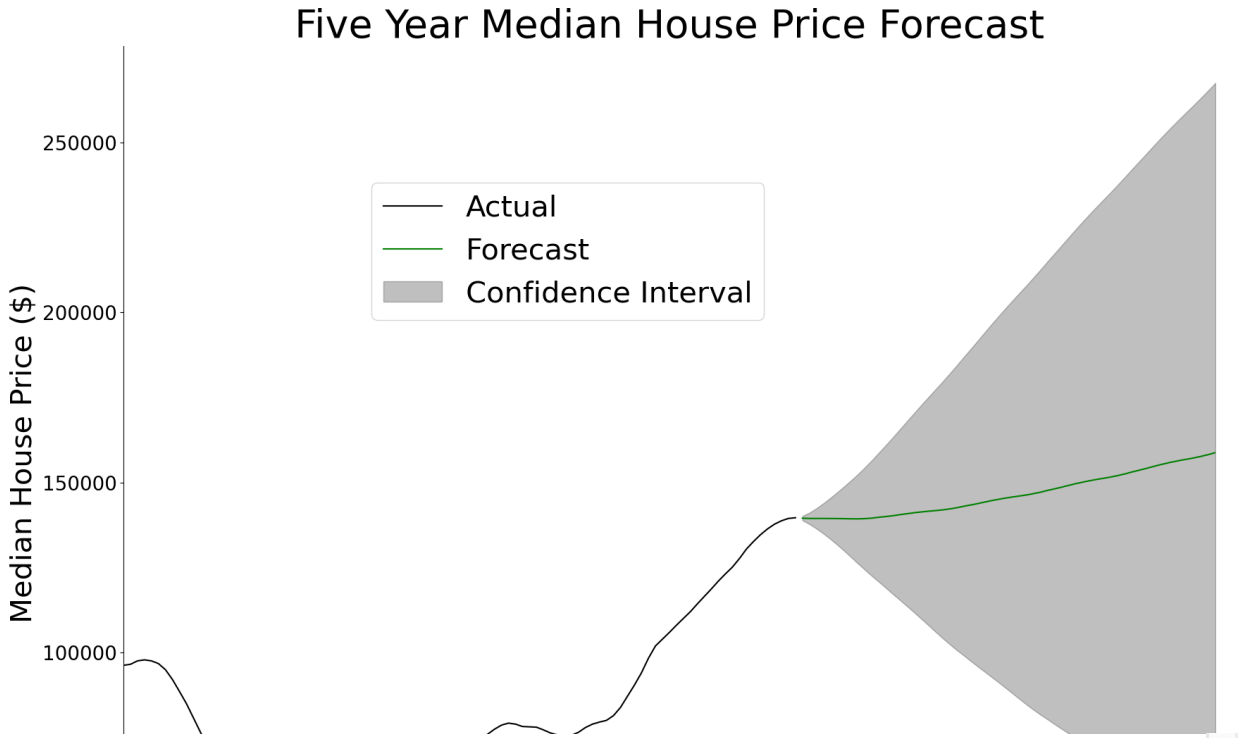




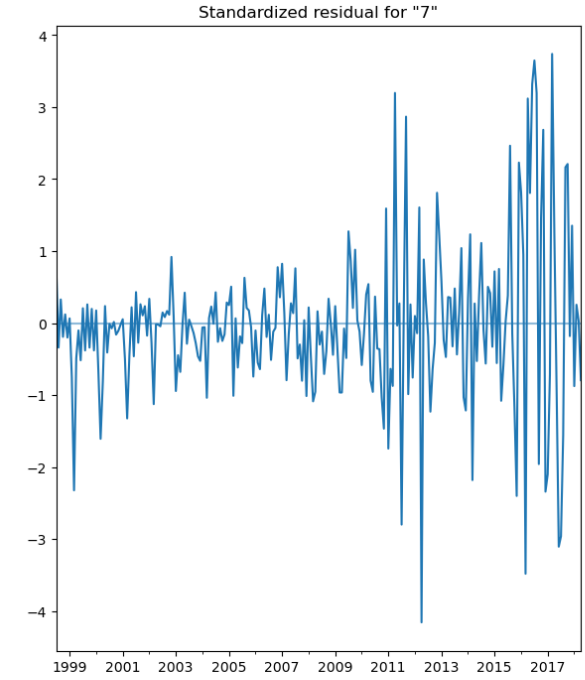

```
['32211']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
==========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
---------
ar.L1          0.9416      0.016     58.691      0.000       0.910
0.973
ma.L1          0.5652      0.043     13.280      0.000       0.482
0.649
ma.S.L12      -0.9980      1.560     -0.640      0.522      -4.056
```

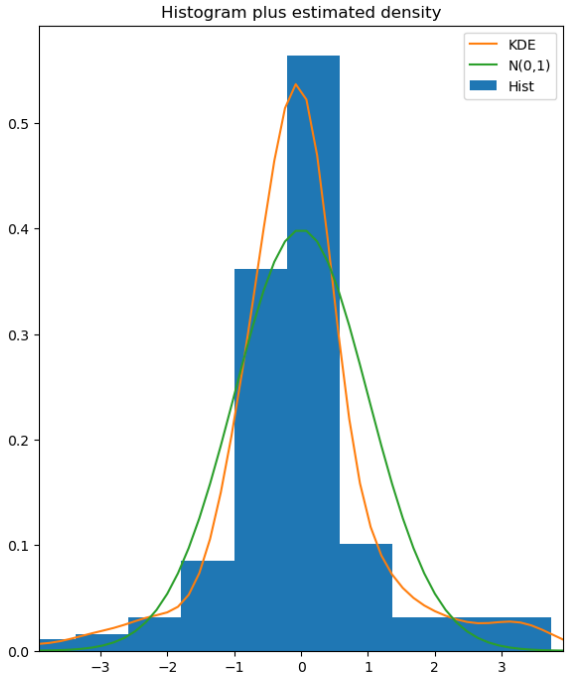
```
2.060
sigma2         9.894e+04    1.55e+05       0.639       0.523    -2.05e+05
4.02e+05
==============================================================================
==========
```

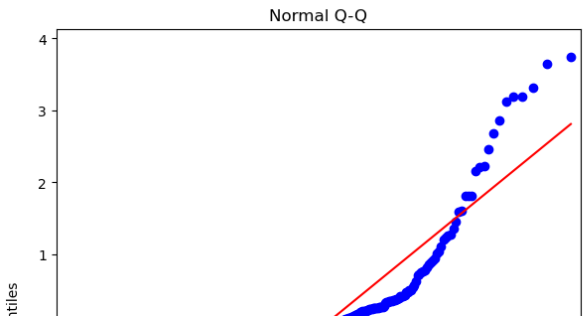


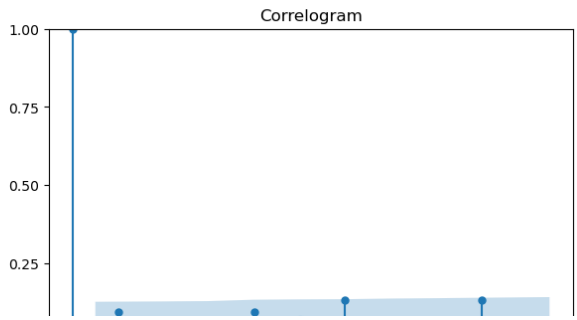



```
['73103']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7872      0.020     39.148      0.000       0.748      0.827
ma.L1          0.6972      0.026     26.317      0.000       0.645      0.749
ar.S.L12       0.0641      0.048      1.324      0.185      -0.031      0.159
ma.S.L12      -0.9775      0.120     -8.173      0.000      -1.212      -0.743
sigma2      3.537e+05   4.15e+04      8.527      0.000    2.72e+05   4.35e+05
==============================================================================
```


Standardized residual for "7"
4
3
2
1
0
−1
−2
−3
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

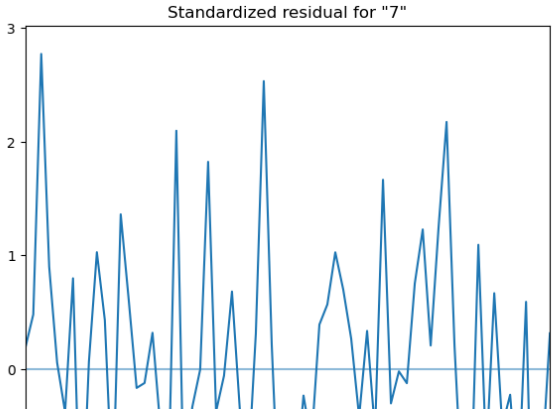

Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

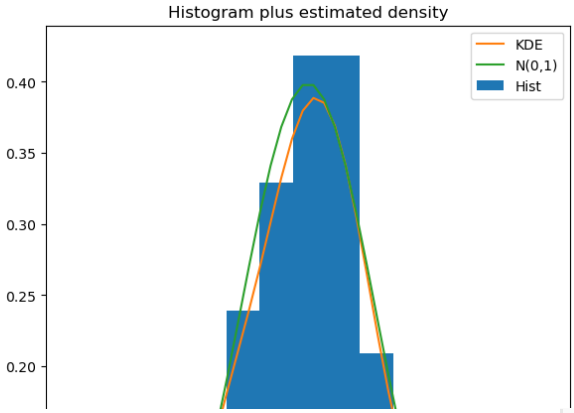

Normal Q-Q
4
3
2
1
ntiles

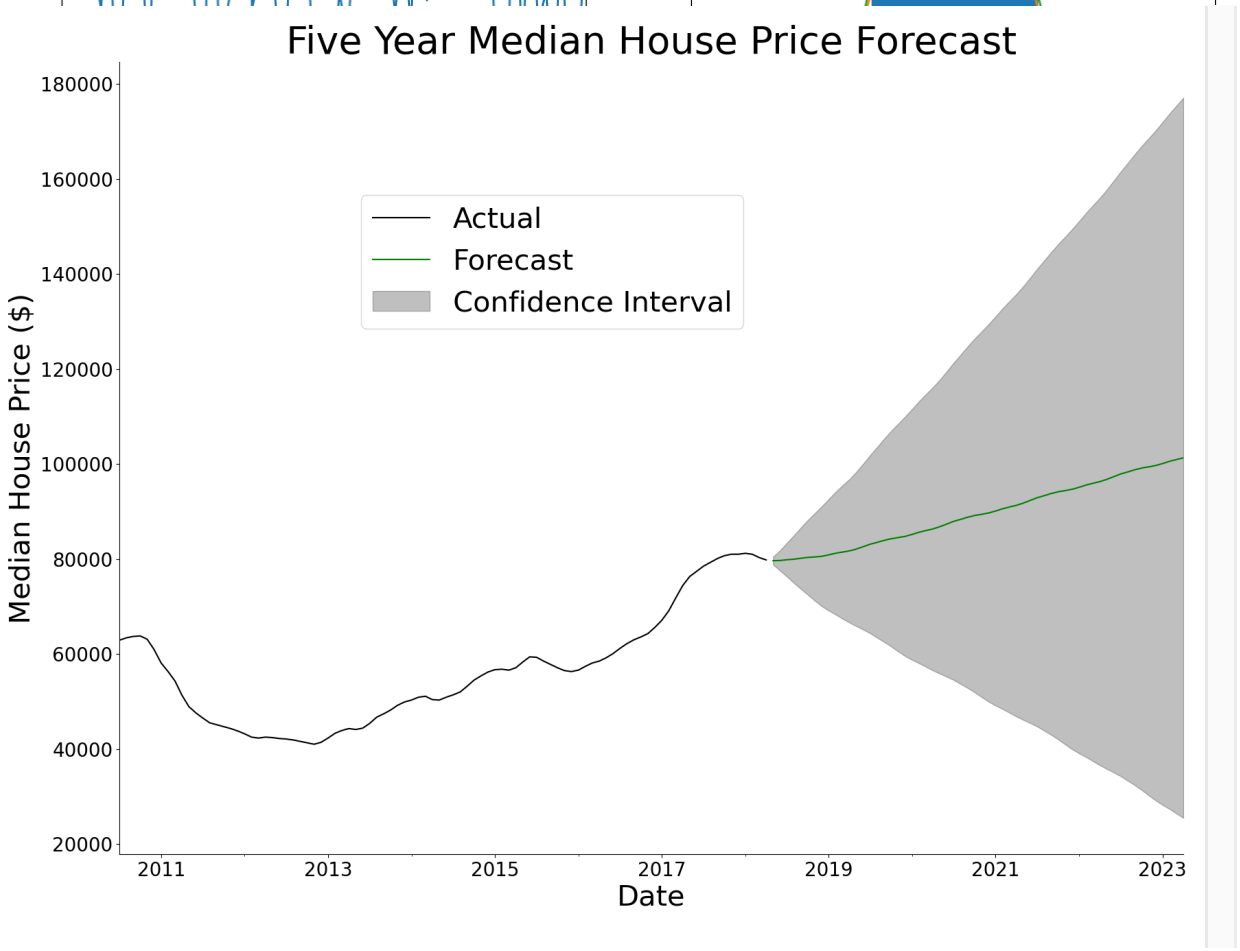

Correlogram
1.00
0.75
0.50
0.25

Five Year Median House Price Forecast

```
['73114']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7765      0.099      7.846      0.000       0.583      0.970
ma.L1          0.5774      0.113      5.120      0.000       0.356      0.799
ar.S.L12       0.0694      0.029      2.364      0.018       0.012      0.127
ma.S.L12      -0.6903      0.135     -5.095      0.000      -0.956     -0.425
sigma2      1.827e+05   3.32e+04      5.505      0.000    1.18e+05   2.48e+05
==============================================================================
```

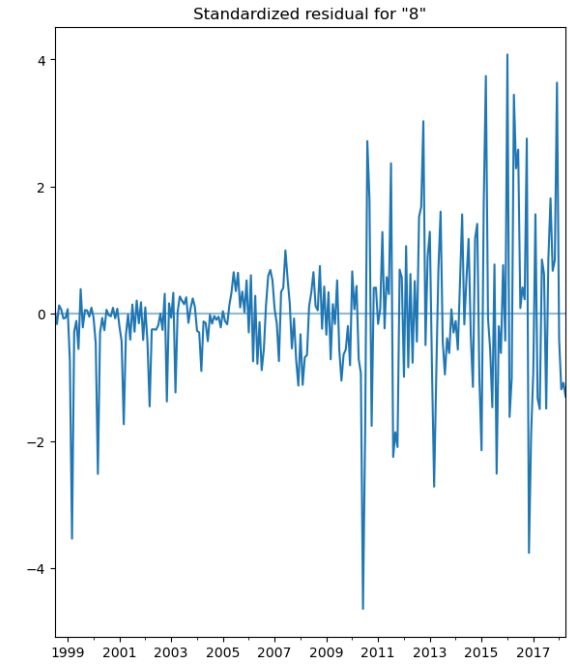


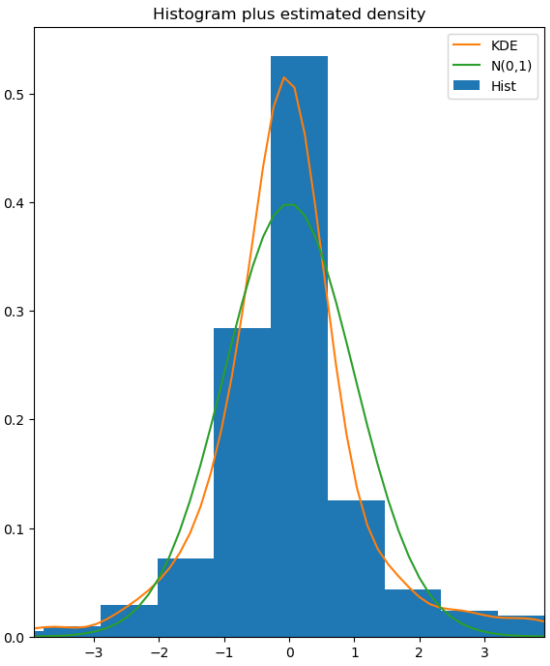


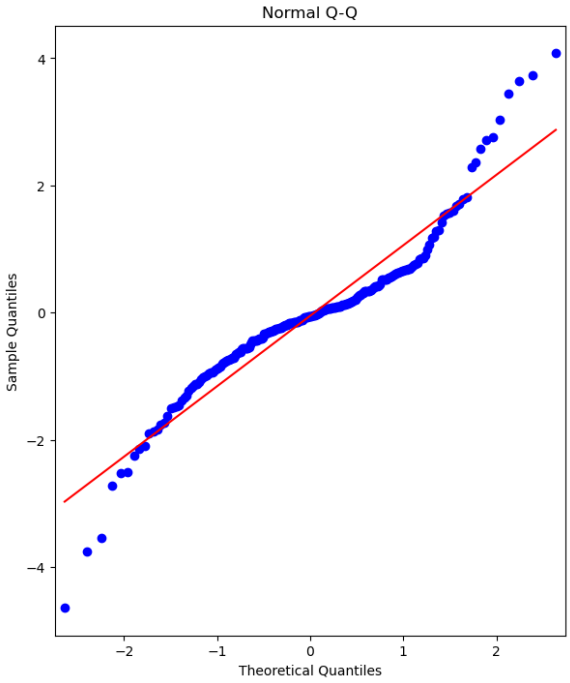


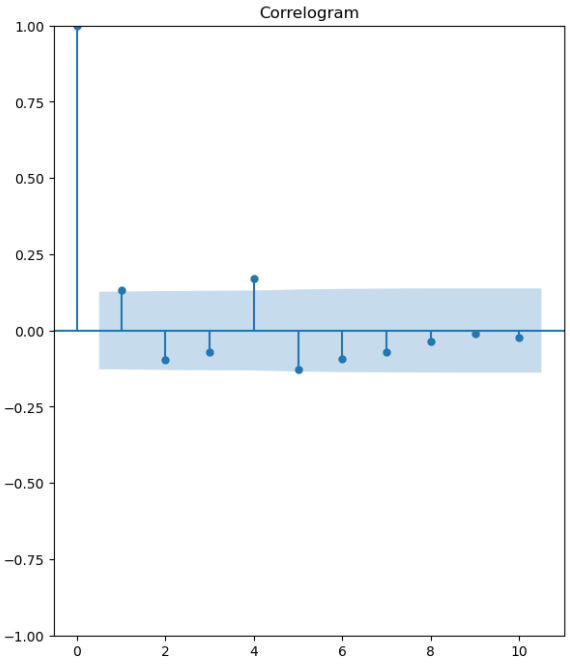

```
['84101']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
==========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
---------
ar.L1          0.7946      0.024     32.434      0.000       0.747
0.843
ma.L1          0.6922      0.030     23.236      0.000       0.634
0.751
ar.S.L12       0.1060      0.049      2.168      0.030       0.010
0.202
```

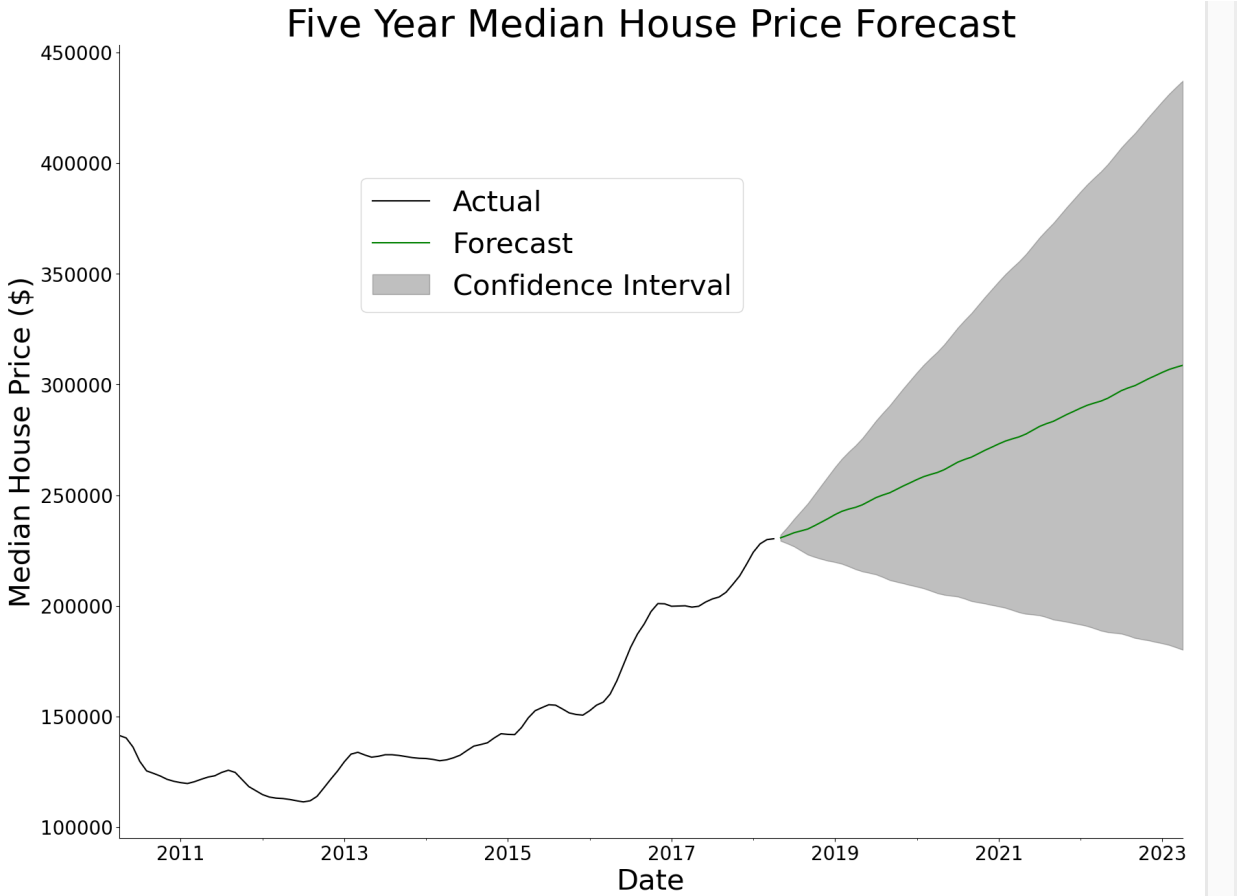
```
ma.S.L12       -0.7942       0.050     -15.779       0.000       -0.893
-0.696
sigma2      4.942e+05     2.9e+04      17.030       0.000     4.37e+05
5.51e+05
==============================================================================
==========
```





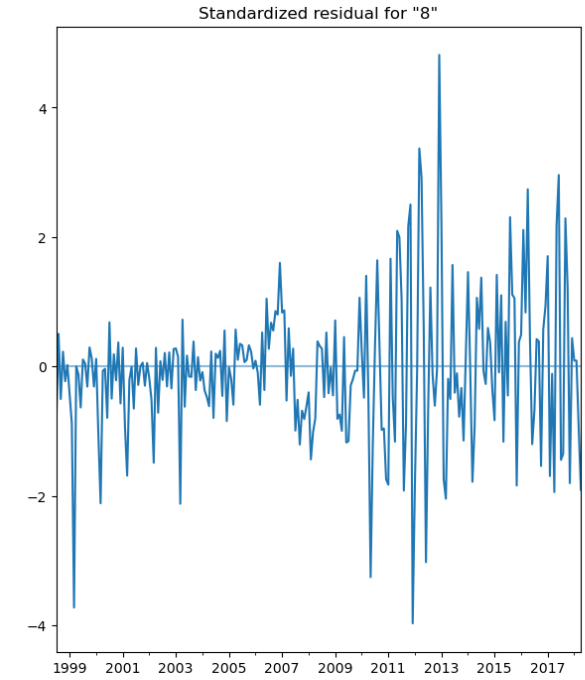


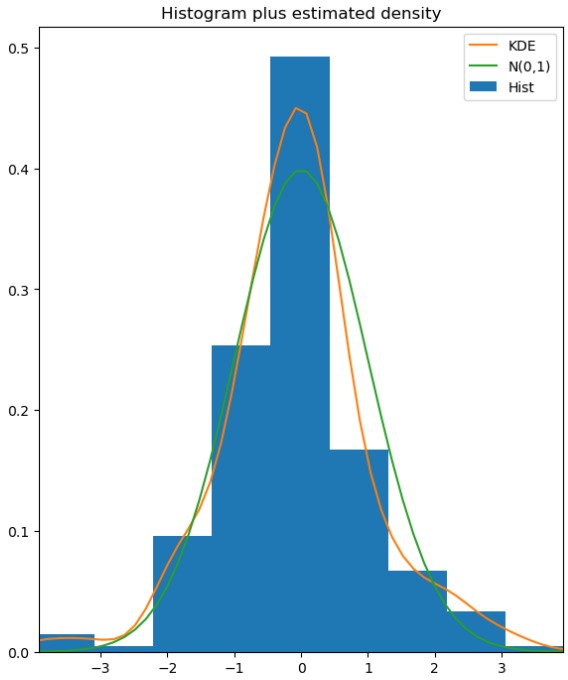


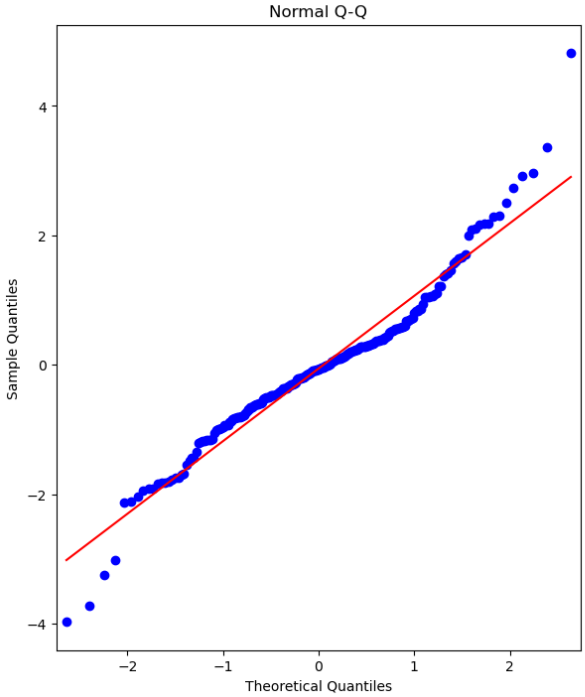


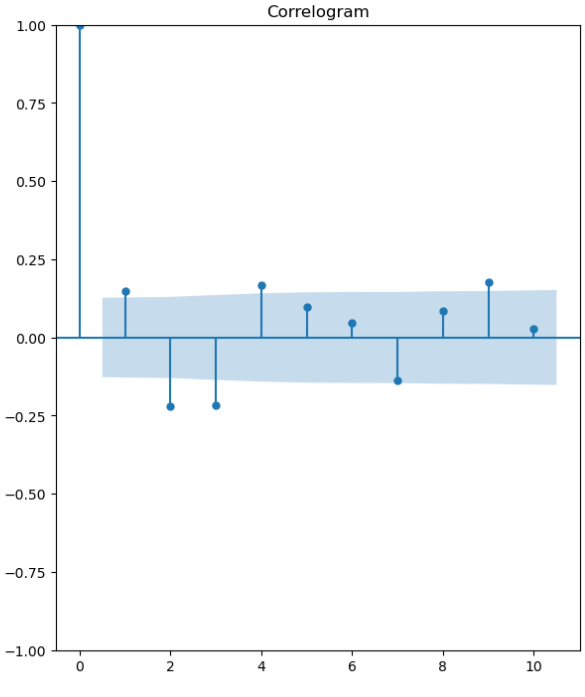

```
['84111']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8690      0.024     35.940      0.000       0.822
0.916
ma.L1          0.6710      0.033     20.342      0.000       0.606
0.736
ar.S.L12       0.0808      0.021      3.808      0.000       0.039
```

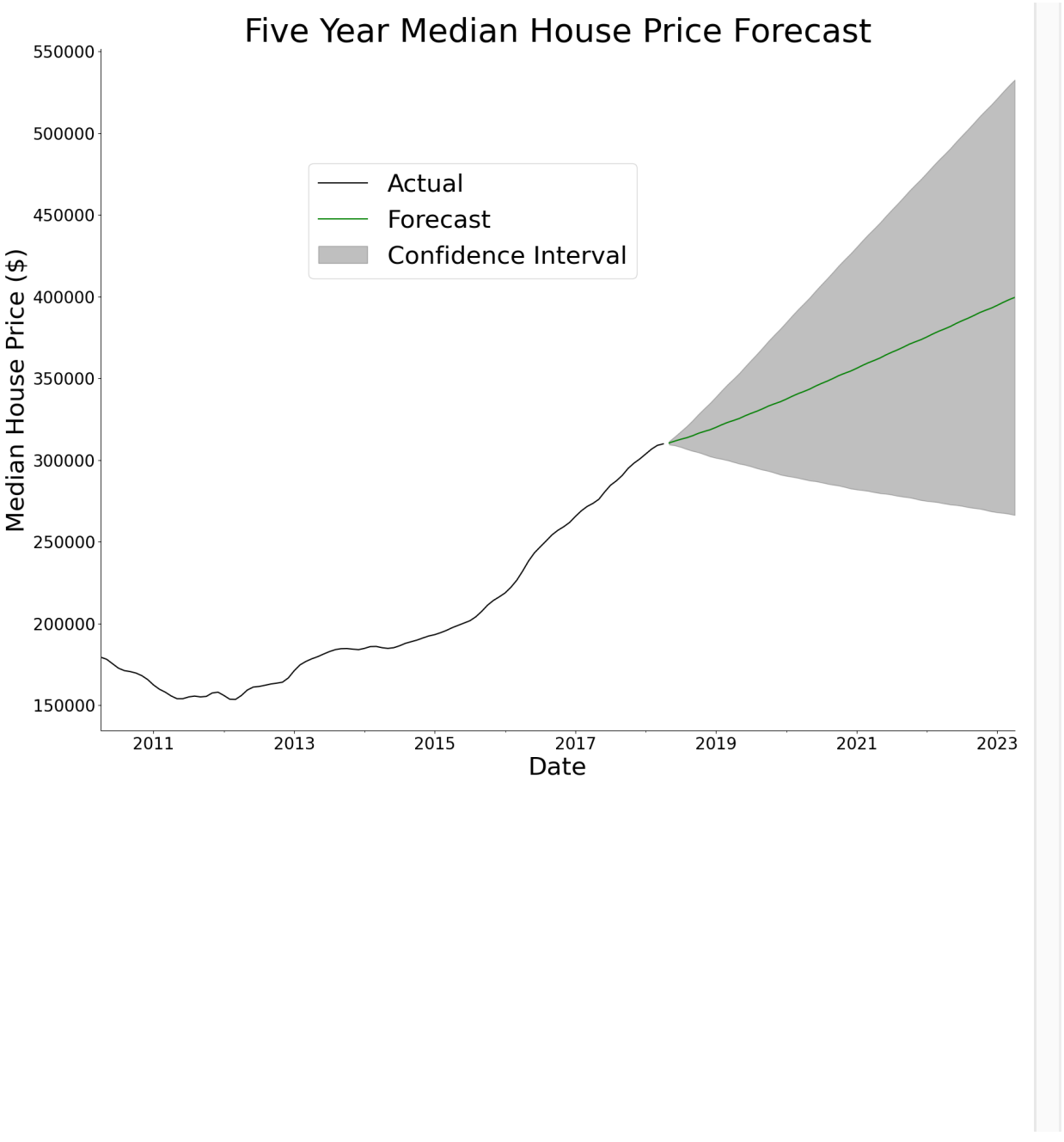
```
0.122
ma.S.L12       -0.8098       0.041    -19.724       0.000       -0.890
-0.729
sigma2      2.669e+05    1.75e+04      15.256       0.000     2.33e+05
3.01e+05
==============================================================================
==========
```










Five Year Median House Price Forecast
Actual
Forecast
Confidence Interval
Median House Price ($)
550000
500000
450000
400000
350000
300000
250000
200000
150000
2011
2013
2015
2017
2019
2021
2023
Date

In [52]:

```
# Create column for predicted price after 5 years
midsize_cities['5_Year_Price'] = pred_list_3

# Calculate Percent Return from time of investment
midsize_cities['Rate_Return'] = ((
    midsize_cities['5_Year_Price']-
    midsize_cities['2018-04'])/
    midsize_cities['2018-04'])*100

midsize_cities

```

executed in 50ms, finished 13:49:06 2022-05-03

Out[52]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05
4121	66102	Kansas City	KS	Kansas City	Wyandotte	4122	38800.0	38800.0
4293	66104	Kansas City	KS	Kansas City	Wyandotte	4294	41300.0	41200.0
4502	48009	Birmingham	MI	Detroit	Oakland	4503	242700.0	242600.0
8459	35222	Birmingham	AL	Birmingham	Jefferson	8460	103400.0	104200.0
8259	32204	Jacksonville	FL	Jacksonville	Duval	8260	41400.0	41400.0
2596	32211	Jacksonville	FL	Jacksonville	Duval	2597	62000.0	61700.0
9540	73103	Oklahoma City	OK	Oklahoma City	Oklahoma	9541	46800.0	47000.0
5702	73114	Oklahoma City	OK	Oklahoma City	Oklahoma	5703	NaN	NaN
9058	84101	Salt Lake City	UT	Salt Lake City	Salt Lake	9059	85100.0	85400.0
6557	84111	Salt Lake City	UT	Salt Lake City	Salt Lake	6558	116700.0	116900.0

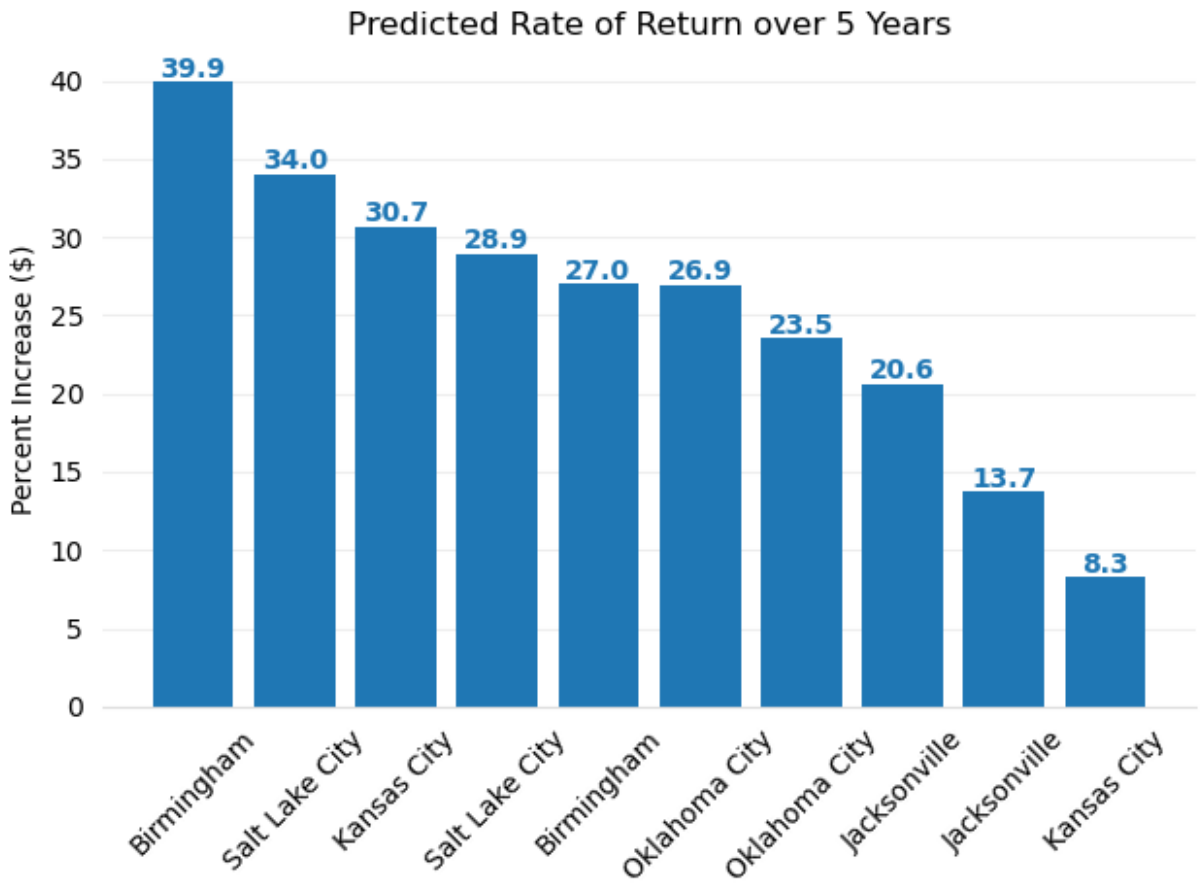
10 rows × 273 columns

5.2.2 Bar Chart: Rate of Return

In [70]:

```
# create df and sort by Rate_Return in Descending Order
rate_3 = midsize_cities[['Rate_Return','City']]
#df.sort_values('importances',inplace=True)
rate_3.sort_values('Rate_Return',inplace=True,ascending=False)

fig, ax = plt.subplots()

# Save the chart so we can loop through the bars below.
bars = ax.bar(
    x=np.arange(rate_3['Rate_Return'].size),
    height=rate_3['Rate_Return'],
    tick_label=rate_3['City']
)

# Axis formatting.
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.spines['bottom'].set_color('#DDDDDD')
ax.tick_params(bottom=False, left=False)
ax.set_axisbelow(True)
ax.yaxis.grid(True, color='#EEEEEE')
ax.xaxis.grid(False)

# Grab the color of the bars so we can make the
# text the same color.
bar_color = bars[0].get_facecolor()

# Add text annotations to the top of the bars.
# Note, you'll have to adjust this slightly (the 0.3)
# with different data.
for bar in bars:
  ax.text(
      bar.get_x() + bar.get_width() / 2,
      bar.get_height() + 0.3,
      round(bar.get_height(), 1),
      horizontalalignment='center',
      color=bar_color,
      weight='bold'
  )

plt.xticks(rotation = 45)
plt.title('Predicted Rate of Return over 5 Years')
plt.ylabel('Percent Increase ($)')

fig.tight_layout()
```

executed in 392ms, finished 13:56:55 2022-05-03



5.2.3 Summary:

- Mid-Size Cities do not have a great rate of return, however a few zip codes could be worth a closer look.
- Birmingham, Michigan 48009 and Salt Lake City, Utah 84101 both have return rates over 34%, but they also have higher risk than the top_prospects zip codes

6 SARIMA Model

- For Top Prospective Zip Codes

In [12]:

```
# Define the p, d and q parameters to take any value between 0 and 2
p = d = q = range(0, 2)

# Generate all different combinations of p, q and q triplets
pdq = list(itertools.product(p, d, q))

# Generate all different combinations of seasonal p, q and q triplets
pdqs = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]
```

executed in 11ms, finished 10:46:33 2022-05-03

6.1 GridSearch optimal parameters, Model, Predict, calculate RMSE

- Run for all zip codes in prospects

In [13]:

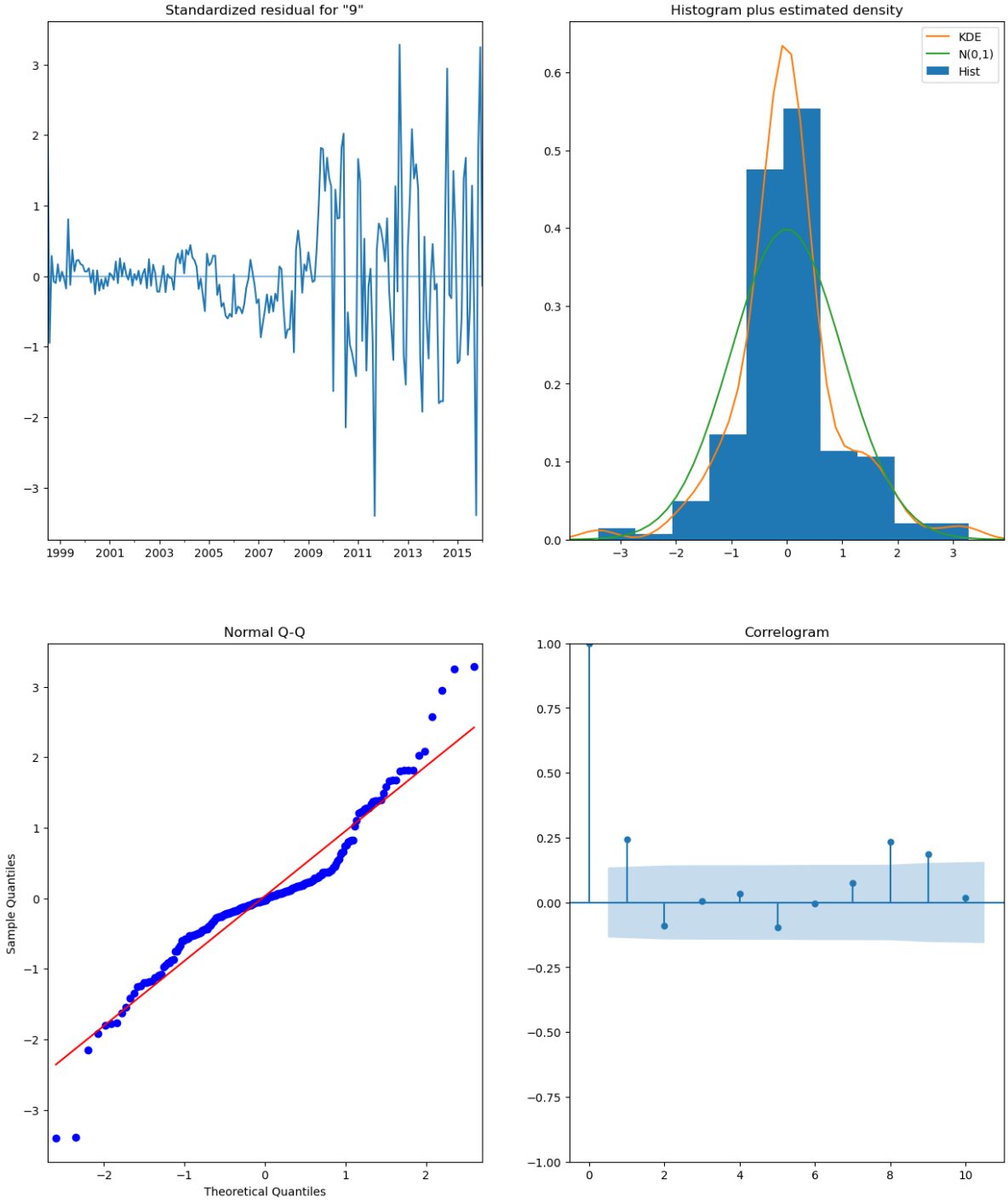
```
# Create list to add rmse values during iteration
rmse_list = []



# iterate over prospects to extract optimal SARIMA values,
# instantiate SARIMAX model, make predictions
# and calculate the RMSE for each zip code
for zipcode in prospects['ZipCode']:
    zips = prospects.loc[prospects['ZipCode'] == zipcode]
    m1 = melt_data(zips)
    m1[str(zipcode)] = m1['value']
    m1 = m1.drop('value', axis=1)

    # create train and test sets
    train = m1.iloc[:238]
    test = m1.iloc[238:]

    # Run a grid with pdq and seasonal pdq parameters calculated above and ge
    ans = []
    outputs = []
    comb_list = []
    combs_list = []

    for comb in pdq:
        for combs in pdqs:
            try:
                mod = sm.tsa.statespace.SARIMAX(train,
                                                order=comb,
                                                seasonal_order=combs,
                                                enforce_stationarity=False,
                                                enforce_invertibility=False)

                output = mod.fit()
                ans.append([comb, combs, output.aic])
                outputs.append([output.aic])
                comb_list.append([comb])
                combs_list.append([combs])
                AIC_df = pd.DataFrame(list(zip(outputs, comb_list, combs_list
                #print('ARIMA {} x {}12 : AIC Calculated ={}'.format(comb, co
            except:
                continue

    # Print the lowest AIC and it's parameters for SARIMA
    AIC_df[0] = AIC_df[0].str.get(0)
    best_AIC = AIC_df.copy()
    lowest_AIC = min(best_AIC[0])
    lowest_AIC_row = AIC_df.loc[AIC_df[0] == lowest_AIC]
    #print(lowest_AIC_row)

    # call value for comb and combs for model input
    lowest_AIC_row[1] = lowest_AIC_row[1].str.get(0)
```

```
    lowest_AIC_row[2] = lowest_AIC_row[2].str.get(0)
    arima = lowest_AIC_row.iat[0,1]
    s = lowest_AIC_row.iat[0,2]
    print(str(m1.columns.values))
    print(arima, s)

    # Plug the optimal parameter values into a new SARIMAX model
    ARIMA_MODEL = sm.tsa.statespace.SARIMAX(train,
                                            order=arima,
                                            seasonal_order=s,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)

    # Fit the model and print results
    output = ARIMA_MODEL.fit()

    print(output.summary().tables[1])

    # Call plot_diagnostics() on the results calculated above
    output.plot_diagnostics(figsize=(15, 18))
    plt.show()

    # Get actual results in dataframe
    price_truth = test['2016-02-01':]

    # Get predictions and forecast in dataframe
    prediction = output.get_forecast(steps=27)
    pred = prediction.predicted_mean
    forecast_result = output.get_forecast(steps=75)
    forecast = forecast_result.predicted_mean

    # Get confidence intervals of forecasts
    #pred_conf = forecast.conf_int()

    # Get the real and predicted values
    prediction = pred[0:]
    price_truth = price_truth[str(zipcode)]

    # Compute the mean square error
    mse = ((prediction - price_truth) ** 2).mean()
    rmse = round(np.sqrt(mse))
    rmse_list.append(rmse)
    print('The Root Mean Squared Error of our forecasts is {}'.format(
        round(rmse, 2)))
    print()
    print()

# Print list of RMSE
print(rmse_list)
```

executed in 8m 58s, finished 10:55:31 2022-05-03

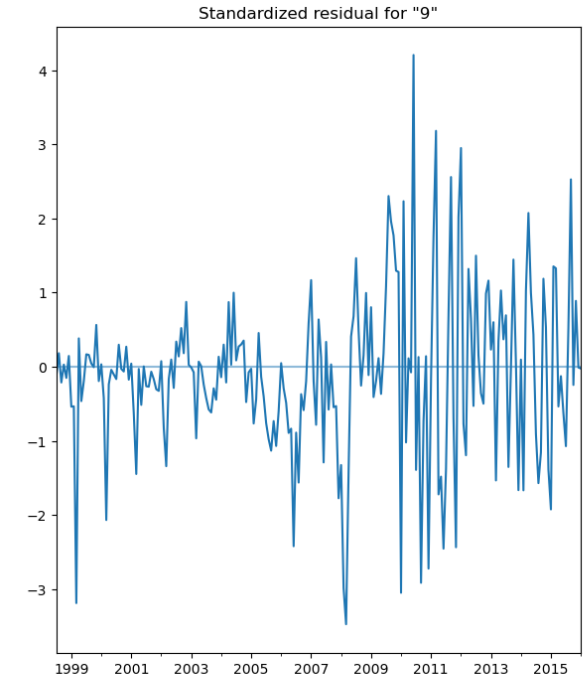
```
['94601']
(1, 1, 1) (1, 1, 1, 12)
================================================================
```

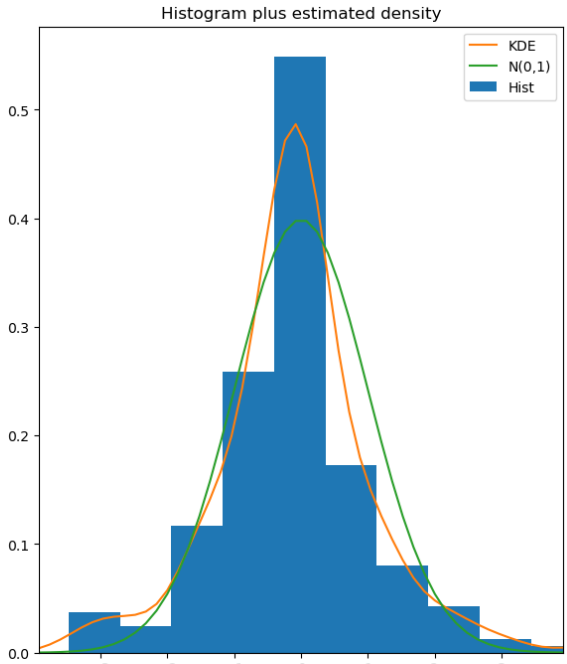
```
===========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------
-----------
ar.L1          0.9596      0.022     44.037      0.000       0.917
1.002
ma.L1          0.3172      0.024     13.207      0.000       0.270
0.364
ar.S.L12      -0.6059      0.067     -9.009      0.000      -0.738
-0.474
ma.S.L12      -0.0231      0.044     -0.520      0.603      -0.110
0.064
sigma2      1.631e+06   1.27e+05     12.846      0.000    1.38e+06
1.88e+06
========================================================================
===========
```

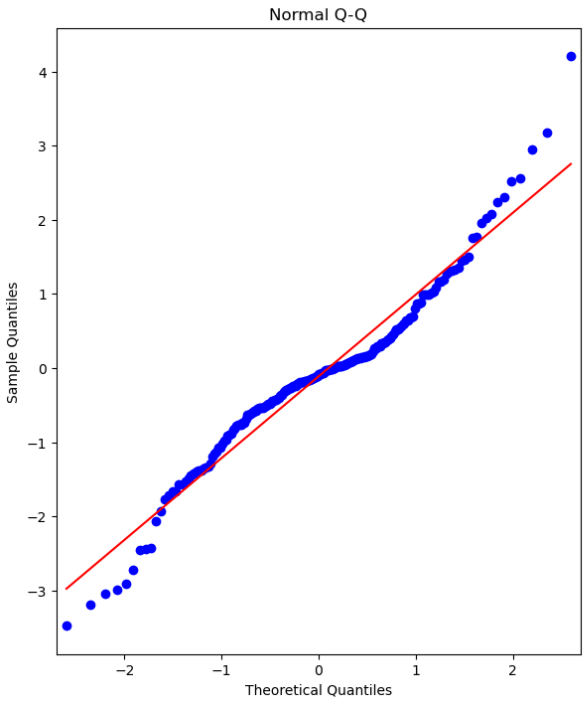


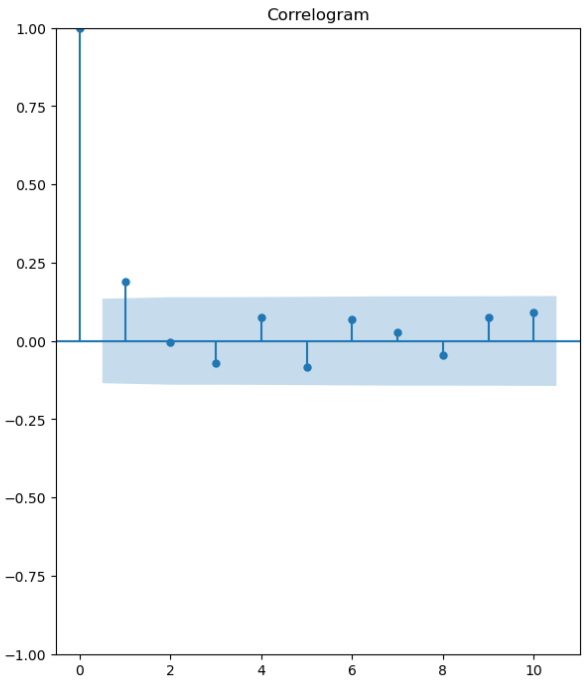

```
The Root Mean Squared Error of our forecasts is 58117.0


['94590']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9712      0.016     61.356      0.000       0.940
1.002
ma.L1          0.5069      0.039     13.013      0.000       0.431
```

```
0.583
ar.S.L12       0.1627       0.032       5.014       0.000         0.099
0.226
ma.S.L12      -0.8593       0.046     -18.616       0.000        -0.950
-0.769
sigma2      4.474e+05    3.39e+04      13.214       0.000     3.81e+05
5.14e+05
==============================================================================
===========
```

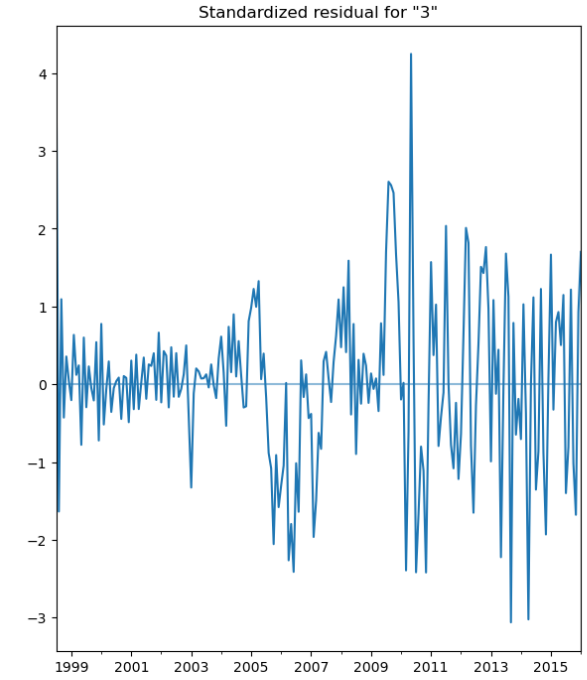


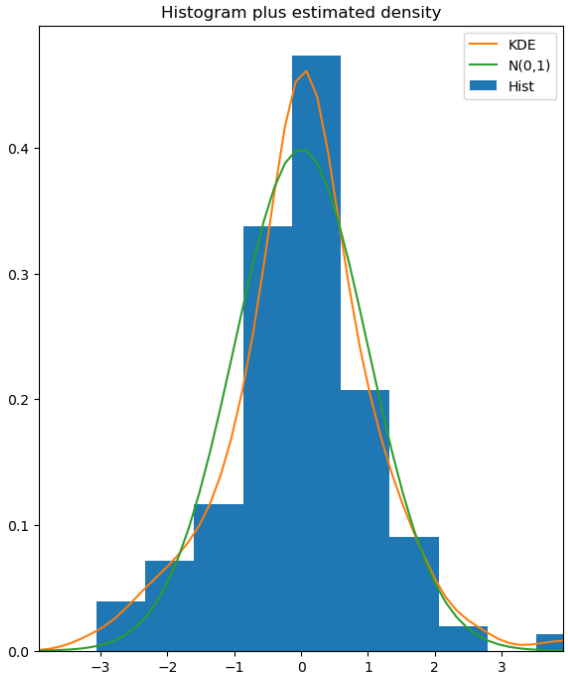


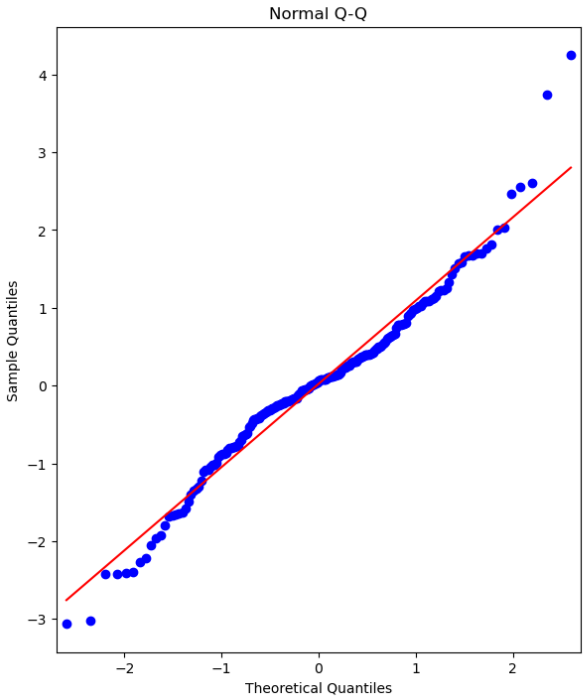


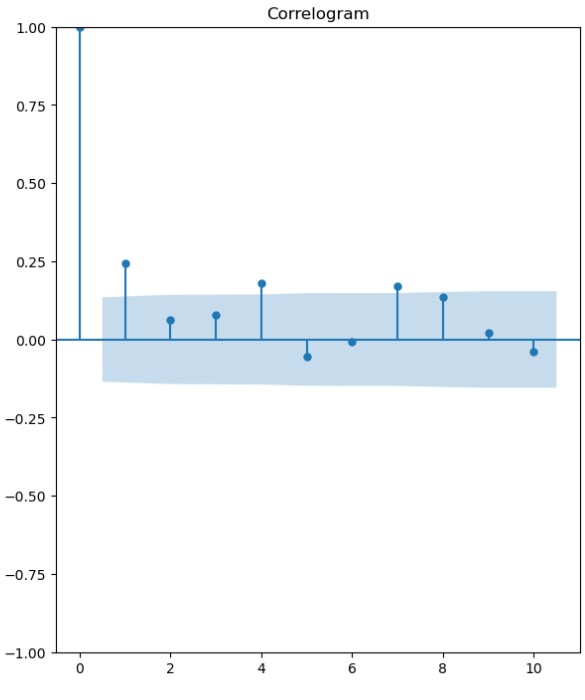


```
The Root Mean Squared Error of our forecasts is 9986.0


['33460']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9778      0.014     68.825      0.000       0.950       1.006
ma.L1          0.5459      0.024     22.808      0.000       0.499       0.593
ar.S.L12      -0.6195      0.063     -9.809      0.000      -0.743      -0.496
ma.S.L12      19.6087     27.280      0.719      0.472     -33.859      73.076
sigma2       403.6864   1126.099      0.358      0.720   -1803.427    2610.799
==============================================================================
```

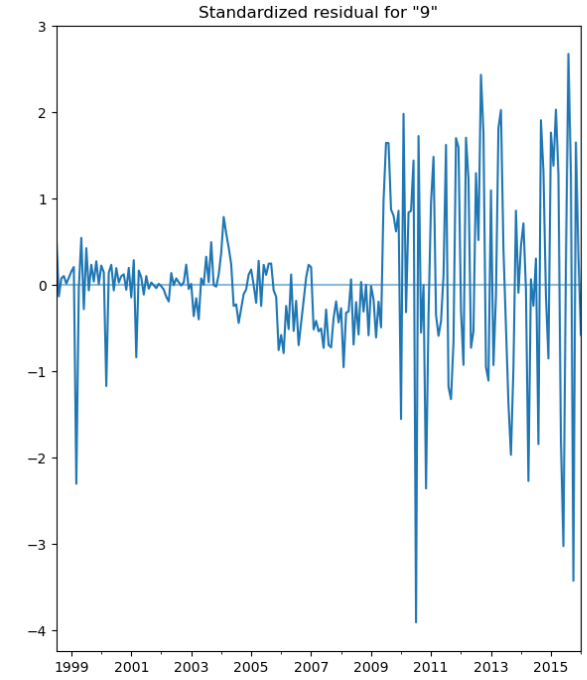

Standardized residual for "3"
4
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015

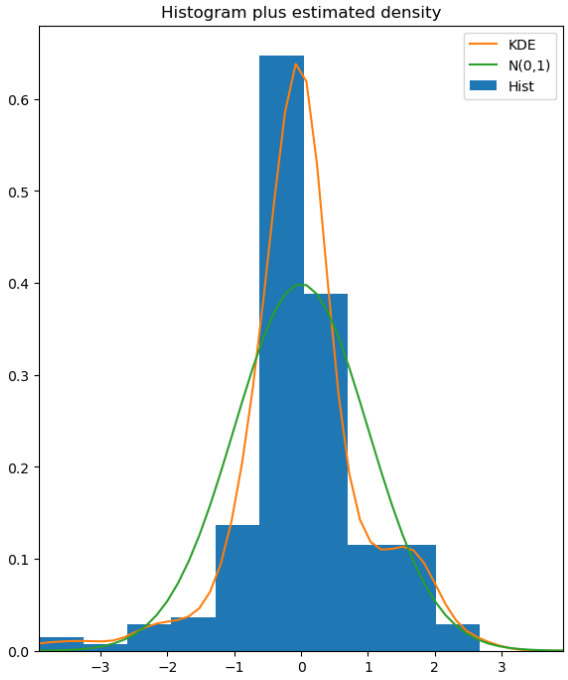

Histogram plus estimated density
KDE
N(0,1)
Hist
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

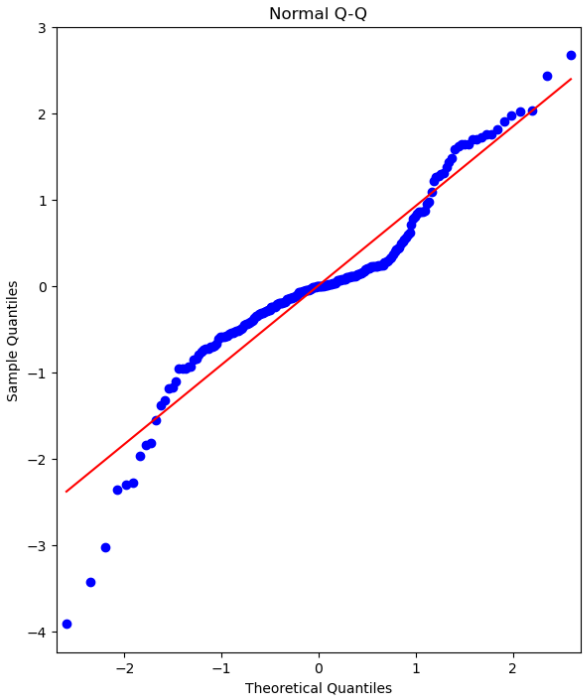

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

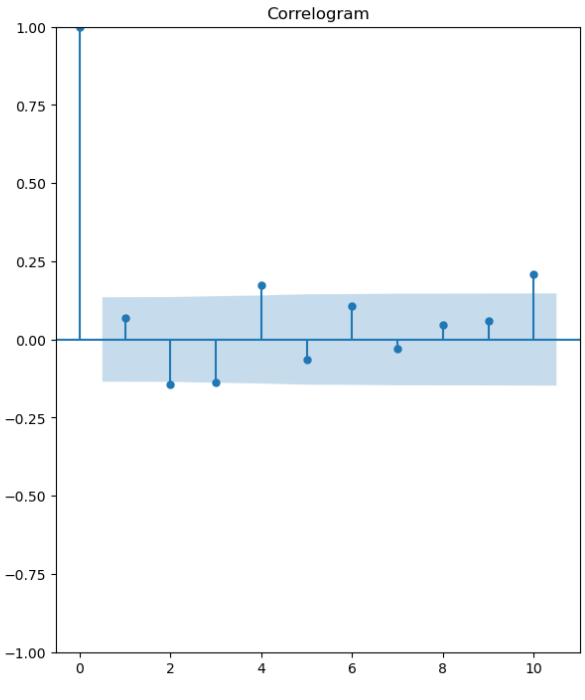

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 2775.0


['94606']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8734      0.034     25.839      0.000       0.807      0.940
ma.L1          0.6692      0.047     14.135      0.000       0.576      0.762
ar.S.L12       0.2098      0.083      2.515      0.012       0.046      0.373
ma.S.L12      -0.5293      0.066     -8.053      0.000      -0.658     -0.400
sigma2      2.408e+06   2.24e+05     10.770      0.000    1.97e+06   2.85e+06
==============================================================================
```

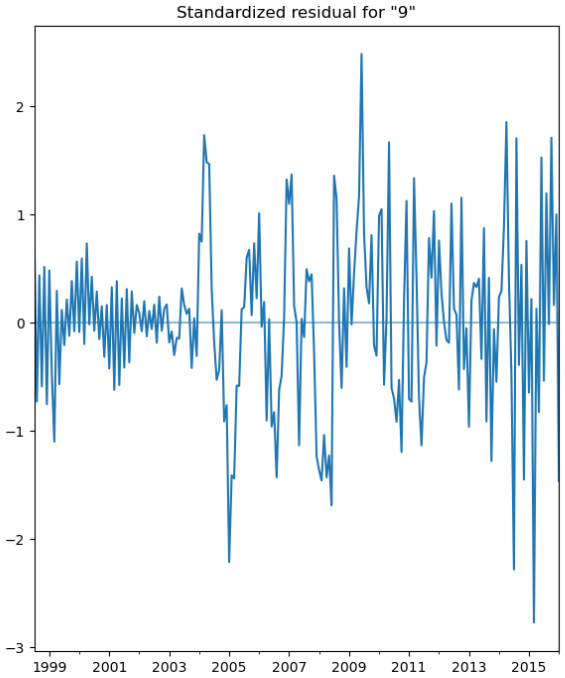

Standardized residual for "9"
3
2
1
0
−1
−2
−3
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015

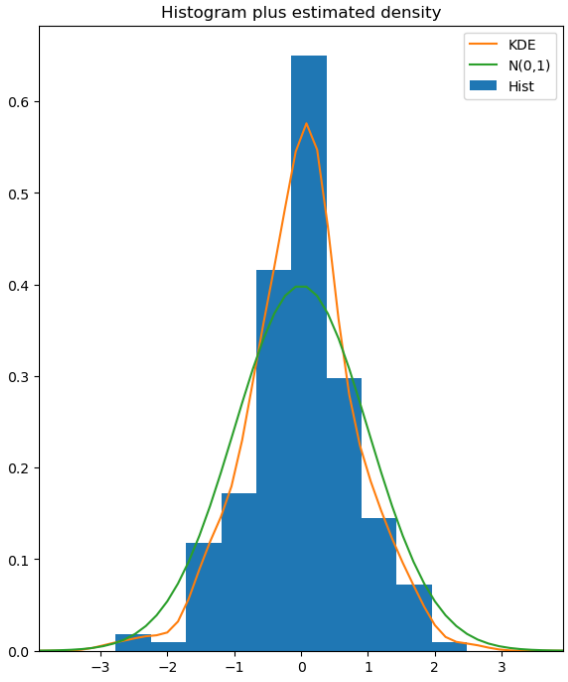

Histogram plus estimated density
KDE
N(0,1)
Hist
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

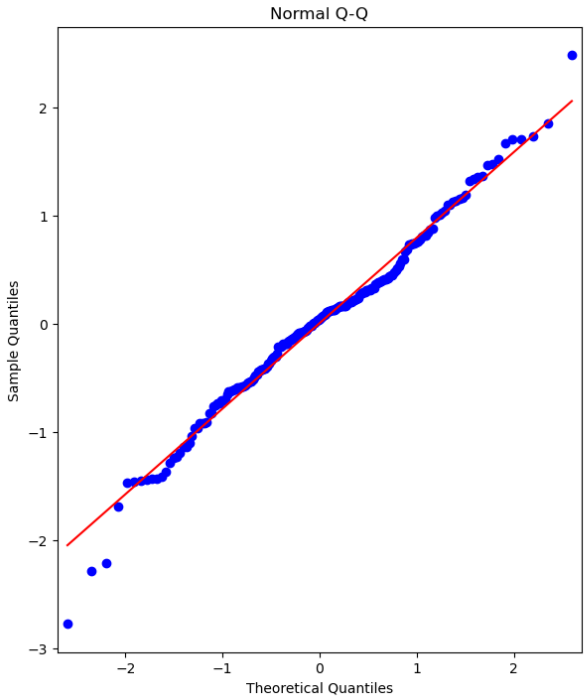

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
3
2
1
0
−1
−2
−3
−4
−2
−1
0
1
2

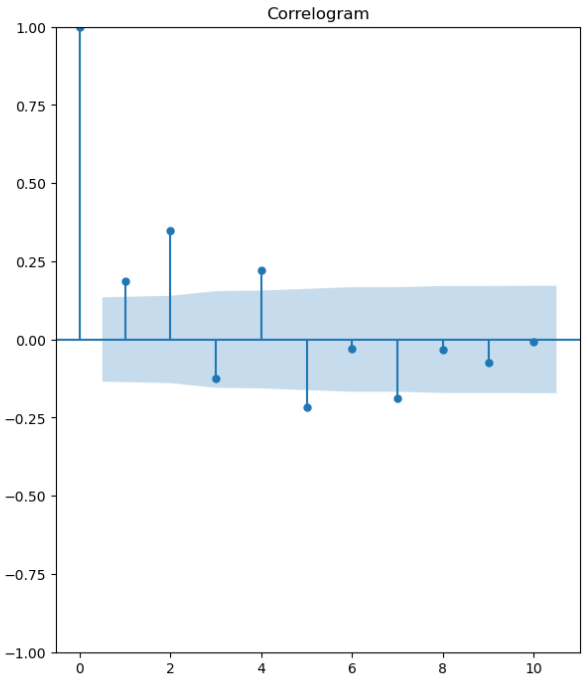

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 20071.0


['94804']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9260      0.038     24.364      0.000       0.852
1.001
ma.L1          1.1079      0.040     27.458      0.000       1.029
1.187
ar.S.L12       0.1276      0.028      4.592      0.000       0.073
0.182
ma.S.L12      -0.2794      0.074     -3.784      0.000      -0.424
-0.135
sigma2      2.364e+06   3.69e+05      6.404      0.000    1.64e+06
3.09e+06
==============================================================================
```

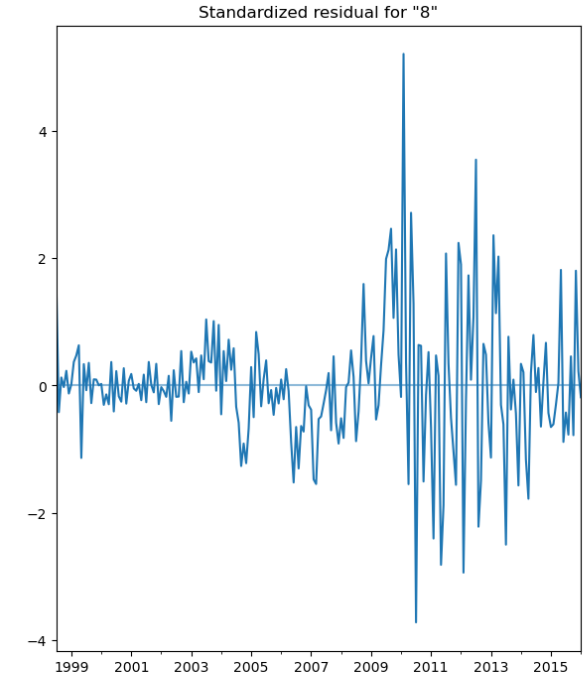

Standardized residual for "9"
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015

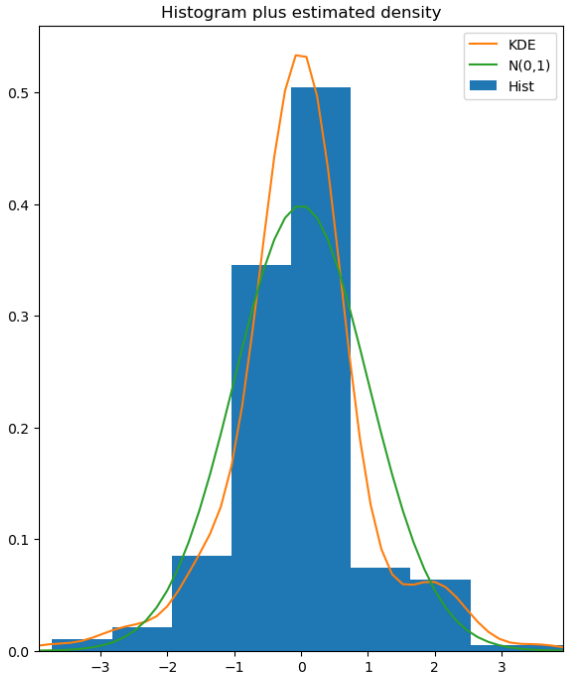

Histogram plus estimated density
KDE
N(0,1)
Hist
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

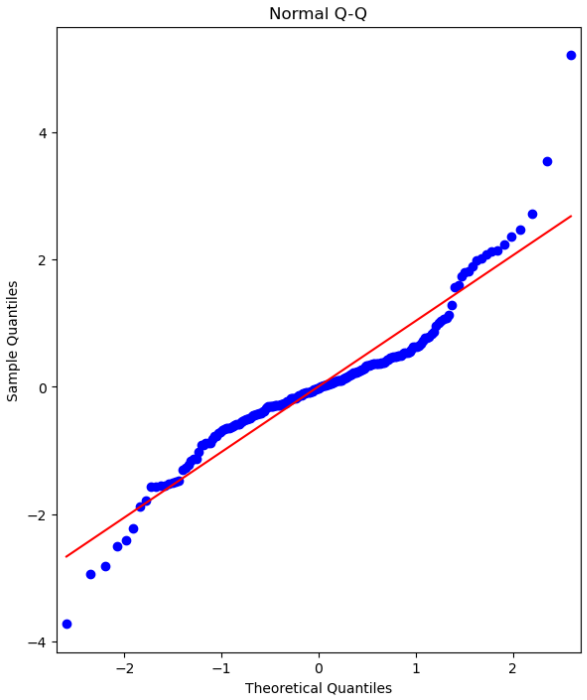

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
2
1
0
−1
−2
−3
−2
−1
0
1
2

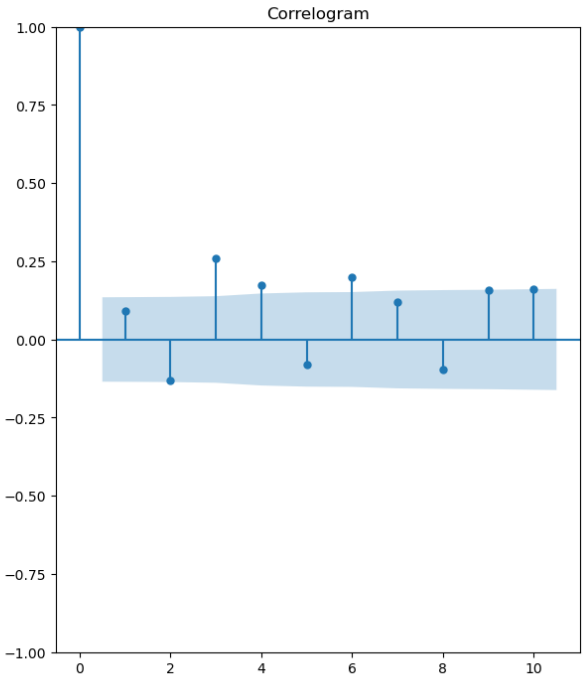

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 44237.0


['89104']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9780      0.013     74.511      0.000       0.952       1.004
ma.L1          0.2738      0.037      7.446      0.000       0.202       0.346
ar.S.L12      -0.5424      0.040    -13.727      0.000      -0.620      -0.465
ma.S.L12      16.2301      8.367      1.940      0.052      -0.170      32.630
sigma2       903.7866    935.906      0.966      0.334    -930.555    2738.128
==============================================================================
```



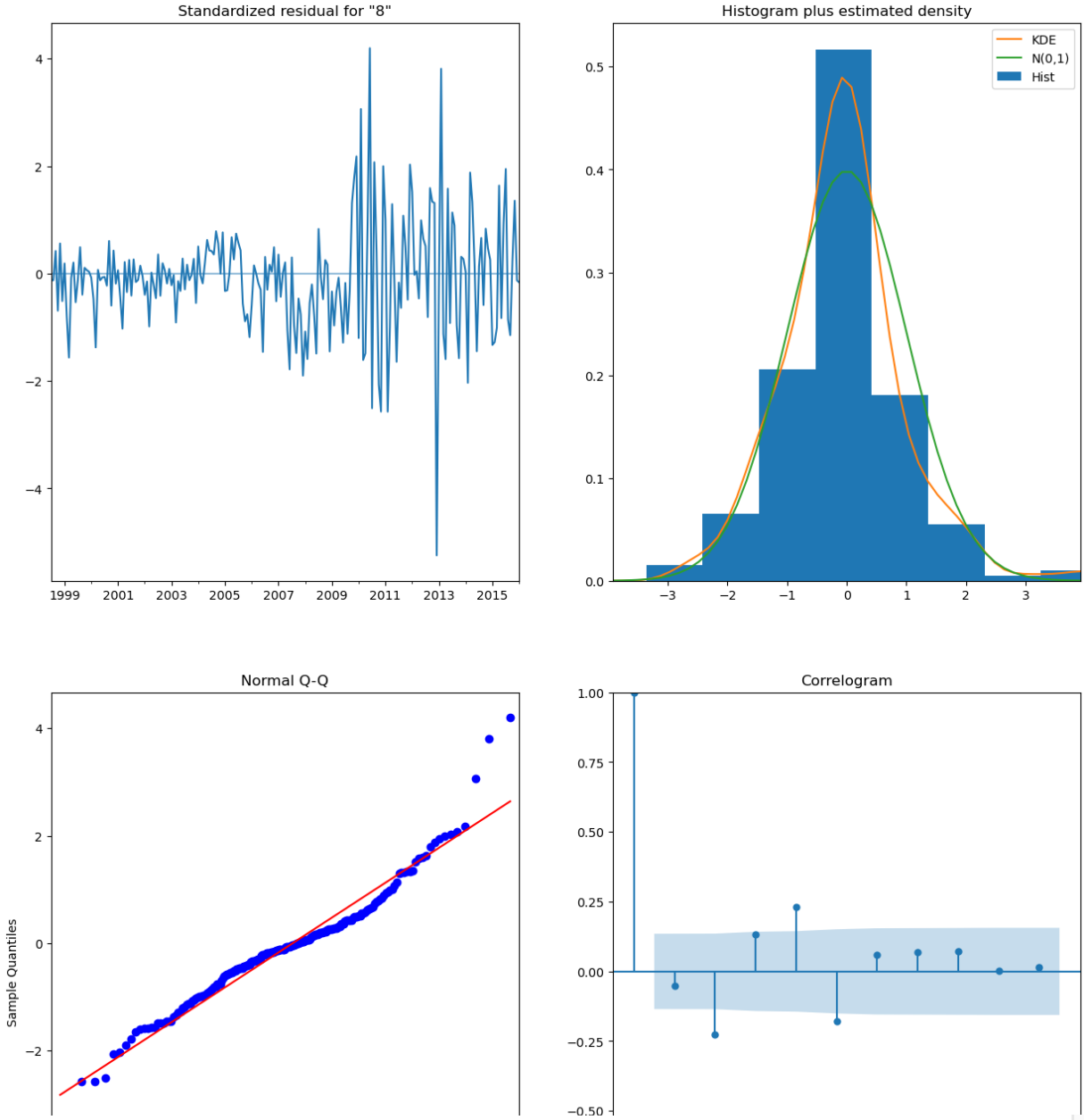







```
The Root Mean Squared Error of our forecasts is 12709.0


['85008']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
```

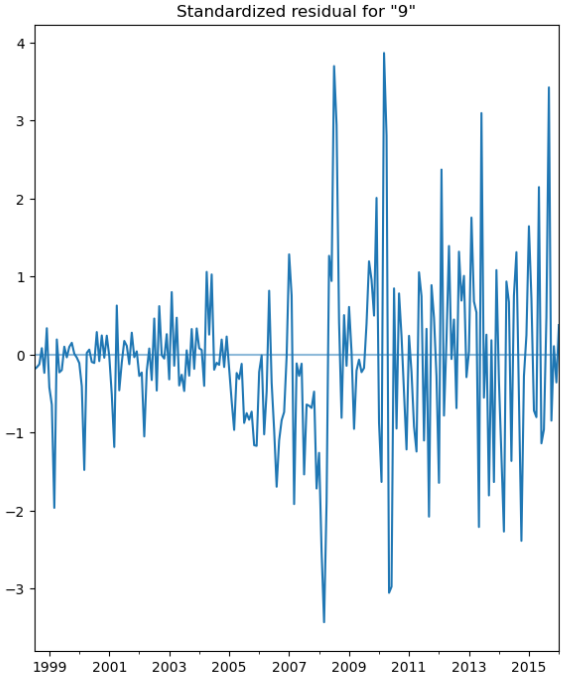
```
------------------------------------------------------------------------------
ar.L1          0.9468      0.023     41.557      0.000       0.902       0.991
ma.L1          0.5435      0.039     14.114      0.000       0.468       0.619
ar.S.L12       0.0930      0.038      2.423      0.015       0.018       0.168
ma.S.L12      -0.9883      0.411     -2.403      0.016      -1.794      -0.182
sigma2      2.325e+05   9.64e+04      2.412      0.016    4.35e+04    4.21e+05
==============================================================================
```

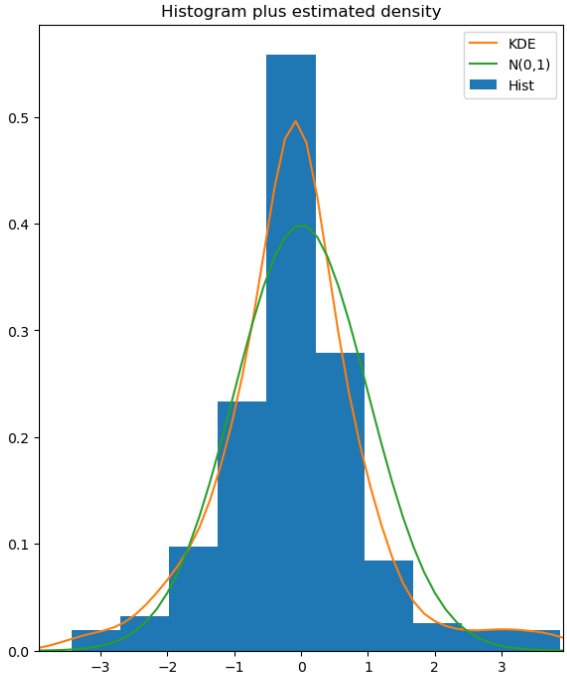


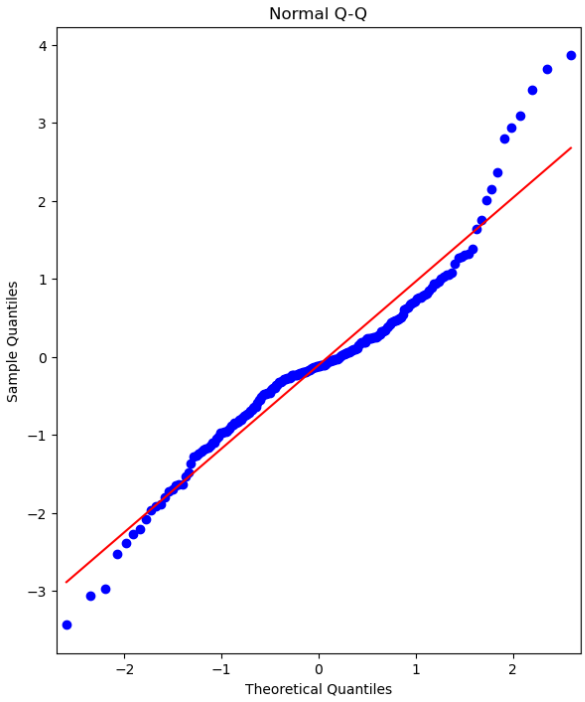

```
The Root Mean Squared Error of our forecasts is 12072.0
```

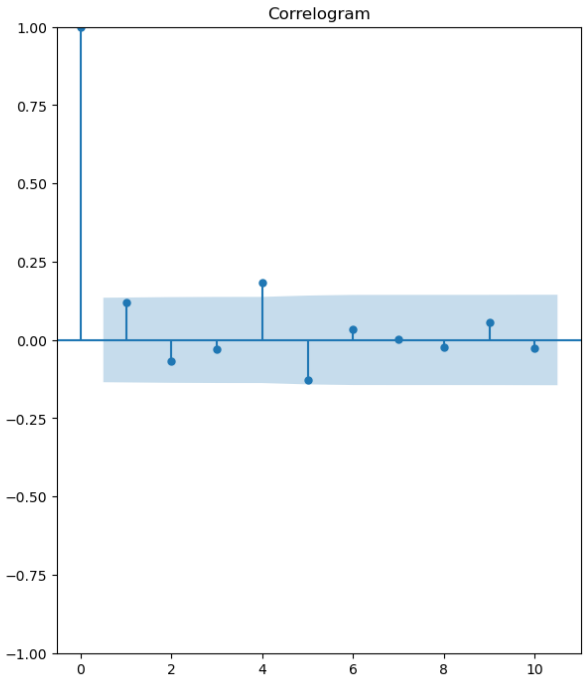
```
['95824']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9656      0.012     82.031      0.000       0.943       0.989
ma.L1          0.6079      0.040     15.376      0.000       0.530       0.685
ar.S.L12       0.1057      0.030      3.530      0.000       0.047       0.164
ma.S.L12      -0.9670      0.112     -8.656      0.000      -1.186      -0.748
```

```
sigma2         2.552e+05    3.45e+04        7.398        0.000     1.88e+05
3.23e+05
==============================================================================
===========
```

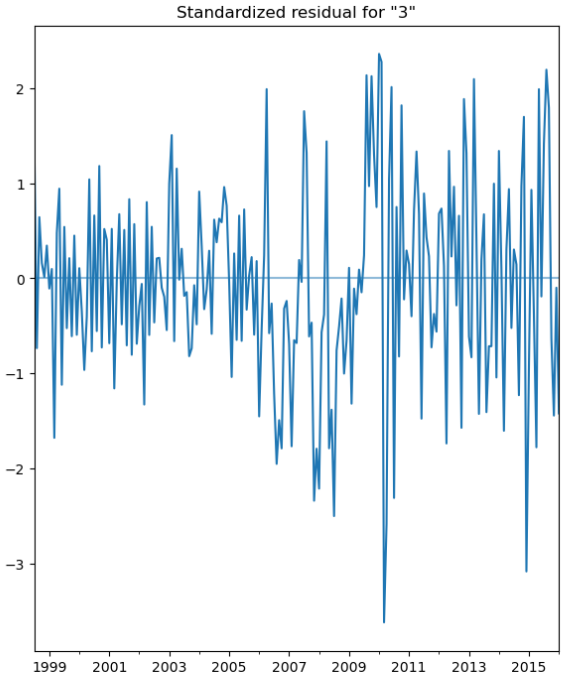


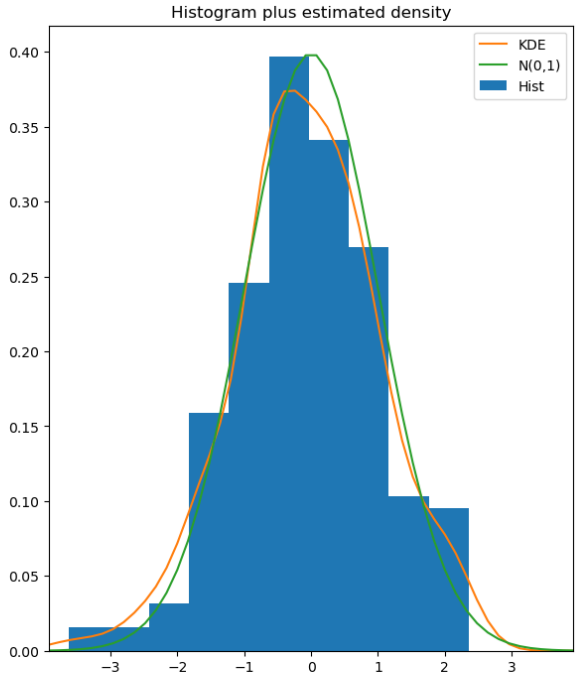


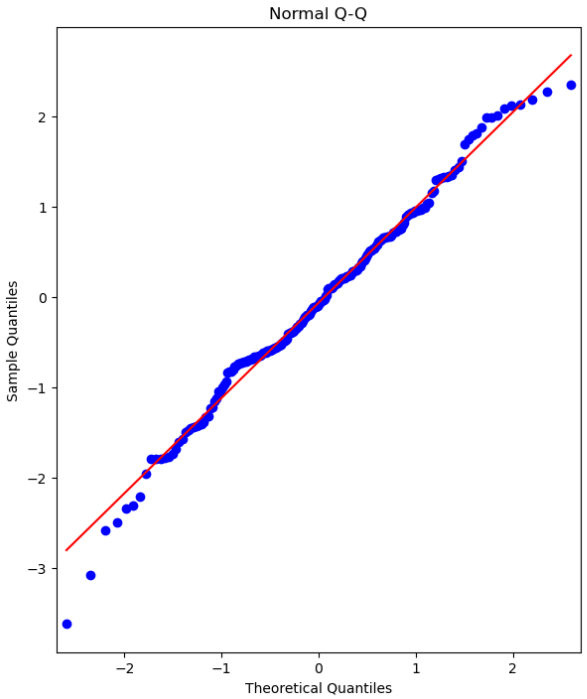


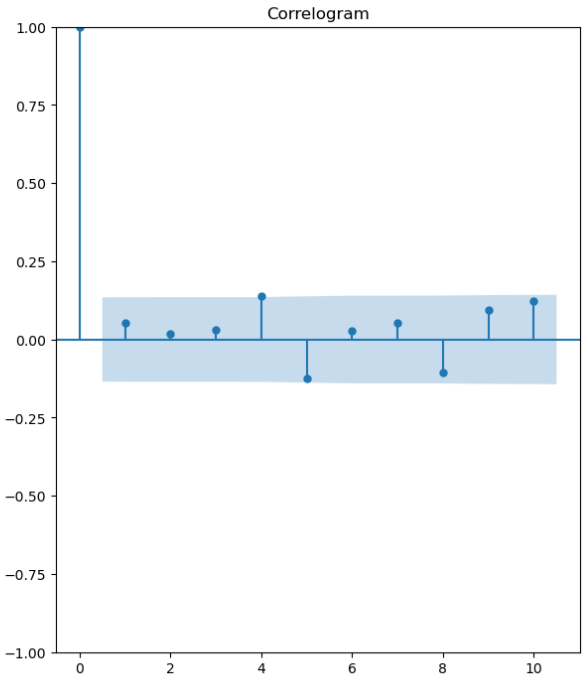


```
The Root Mean Squared Error of our forecasts is 5100.0


['33404']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9743      0.015     65.986      0.000       0.945
1.003
ma.L1          0.5756      0.045     12.882      0.000       0.488
0.663
ma.S.L12      -0.9976      1.736     -0.575      0.566      -4.401
2.406
sigma2      8.182e+04    1.4e+05      0.583      0.560   -1.93e+05
3.57e+05
==============================================================================
```

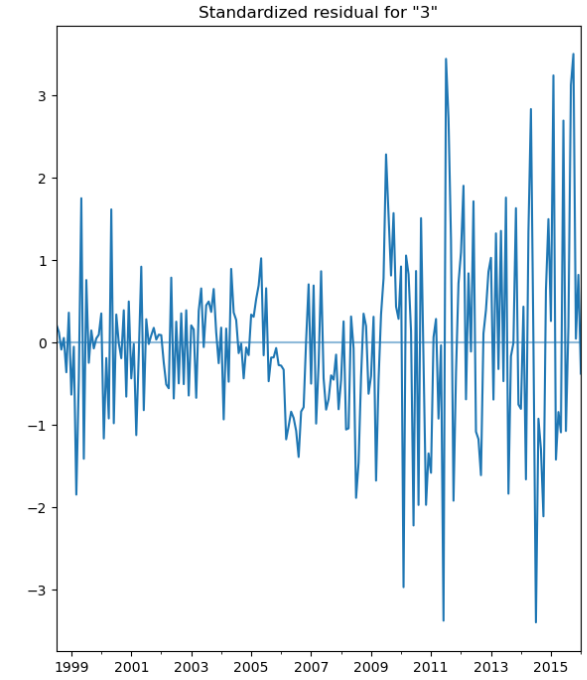

Standardized residual for "3"
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015

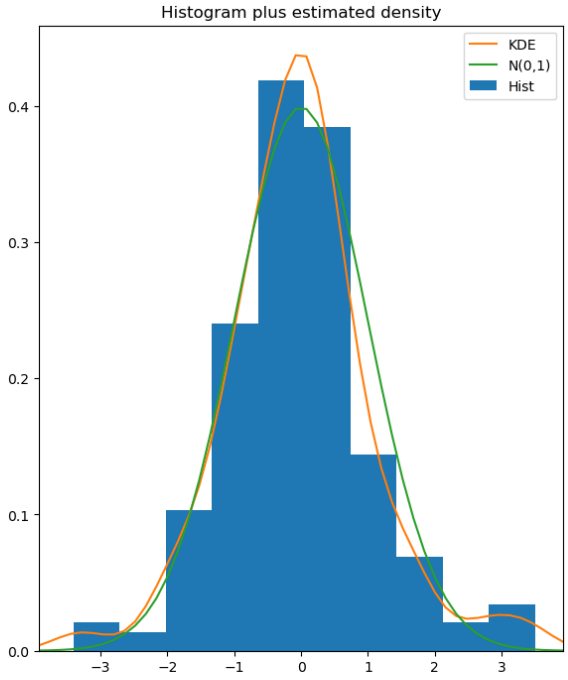

Histogram plus estimated density
KDE
N(0,1)
Hist
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−3
−2
−1
0
1
2
3

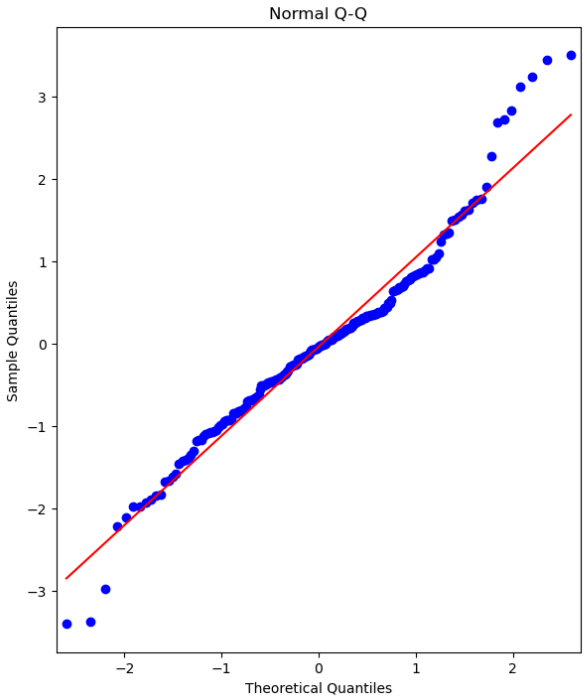

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
2
1
0
−1
−2
−3
−2
−1
0
1
2

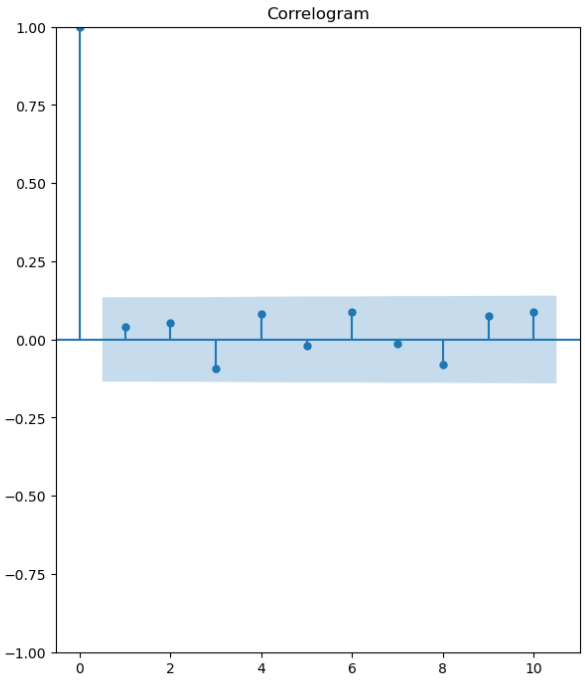

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 20612.0


['33705']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9582      0.020     48.155      0.000       0.919
0.997
ma.L1          0.5818      0.035     16.674      0.000       0.513
0.650
ma.S.L12      -0.9439      0.065    -14.448      0.000      -1.072
-0.816
sigma2      1.483e+05   1.21e+04     12.218      0.000    1.25e+05
1.72e+05
==============================================================================
```

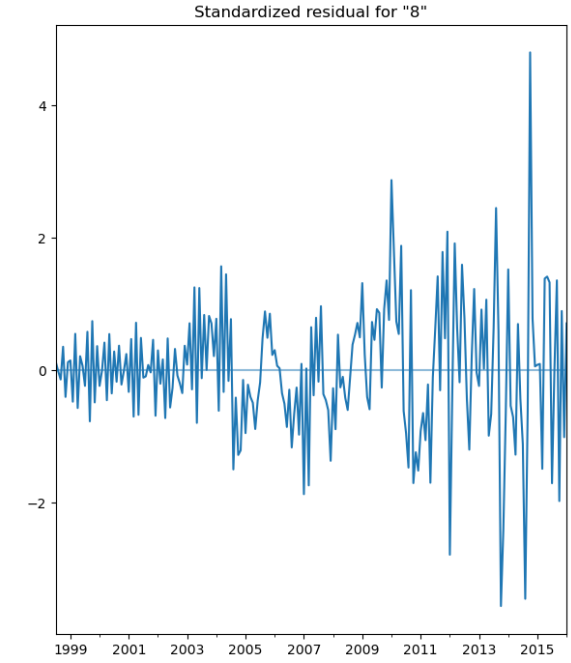

Standardized residual for "3"
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015

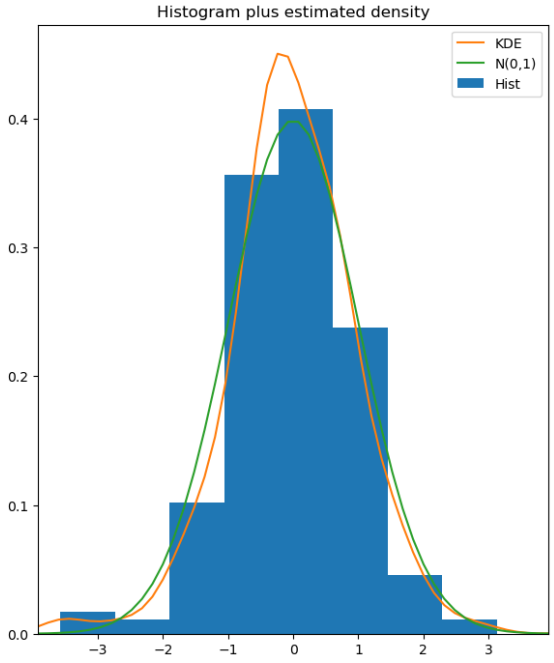

Histogram plus estimated density
KDE
N(0,1)
Hist
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

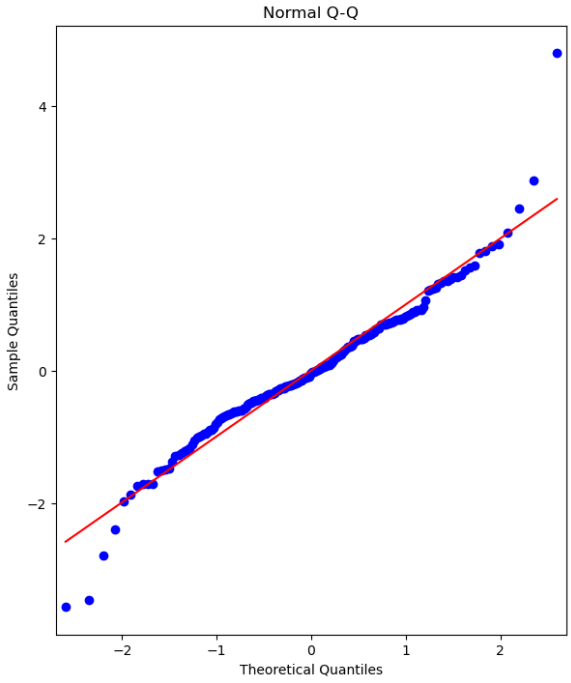

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

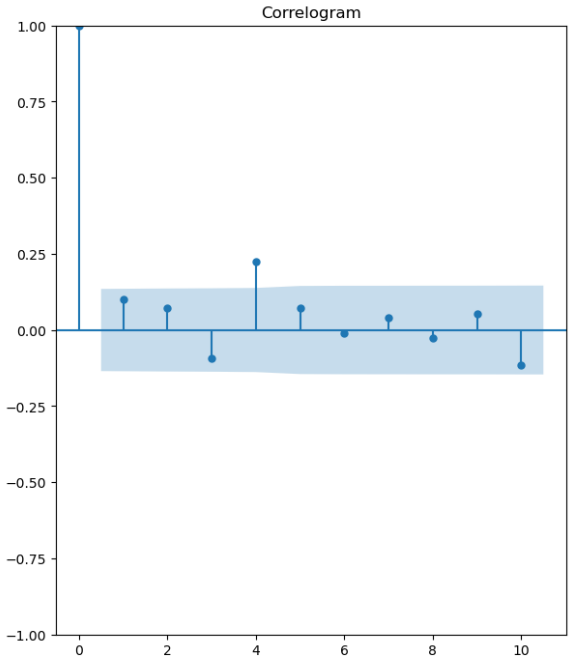

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 19611.0


['89107']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9414      0.020     47.806      0.000       0.903      0.980
ma.L1          0.6926      0.047     14.810      0.000       0.601      0.784
ma.S.L12      32.9930     60.954      0.541      0.588     -86.474    152.460
sigma2       219.4013    820.043      0.268      0.789   -1387.853   1826.655
==============================================================================
```





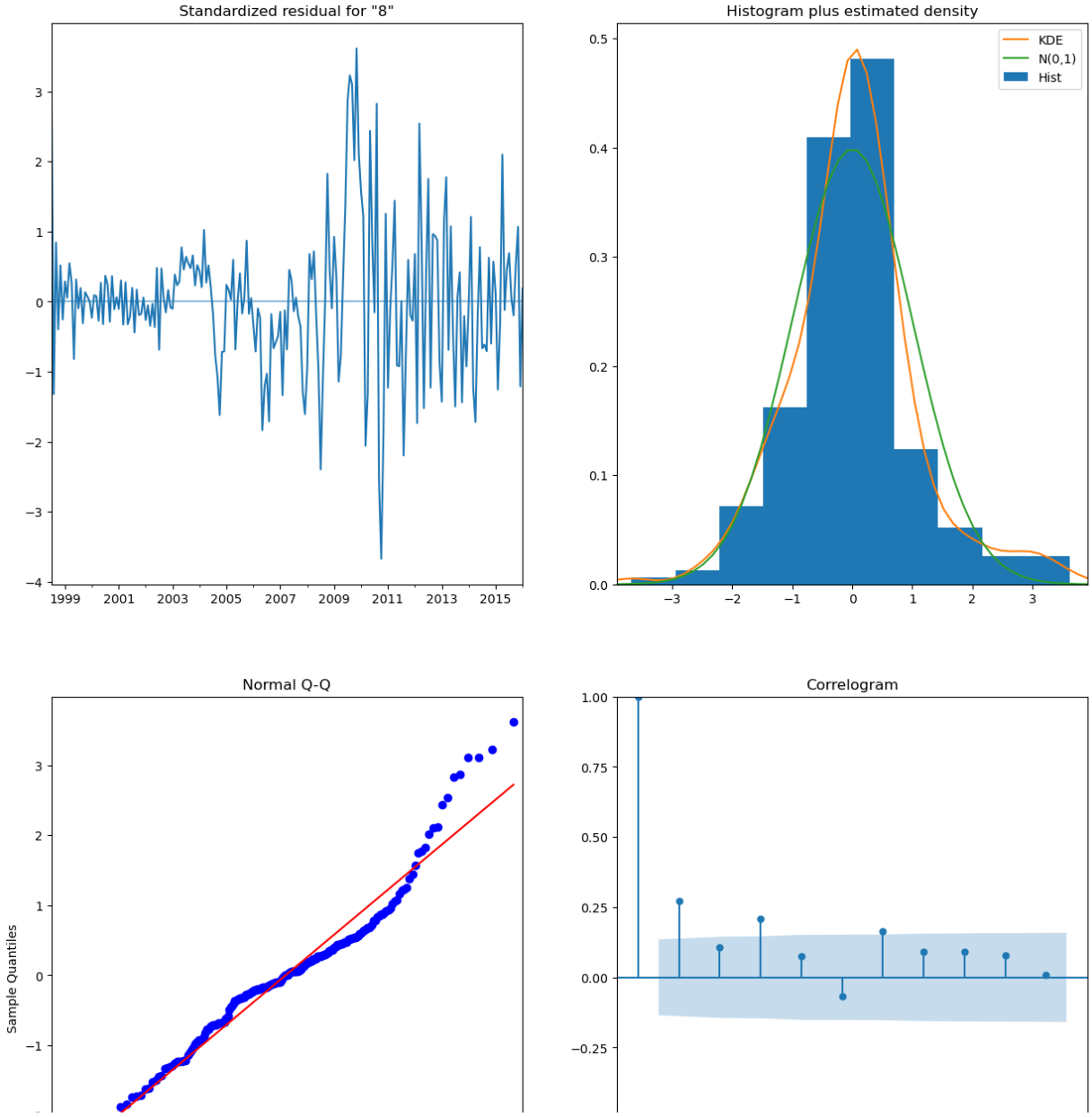





```
The Root Mean Squared Error of our forecasts is 5250.0
```

```
['89115']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
```

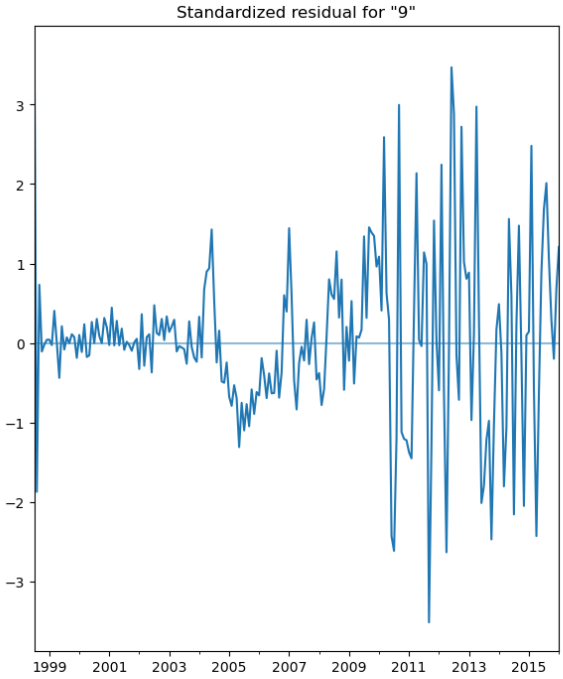
```
------------------------------------------------------------------------------
coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9762      0.012     78.954      0.000       0.952
1.000
ma.L1          0.3515      0.019     18.566      0.000       0.314
0.389
ar.S.L12      -0.4913      0.065     -7.536      0.000      -0.619
-0.364
ma.S.L12      19.4042     17.505      1.108      0.268     -14.905
53.714
sigma2       445.5423    807.393      0.552      0.581   -1136.920
2028.004
==============================================================================
```

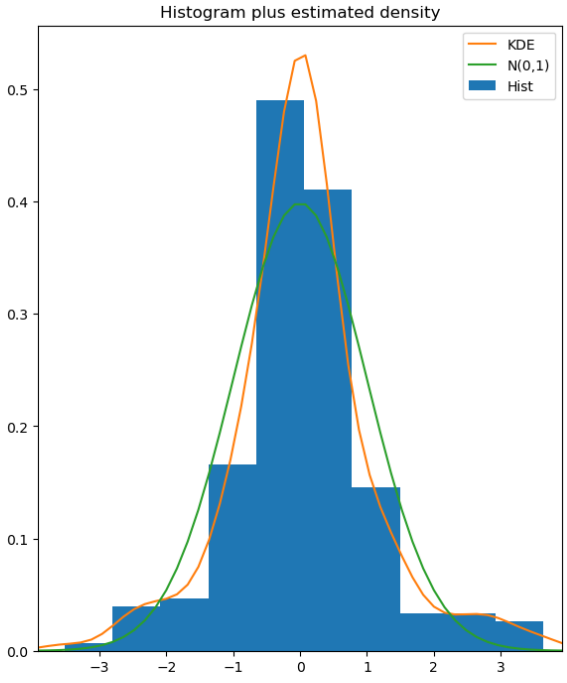


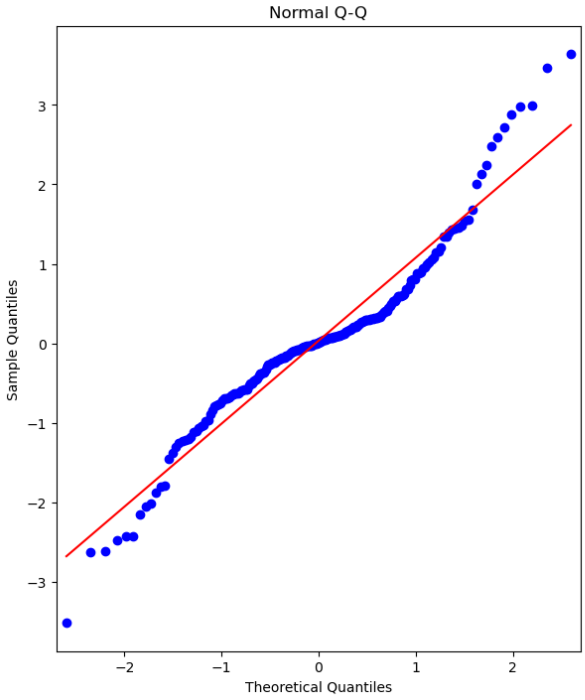

```
The Root Mean Squared Error of our forecasts is 3654.0
```

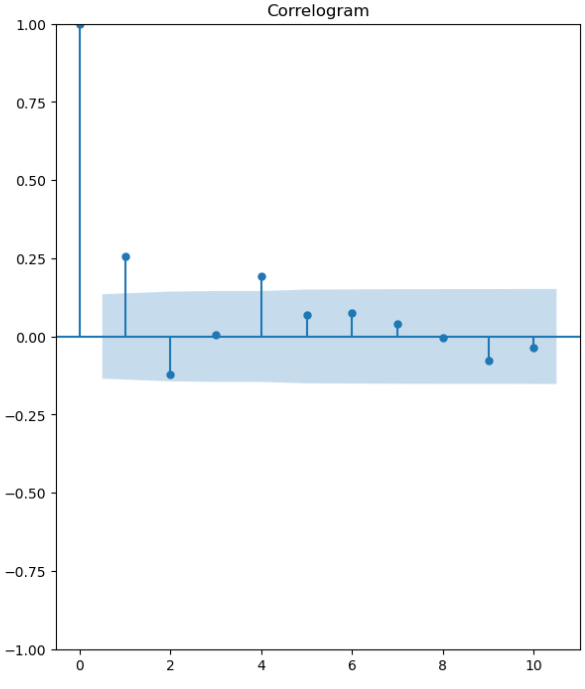
```
['95820']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9642      0.017     55.430      0.000       0.930
0.998
ma.L1          0.4264      0.019     22.201      0.000       0.389
0.464
ar.S.L12      -0.5107      0.073     -6.988      0.000      -0.654
-0.367
ma.S.L12      21.1874     38.151      0.555      0.579     -53.588
95.963
```

```
sigma2         650.8028    2366.818        0.275        0.783    -3988.075
5289.681
================================================================================
=========
```

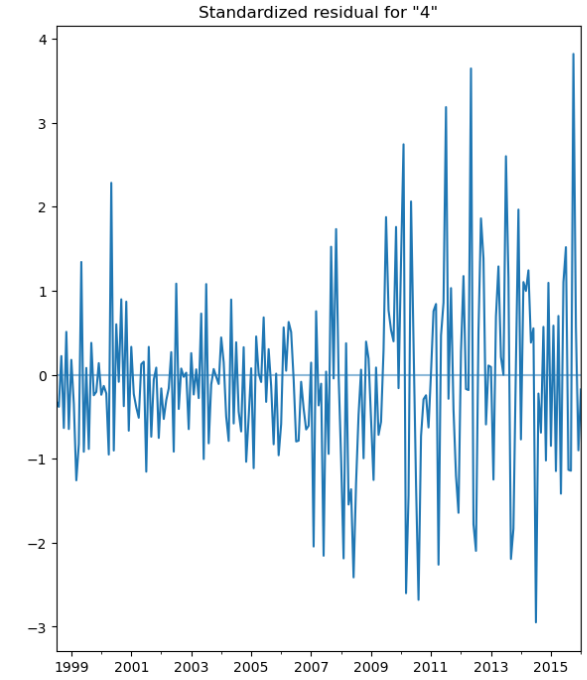


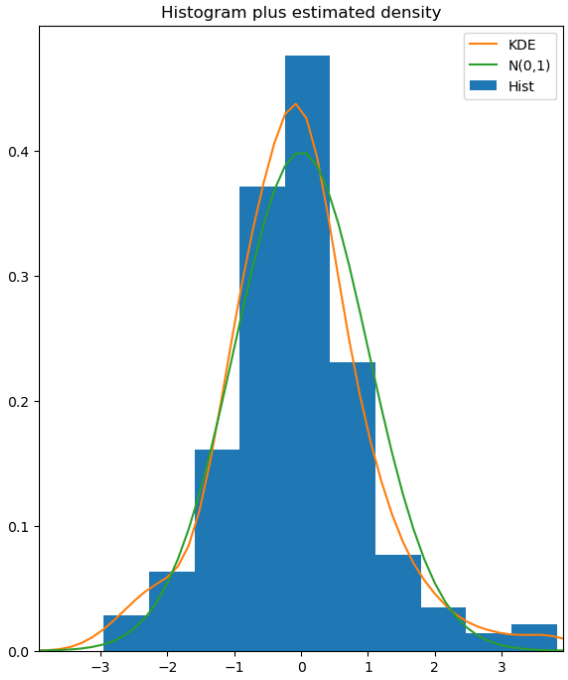


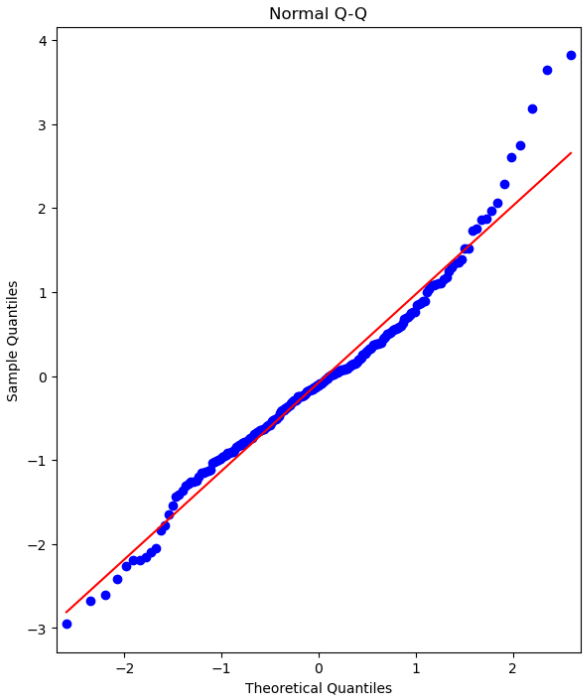


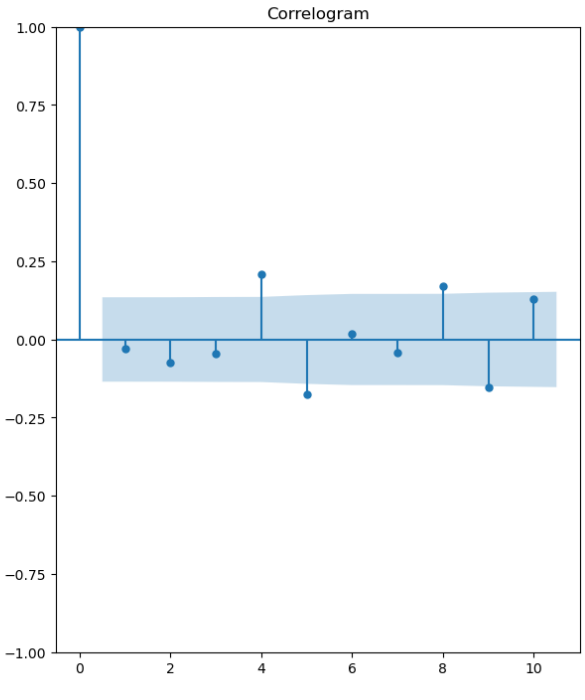


```
The Root Mean Squared Error of our forecasts is 19365.0


['48240']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9596      0.021     45.757      0.000       0.918
1.001
ma.L1          0.4983      0.053      9.352      0.000       0.394
0.603
ma.S.L12      -0.9980      1.989     -0.502      0.616      -4.896
2.900
sigma2      4.774e+04   9.38e+04      0.509      0.611   -1.36e+05
2.31e+05
==============================================================================
```

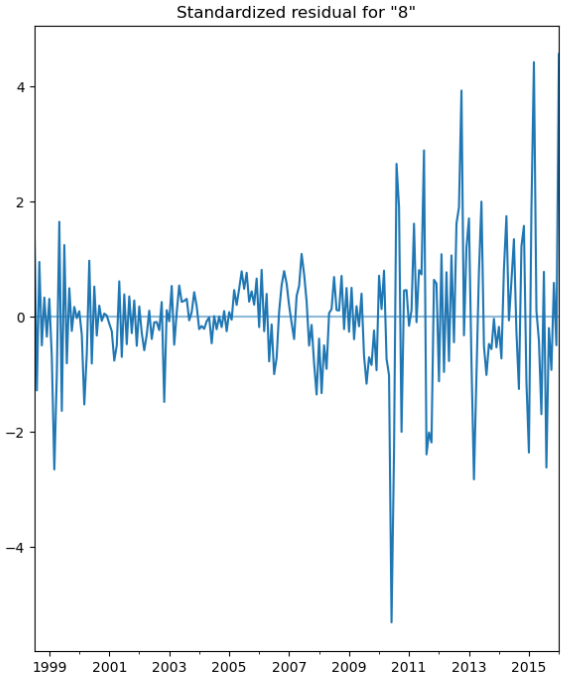

Standardized residual for "4"
4
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015

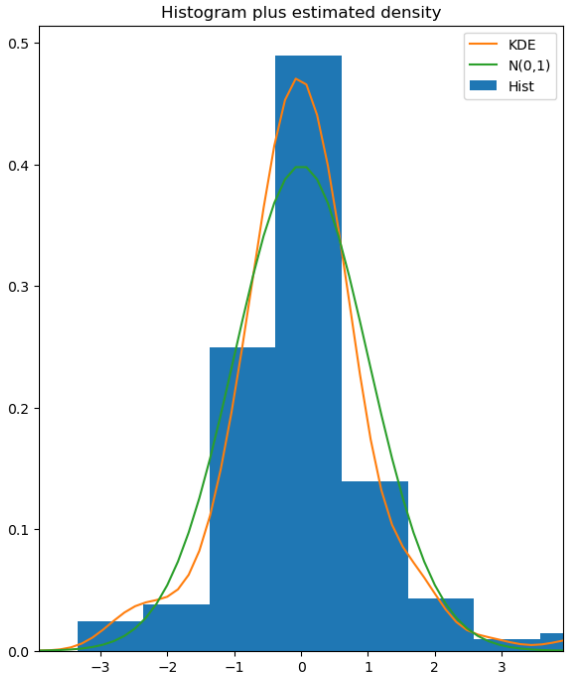

Histogram plus estimated density
KDE
N(0,1)
Hist
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

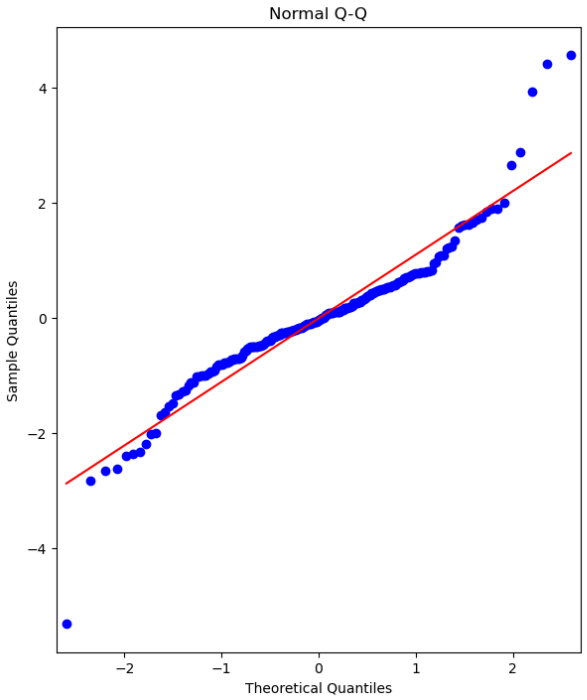

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

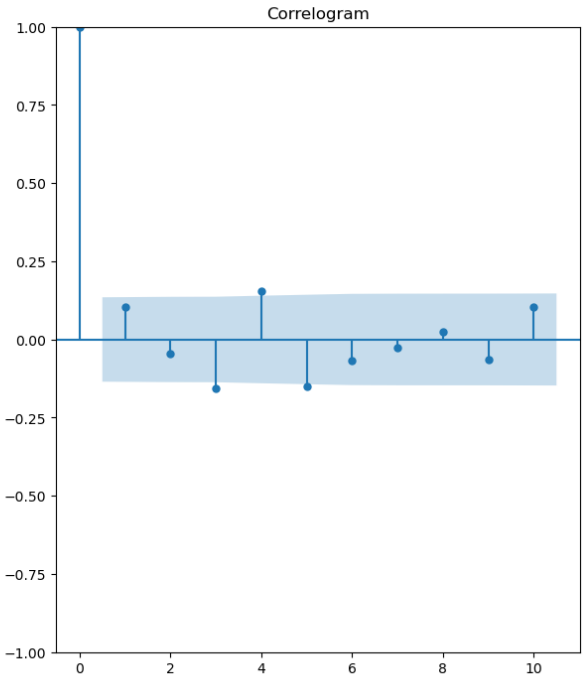

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 2626.0


['84101']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7415      0.029     25.721      0.000       0.685      0.798
ma.L1          0.7009      0.033     21.038      0.000       0.636      0.766
ma.S.L12      -0.5911      0.037    -16.026      0.000      -0.663     -0.519
sigma2      4.191e+05   2.19e+04     19.124      0.000    3.76e+05   4.62e+05
==============================================================================
```

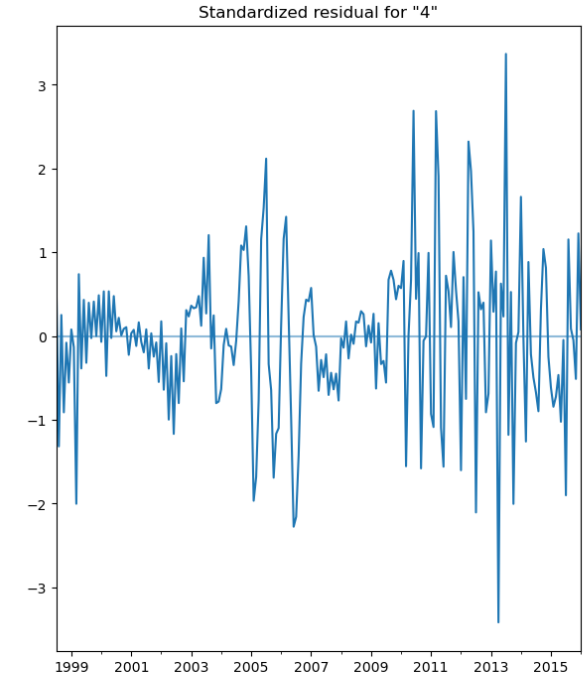

Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015

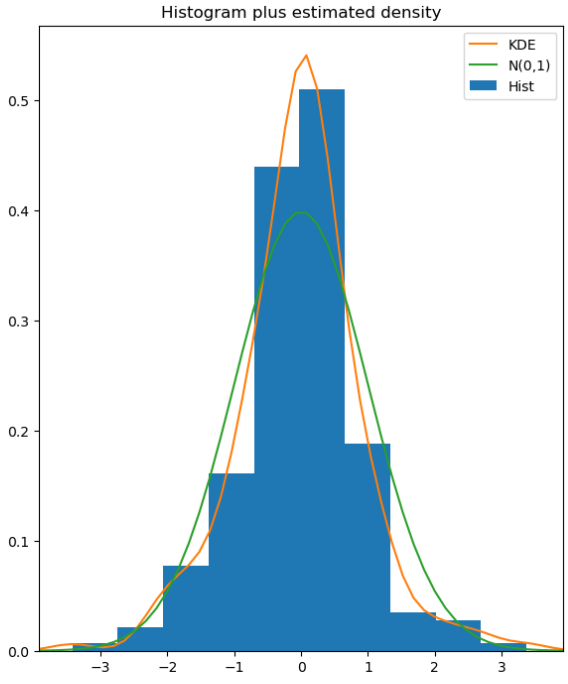

Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

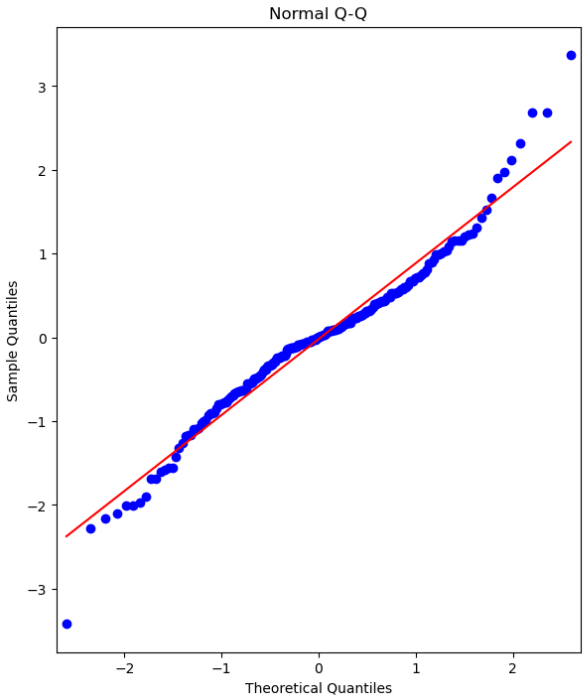

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2

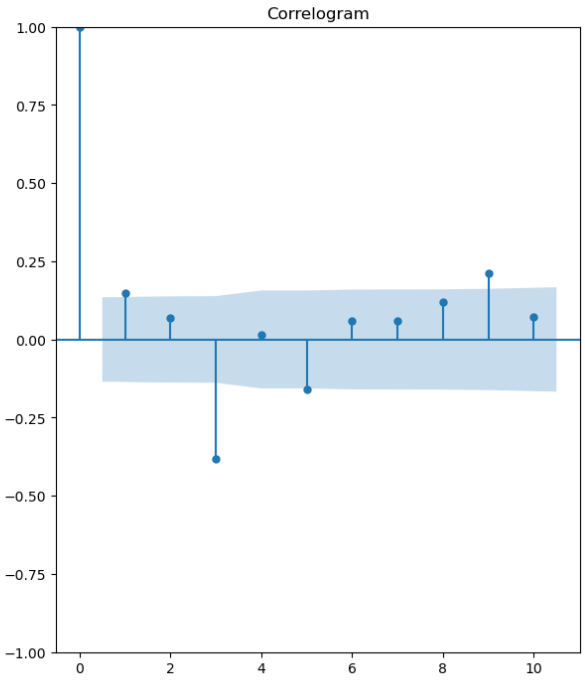

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 24559.0


['48009']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8114      0.047     17.207      0.000       0.719      0.904
ma.L1          1.1354      0.042     26.998      0.000       1.053      1.218
ar.S.L12       0.0805      0.013      6.419      0.000       0.056      0.105
ma.S.L12      -0.2772      0.031     -9.055      0.000      -0.337     -0.217
sigma2      1.275e+06    1.6e+05      7.959      0.000    9.61e+05    1.59e+06
==============================================================================
```



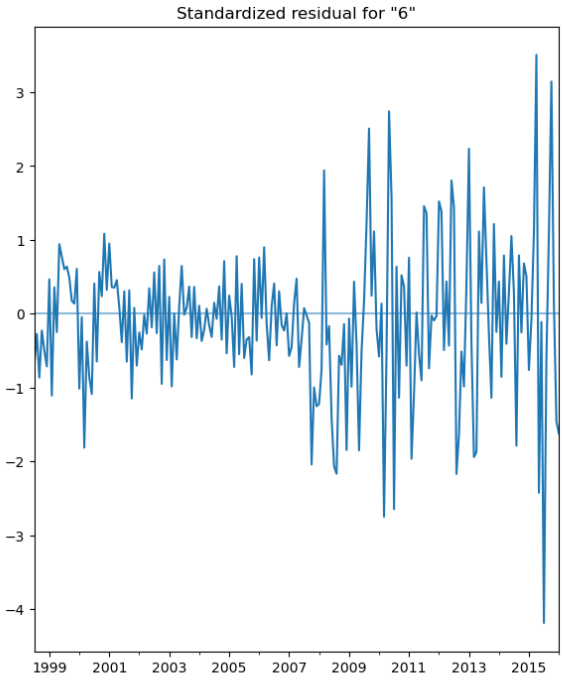


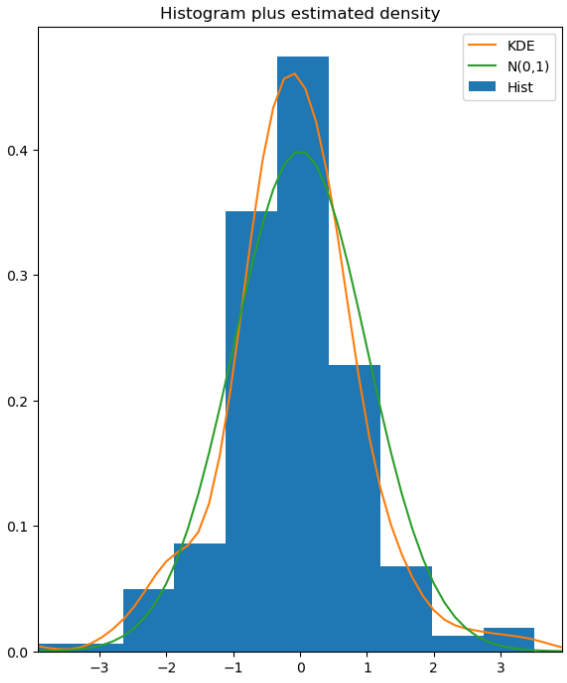


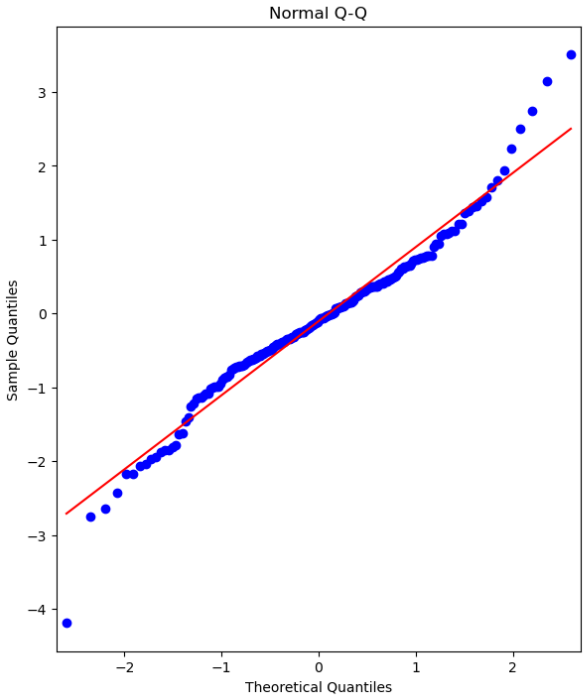


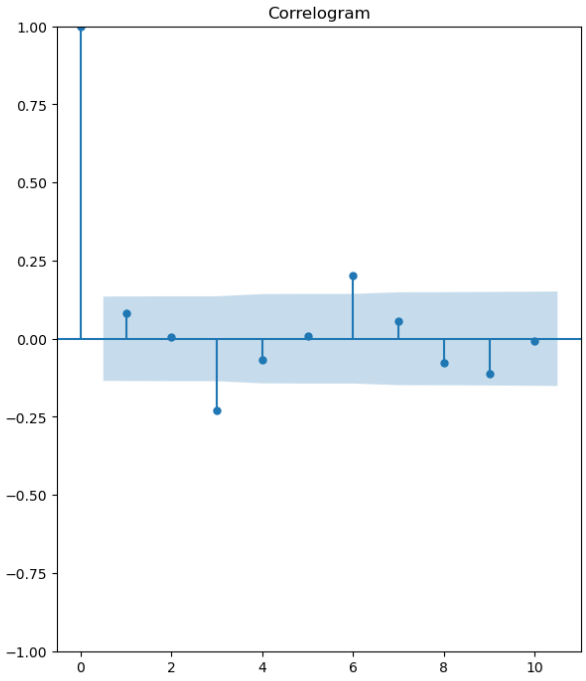

```
The Root Mean Squared Error of our forecasts is 22105.0


['66104']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
```

```
------------------------------------------------------------------------------
ar.L1          0.8911      0.029     30.434      0.000       0.834       0.948
ma.L1          0.6211      0.050     12.362      0.000       0.523       0.720
ma.S.L12      -0.9982      2.608     -0.383      0.702      -6.110       4.114
sigma2       5.87e+04   1.53e+05      0.385      0.701   -2.41e+05    3.58e+05
==============================================================================
```



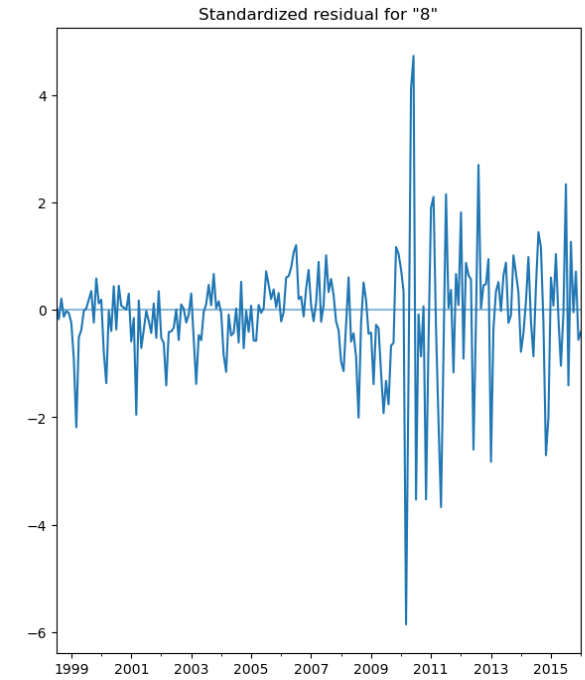


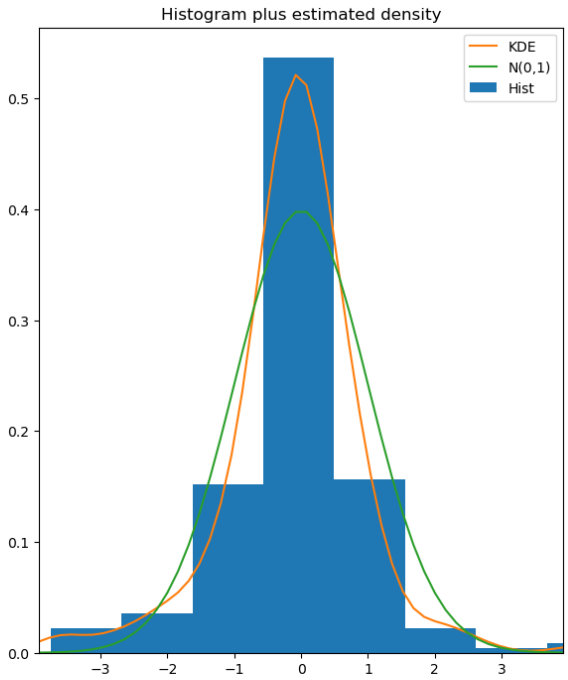


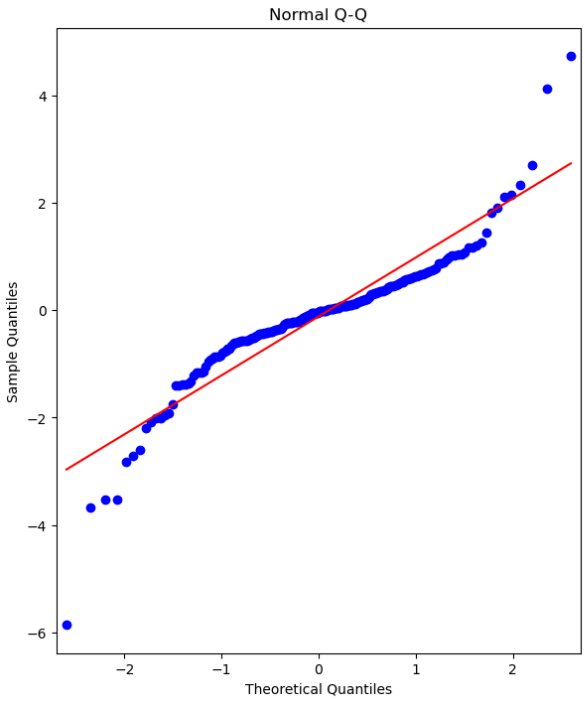


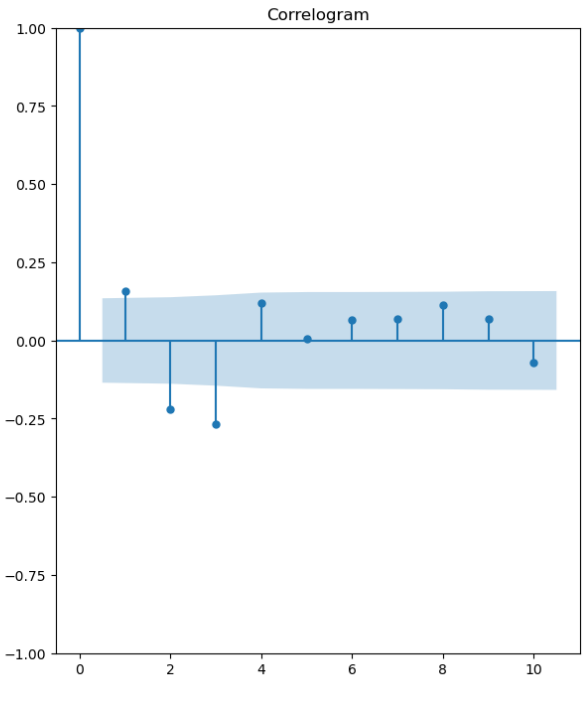
```
The Root Mean Squared Error of our forecasts is 17671.0


['84606']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8445      0.033     25.855      0.000       0.781
0.909
ma.L1          0.3847      0.041      9.321      0.000       0.304
0.466
ar.S.L12       0.0900      0.016      5.778      0.000       0.059
0.121
ma.S.L12      -0.9551      0.118     -8.128      0.000      -1.185
-0.725
sigma2      1.611e+05   1.27e+04     12.648      0.000    1.36e+05
```

```
1.86e+05
==============================================================================
=========
```





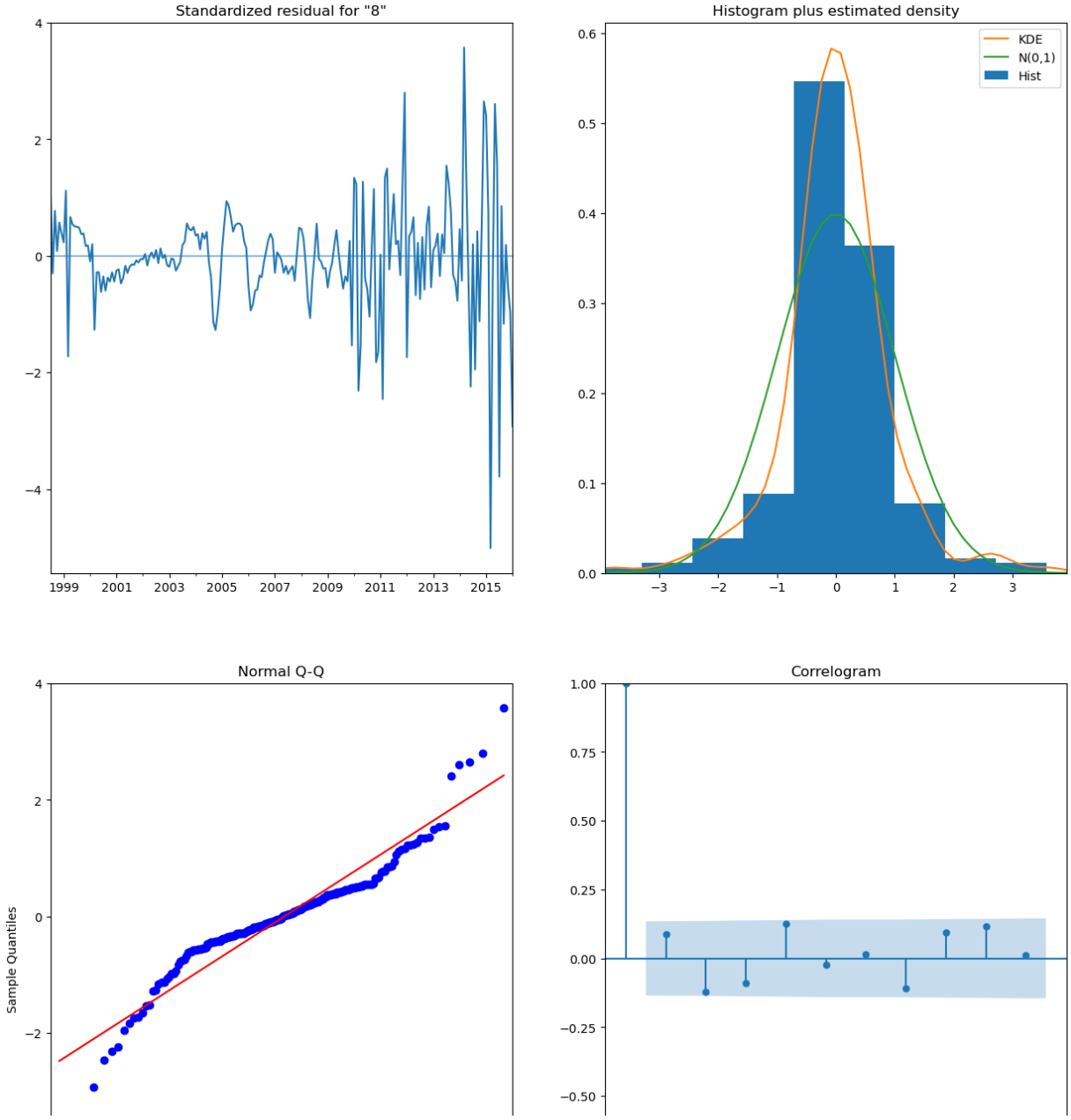





```
The Root Mean Squared Error of our forecasts is 31924.0
```

```
['87506']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
=========
                 coef    std err          z      P>|z|      [0.025
```

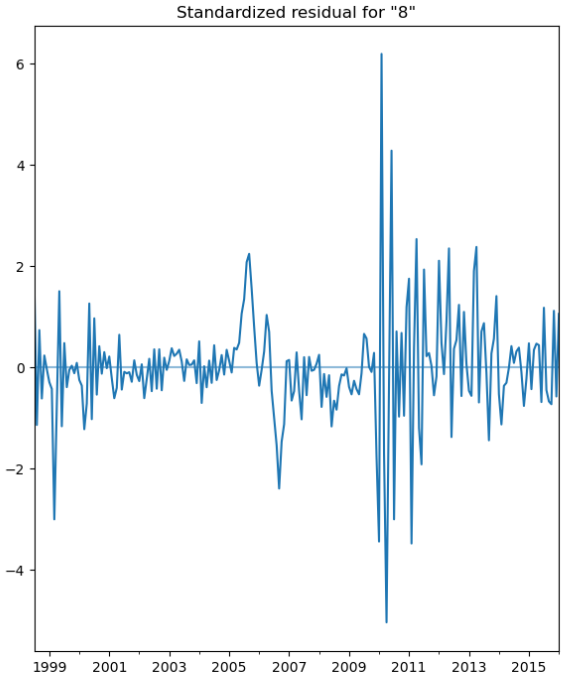
```
0.975]
------------------------------------------------------------------------------
---------
ar.L1          0.7522      0.041     18.431      0.000        0.672
0.832
ma.L1          0.7636      0.042     18.162      0.000        0.681
0.846
ar.S.L12       0.2102      0.040      5.236      0.000        0.132
0.289
ma.S.L12      -0.3916      0.046     -8.568      0.000       -0.481
-0.302
sigma2      5.411e+06   3.93e+05     13.767      0.000    4.64e+06
6.18e+06
==============================================================================
=========
```

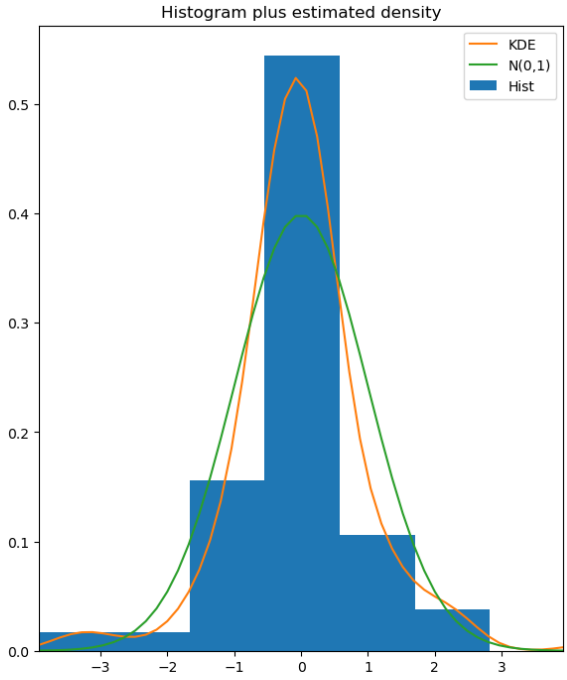


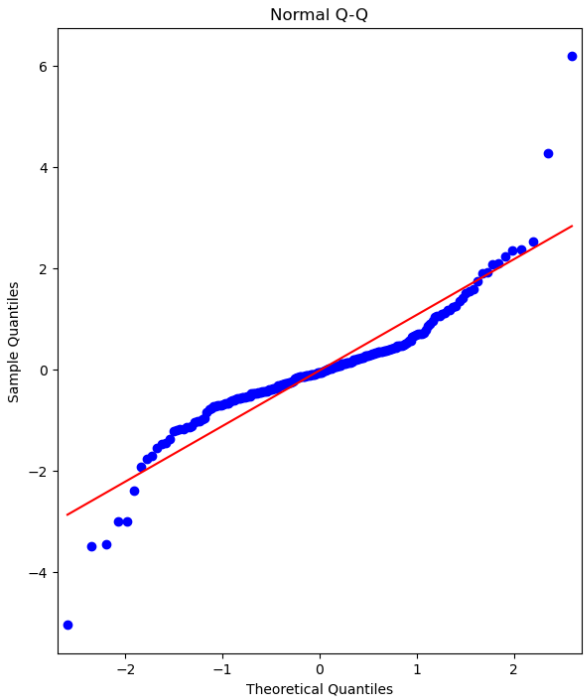

```
The Root Mean Squared Error of our forecasts is 137358.0
```

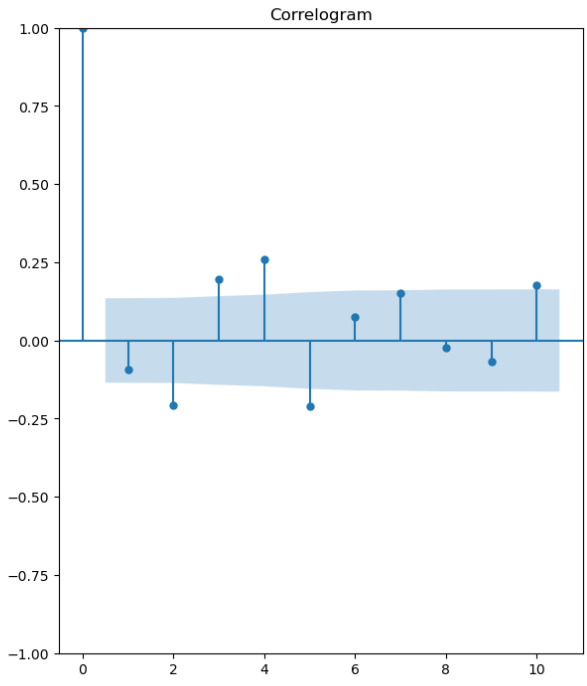
```
['83703']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.8095      0.033     24.782      0.000       0.746
0.874
ma.L1          0.5904      0.026     22.479      0.000       0.539
0.642
ma.S.L12      -0.5333      0.027    -19.633      0.000      -0.587
-0.480
sigma2      3.998e+05   2.04e+04     19.581      0.000     3.6e+05
```

```
4.4e+05
================================================================================
```


Standardized residual for "8"
6
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015


Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3


Normal Q-Q
Sample Quantiles
Theoretical Quantiles
6
4
2
0
−2
−4
−2
−1
0
1
2


Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

```
The Root Mean Squared Error of our forecasts is 21177.0
```

```
[58117.0, 9986.0, 2775.0, 20071.0, 44237.0, 12709.0, 12072.0, 5100.0,
20612.0, 19611.0, 5250.0, 3654.0, 19365.0, 2626.0, 24559.0, 22105.0,
17671.0, 31924.0, 137358.0, 21177.0]
```

6.2 RMSE and RMSE Ratio to Median Value

In [14]:

```
1 rmse_list
```

executed in 20ms, finished 10:55:31 2022-05-03

Out[14]:

```
[58117.0,
 9986.0,
 2775.0,
 20071.0,
 44237.0,
 12709.0,
 12072.0,
 5100.0,
 20612.0,
 19611.0,
 5250.0,
 3654.0,
 19365.0,
 2626.0,
 24559.0,
 22105.0,
 17671.0,
 31924.0,
 137358.0,
 21177.0]
```

In [15]:

```
prospects['RMSE'] = rmse_list
prospects['RMSE_Ratio'] = prospects['RMSE']/prospects['2018-04']
#top_prospects = prospects.loc[[33460,94606,4298,815,5830]]
prospects
```

executed in 60ms, finished 10:57:06 2022-05-03

Out[15]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05	1996-06	199
1477	94601	Oakland	CA	San Francisco	Alameda	1478	114600.0	114500.0	114500.0	1144
1239	94590	Vallejo	CA	Vallejo	Solano	1240	108200.0	107800.0	107500.0	1073
2627	33460	Lake Worth	FL	Miami-Fort Lauderdale	Palm Beach	2628	59800.0	59900.0	60000.0	602
1681	94606	Oakland	CA	San Francisco	Alameda	1682	120400.0	120300.0	120300.0	1202
1853	94804	Richmond	CA	San Francisco	Contra Costa	1854	171300.0	170300.0	169400.0	1685
1960	89104	Las Vegas	NV	Las Vegas	Clark	1961	94800.0	94700.0	94700.0	947
474	85008	Phoenix	AZ	Phoenix	Maricopa	475	61600.0	61900.0	62200.0	626

6.3 Create top_prospects df

- Zip Codes Included:
  - Redford, MI 48240
  - Las Vegas, NV 89115
  - Sacramento, CA 95824
  - Oakland, CA 94606
  - Lake Worth, FL 33460

In [44]:

```
top_prospects = prospects.loc[[2627,1681,4298,815,5830,5764]]
```

executed in 39ms, finished 13:43:28 2022-05-03

In [45]:

```
# drop RMSE and RMSE_Ratio colums
top_prospects = top_prospects.drop(columns=['RMSE','RMSE_Ratio'])
top_prospects
```

executed in 69ms, finished 13:43:29 2022-05-03

Out[45]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05
2627	33460	Lake Worth	FL	Miami-Fort Lauderdale	Palm Beach	2628	59800.0	59900.0
1681	94606	Oakland	CA	San Francisco	Alameda	1682	120400.0	120300.0
4298	95824	Sacramento	CA	Sacramento	Sacramento	4299	73800.0	73400.0
815	89115	Las Vegas	NV	Las Vegas	Clark	816	92500.0	92500.0
5830	48240	Redford	MI	Detroit	Wayne	5831	67800.0	68200.0
5764	83703	Boise	ID	Boise City	Ada	5765	107500.0	107500.0

6 rows × 271 columns

7 Price Forecast - Top Zip Codes

- 5 year investment horizon

In [74]:

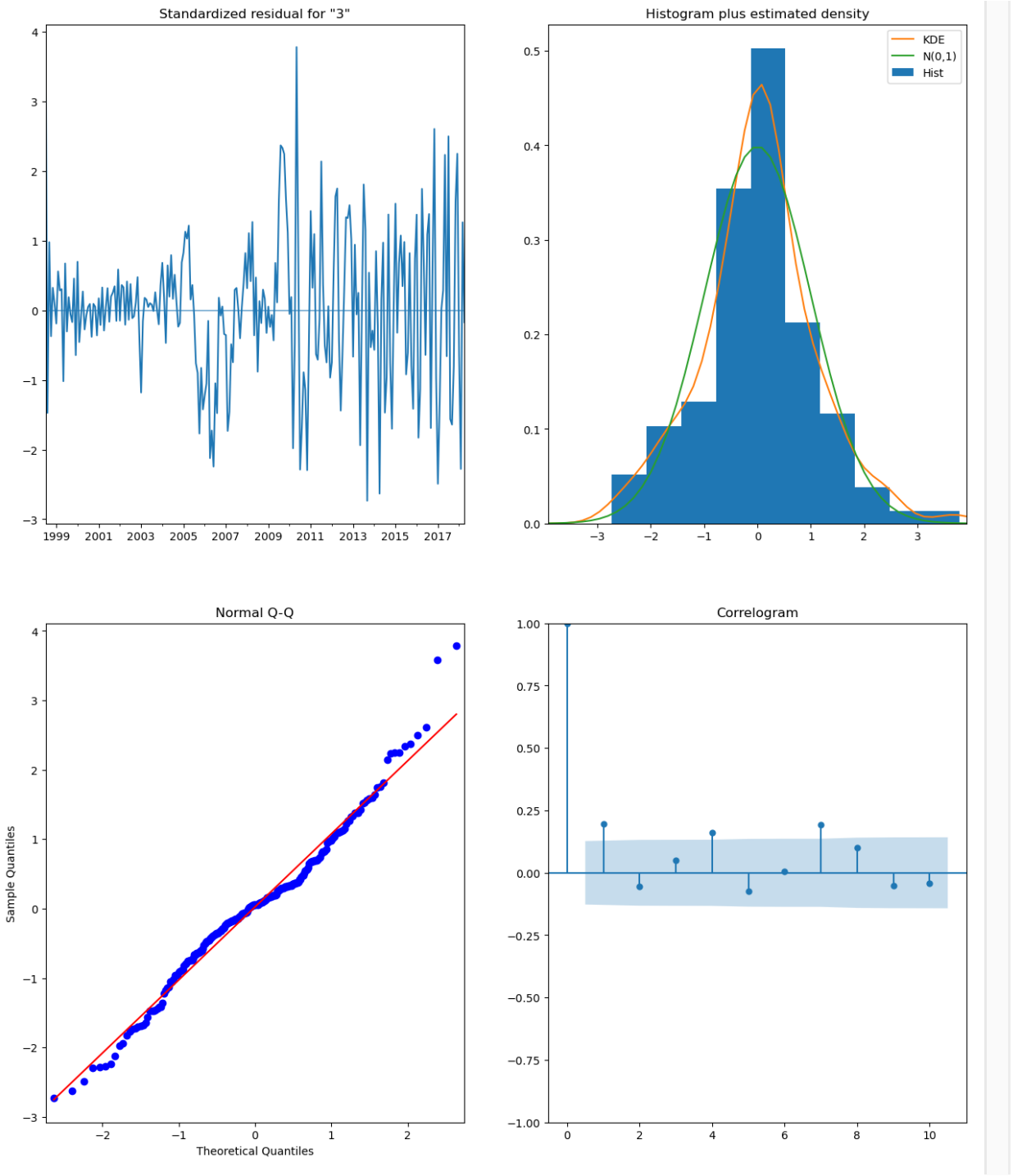
```
# Create list for final predicted value
pred_list_1 = []

# Construct model from entirety of data - no train/test
# iterate over prospects to extract optimal SARIMA values,
# instantiate SARIMAX model, make predictions
# and produce forecast with confidence intervals
for zipcode in top_prospects['ZipCode']:
    zips = top_prospects.loc[top_prospects['ZipCode'] == zipcode]
    m1 = melt_data(zips)
    m1[str(zipcode)] = m1['value']
    m1 = m1.drop('value', axis=1)

    # do NOT create train and test sets
    #train = m1.iloc[:238]
    #test = m1.iloc[238:]

    # Run a grid with pdq and seasonal pdq parameters calculated above and ge
    ans = []
    outputs = []
    comb_list = []
    combs_list = []

    for comb in pdq:
        for combs in pdqs:
            try:
                mod = sm.tsa.statespace.SARIMAX(m1,
                                                order=comb,
                                                seasonal_order=combs,
                                                enforce_stationarity=False,
                                                enforce_invertibility=False)

                output = mod.fit()
                ans.append([comb, combs, output.aic])
                outputs.append([output.aic])
                comb_list.append([comb])
                combs_list.append([combs])
                AIC_df = pd.DataFrame(list(zip(outputs, comb_list, combs_list
                #print('ARIMA {} x {}12 : AIC Calculated ={}'.format(comb, co
            except:
                continue

    # Print the lowest AIC and it's parameters for SARIMA
    AIC_df[0] = AIC_df[0].str.get(0)
    best_AIC = AIC_df.copy()
    lowest_AIC = min(best_AIC[0])
    lowest_AIC_row = AIC_df.loc[AIC_df[0] == lowest_AIC]
    #print(lowest_AIC_row)

    # call value for comb and combs for model input
    lowest_AIC_row[1] = lowest_AIC_row[1].str.get(0)
    lowest_AIC_row[2] = lowest_AIC_row[2].str.get(0)
```

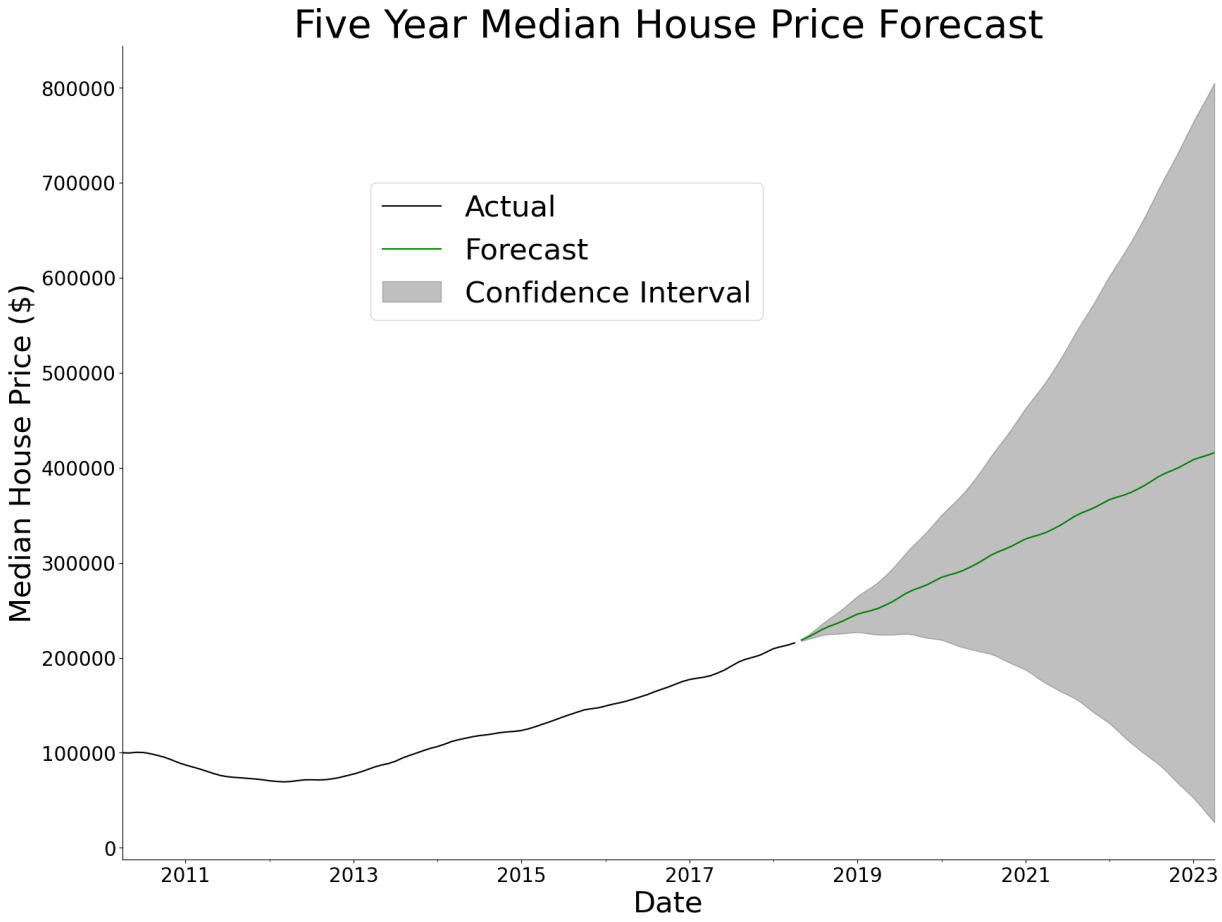
```
    arima = lowest_AIC_row.iat[0,1]
    s = lowest_AIC_row.iat[0,2]
    print(str(m1.columns.values))
    print(arima, s)

    # Plug the optimal parameter values into a new SARIMAX model
    ARIMA_MODEL = sm.tsa.statespace.SARIMAX(m1,
                                            order=arima,
                                            seasonal_order=s,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)

    # Fit the model and print results
    output = ARIMA_MODEL.fit()

    print(output.summary().tables[1])

    # Call plot_diagnostics() on the results calculated above
    output.plot_diagnostics(figsize=(15, 18))
    plt.show()

    # Get actual results in dataframe
    price_truth = m1['2018-04-01':]

    # Get forecast 500 steps ahead in future
    prediction = output.get_forecast(steps=60)

    # Create predictions df
    pred = prediction.predicted_mean

    # Add prediction for 5 year forecasted value to list
    pred_list_1.append(pred['2023-04-01'])

    # Get confidence intervals of forecasts
    pred_conf = prediction.conf_int()

    # Only use data from April 2010 to improve visual of forecast
    plot_vals = m1.loc['2010-04-01':]

    # Plot future predictions with confidence intervals
    ax = plot_vals.plot(label='observed', figsize=(20, 15), color='black')
    prediction.predicted_mean.plot(ax=ax, label='Forecast', color='green')
    ax.fill_between(pred_conf.index,
                    pred_conf.iloc[:, 0],
                    pred_conf.iloc[:, 1], color='k', alpha=0.25)
    ax.set_xlabel('Date').set_fontsize(30)
    ax.set_ylabel('Median House Price ($)').set_fontsize(30)
    ax.set_title('Five Year Median House Price Forecast').set_fontsize(40)

    for item in (ax.get_xticklabels() + ax.get_yticklabels()):
        item.set_fontsize(20)

    right_side = ax.spines["right"]
    right_side.set_visible(False)
```

```
107
108     top_side = ax.spines["top"]
109     top_side.set_visible(False)
110
111     mylabels = 'Actual', 'Forecast', 'Confidence Interval'
112     plt.legend(fontsize=30,labels=mylabels,bbox_to_anchor=(0.6,0.85))
113     plt.show()
114     print()
115     print()
116
117
```

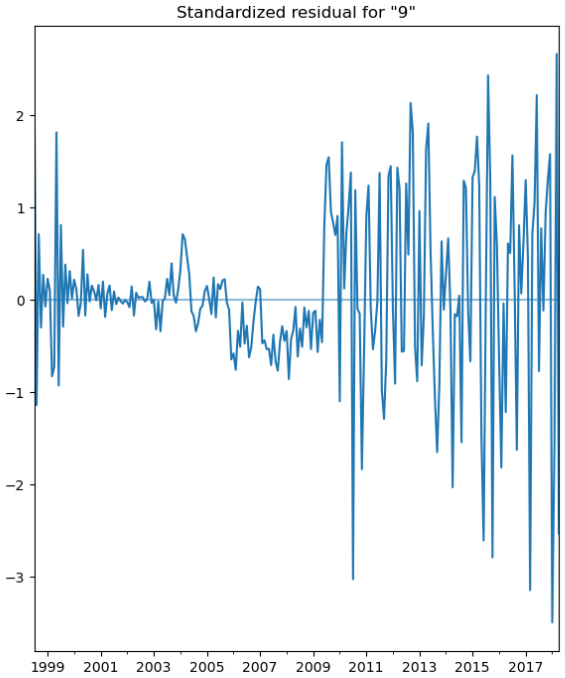
executed in 3m 10s, finished 14:01:19 2022-05-03

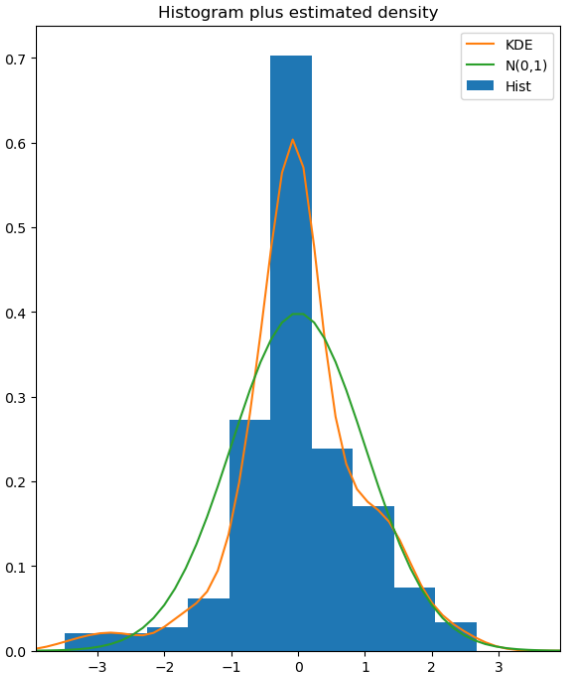
```
['33460']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9646      0.017     57.776      0.000       0.932
0.997
ma.L1          0.5143      0.022     23.638      0.000       0.472
0.557
ar.S.L12      -0.5424      0.065     -8.383      0.000      -0.669
-0.416
ma.S.L12      14.5237     12.200      1.190      0.234      -9.389
38.436
sigma2       925.0061   1559.321      0.593      0.553   -2131.207
3981.219
==============================================================================
```

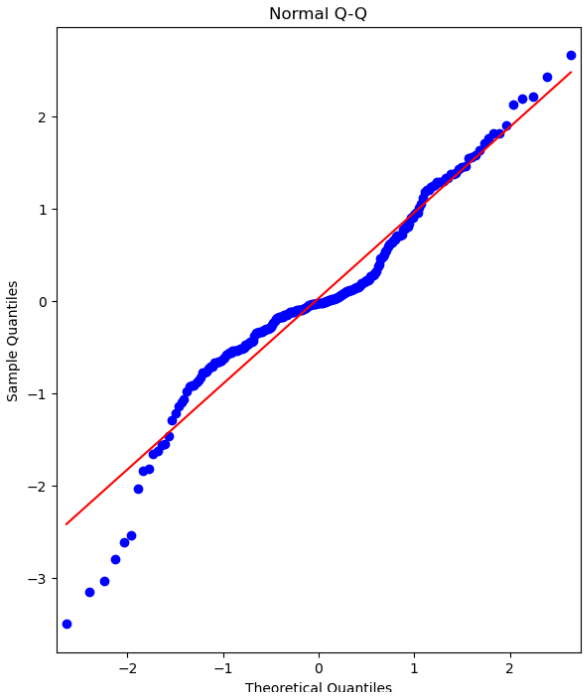

Standardized residual for "3"
Histogram plus estimated density
KDE
N(0,1)
Hist
Normal Q-Q
Sample Quantiles
Theoretical Quantiles
Correlogram

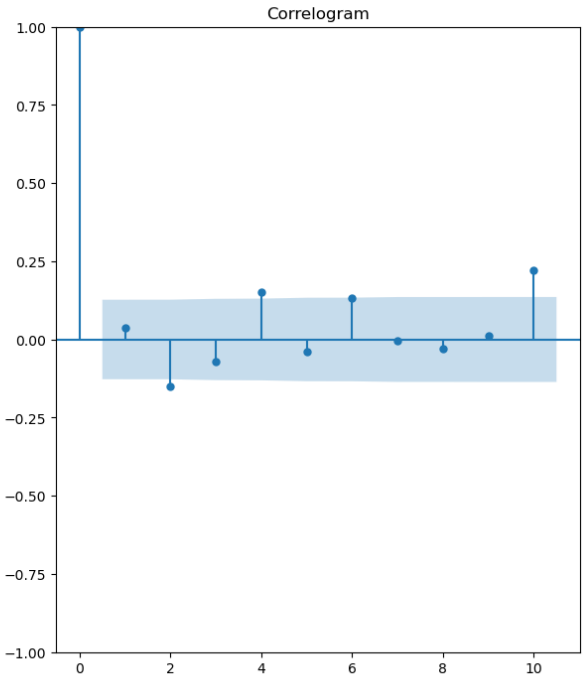


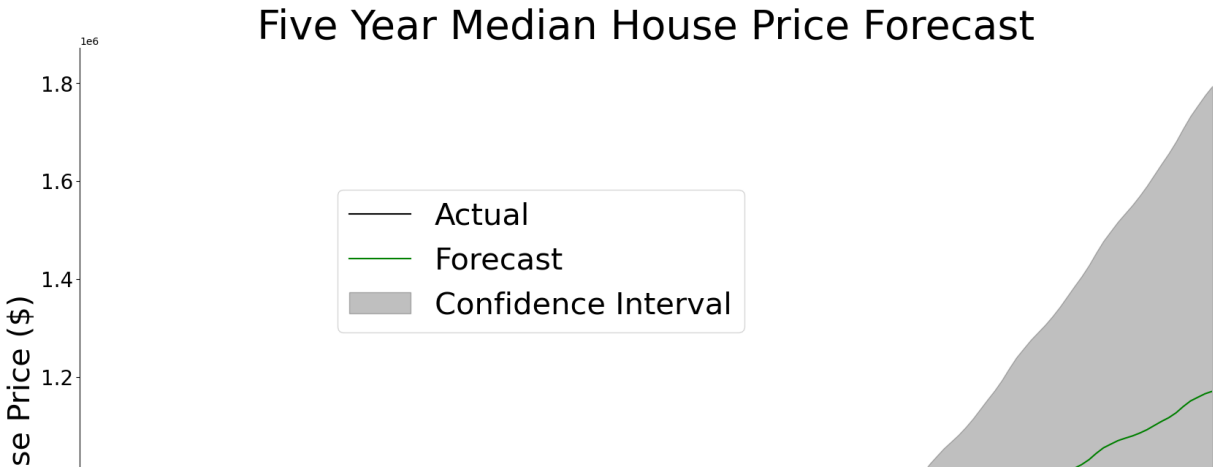

```
['94606']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8335      0.039     21.543      0.000       0.758      0.909
ma.L1          0.5934      0.048     12.337      0.000       0.499      0.688
ma.S.L12      -0.2920      0.037     -7.855      0.000      -0.365     -0.219
sigma2      3.391e+06   3.21e+05     10.559      0.000    2.76e+06   4.02e+06
==============================================================================
```


Standardized residual for "9"
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

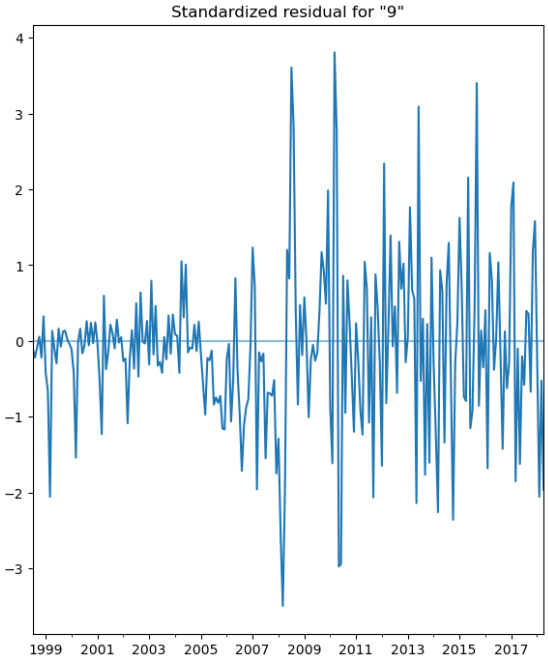

Histogram plus estimated density
KDE
N(0,1)
Hist
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

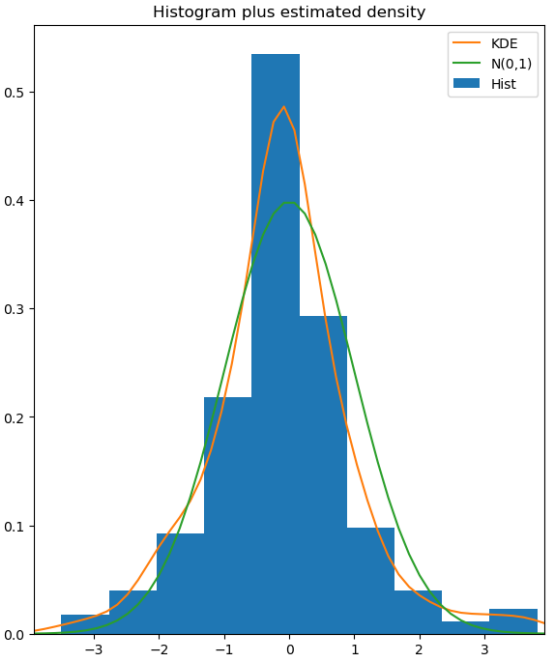

Normal Q-Q
Sample Quantiles
2
1
0
−1
−2
−3
−2
−1
0
1
2
Theoretical Quantiles

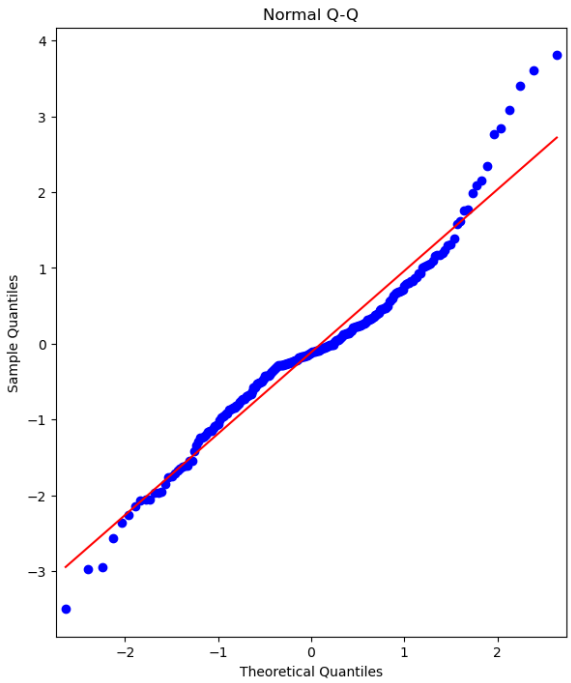

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

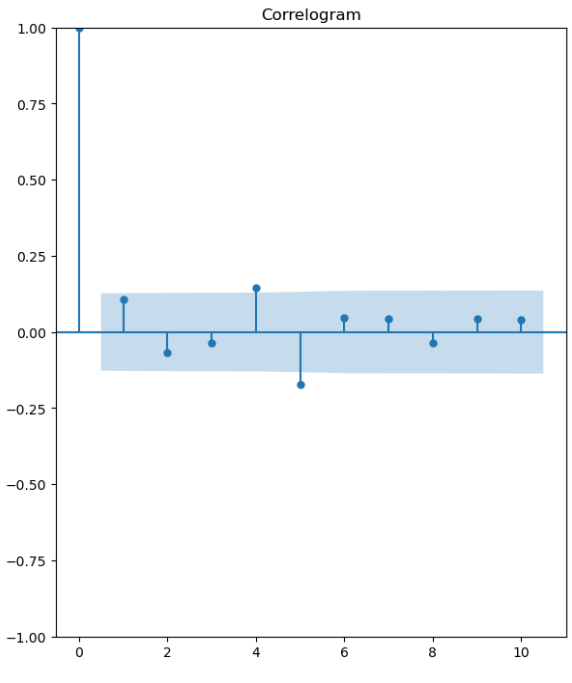


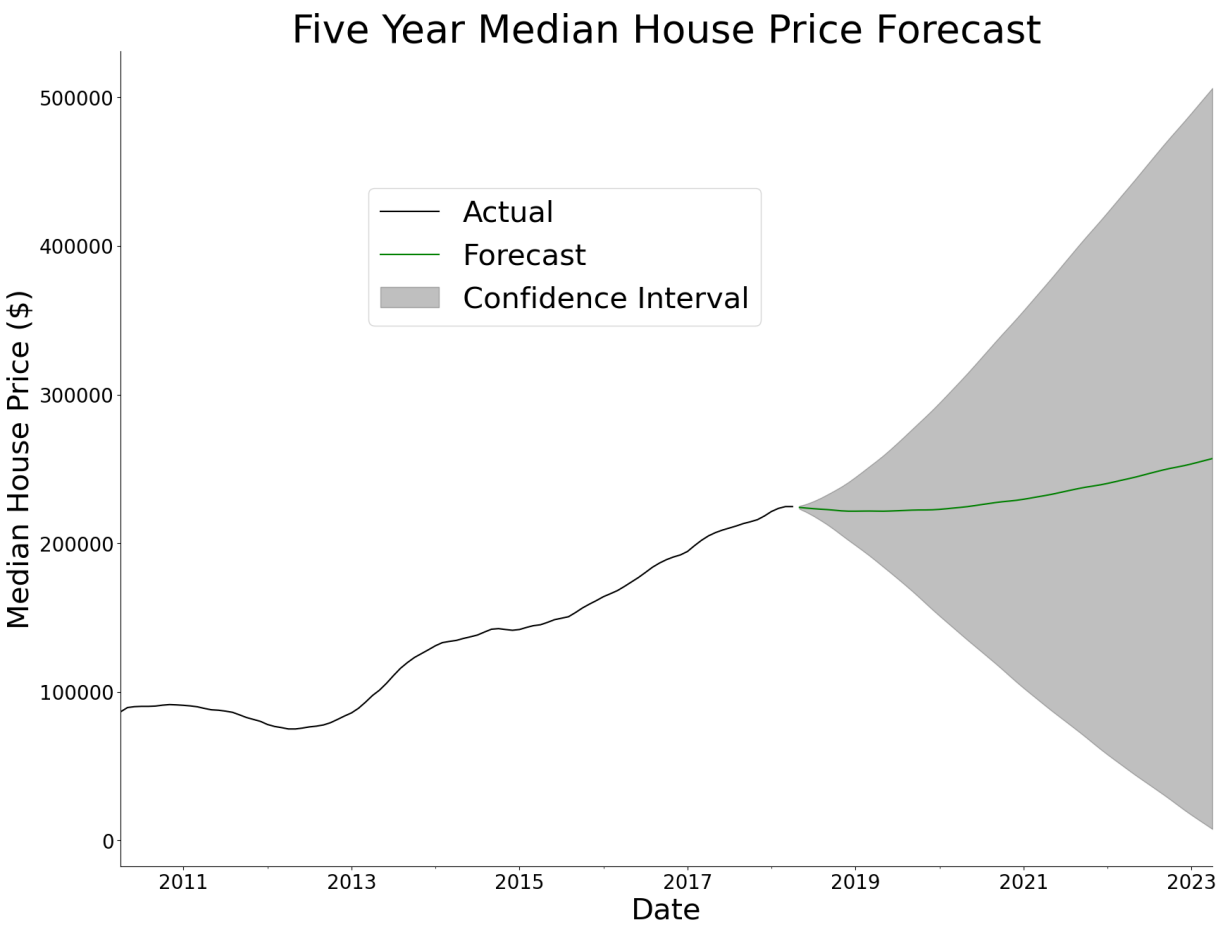

```
['95824']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9601      0.012     82.851      0.000       0.937
0.983
ma.L1          0.6136      0.038     15.984      0.000       0.538
0.689
ar.S.L12       0.1000      0.030      3.377      0.001       0.042
0.158
ma.S.L12      -0.9692      0.104     -9.328      0.000      -1.173
-0.766
sigma2      2.585e+05   3.27e+04      7.913      0.000    1.94e+05
3.23e+05
==============================================================================
```


Standardized residual for "9"
4
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017

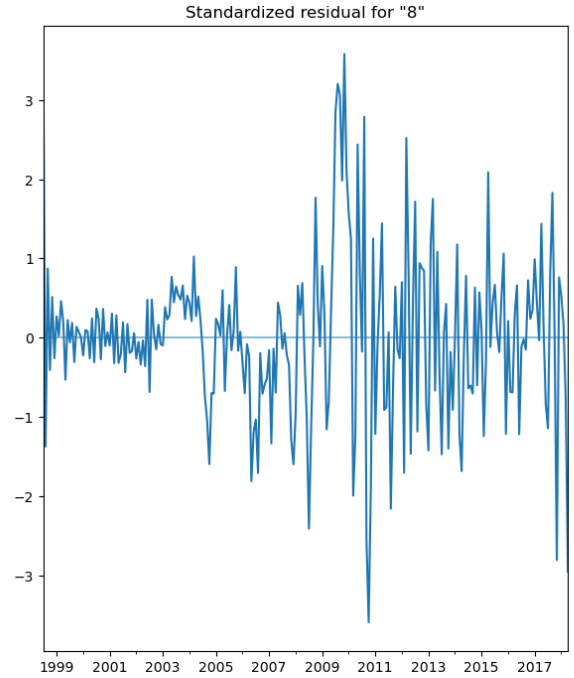

Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

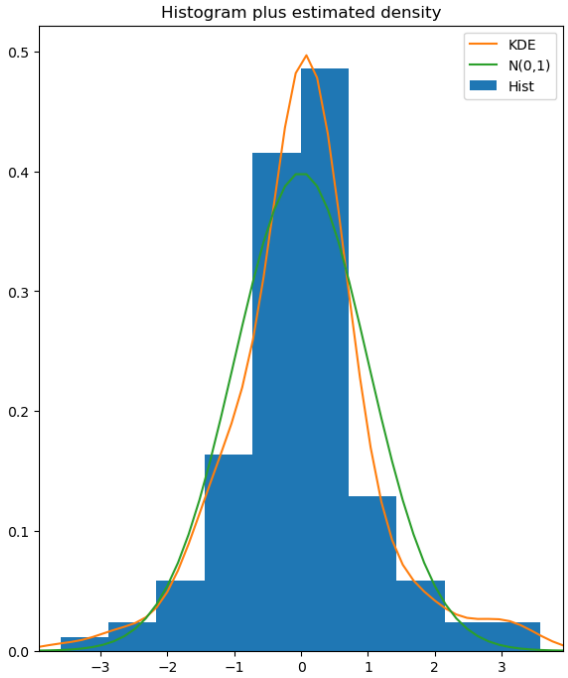

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−2
−1
0
1
2


Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

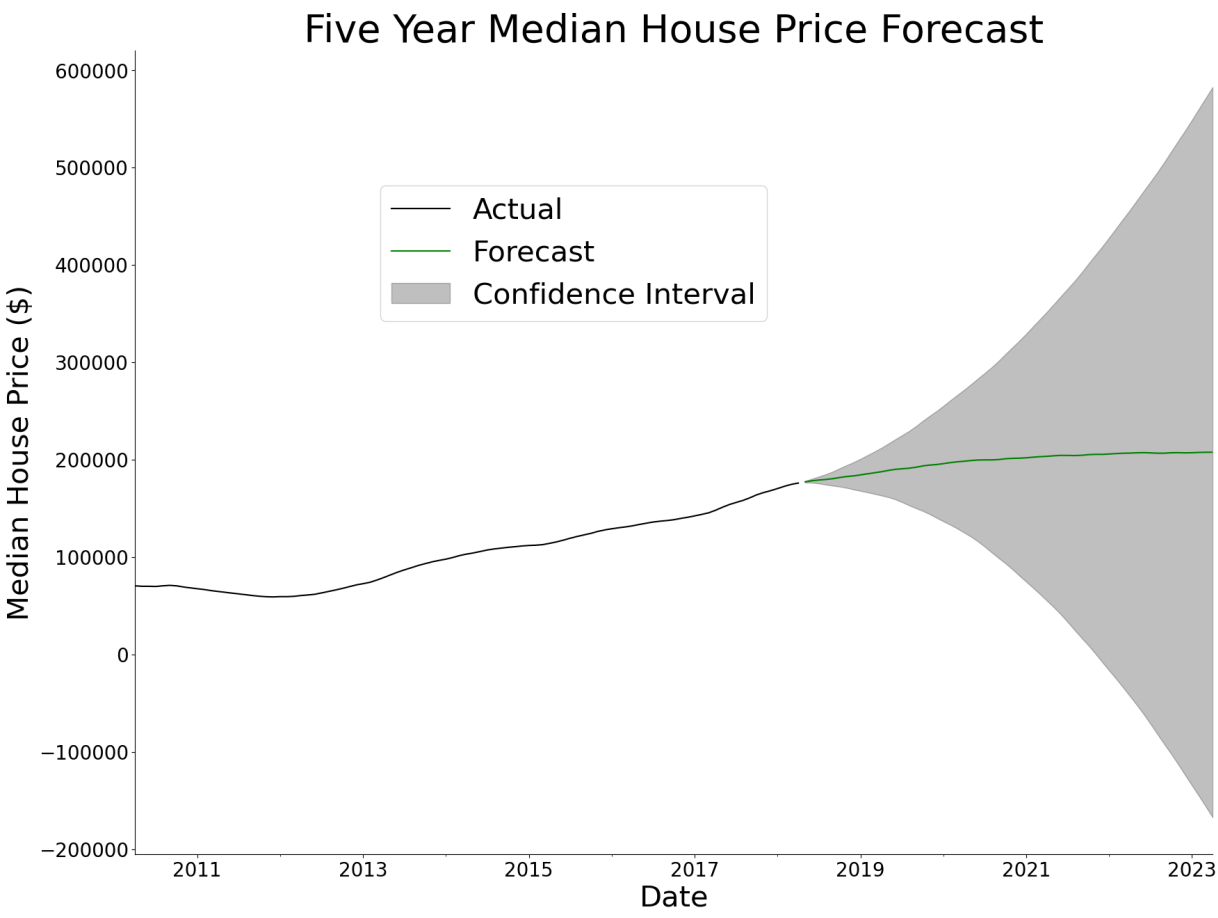

Five Year Median House Price Forecast
Median House Price ($)
500000
400000
300000
200000
100000
0
Actual
Forecast
Confidence Interval
2011
2013
2015
2017
2019
2021
2023
Date

```
['89115']
(1, 1, 1) (1, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9733      0.012     77.911      0.000       0.949
0.998
ma.L1          0.3587      0.019     19.009      0.000       0.322
0.396
ar.S.L12      -0.4685      0.074     -6.347      0.000      -0.613
-0.324
ma.S.L12      25.3835     42.137      0.602      0.547     -57.203
107.970
sigma2       264.1590    880.163      0.300      0.764   -1460.928
1989.246
==============================================================================
```

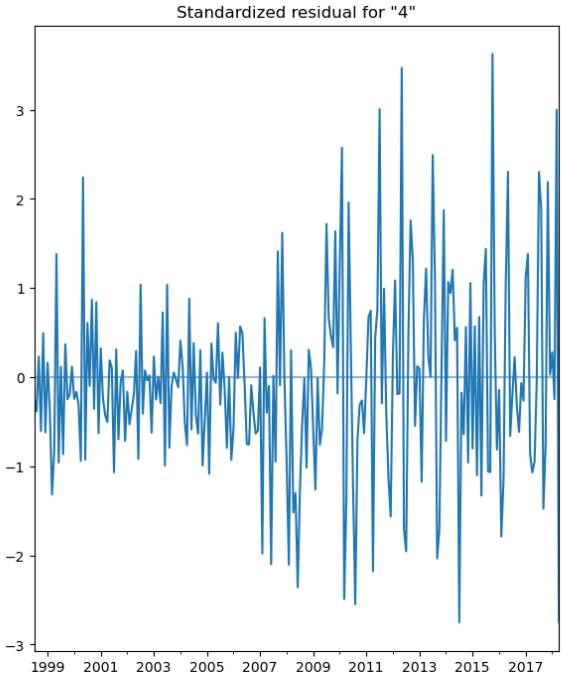


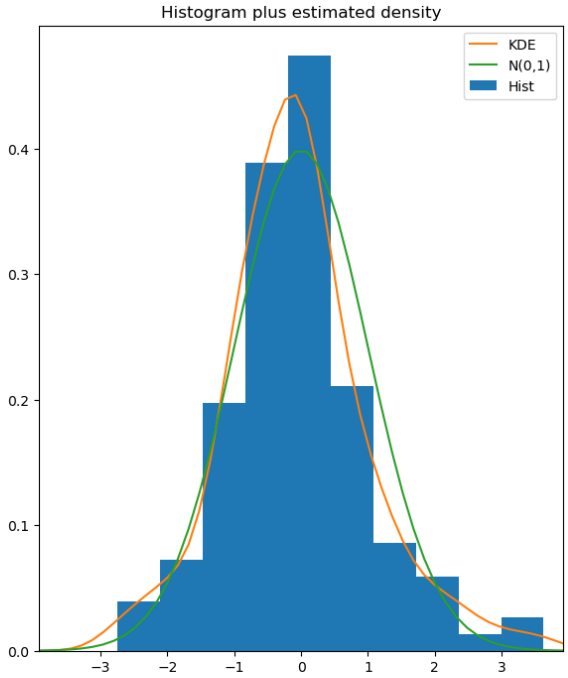


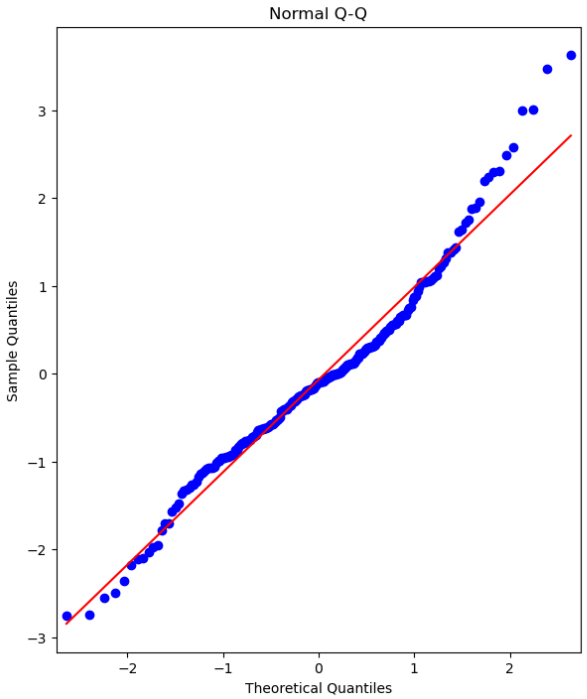
Normal Q-Q Correlogram

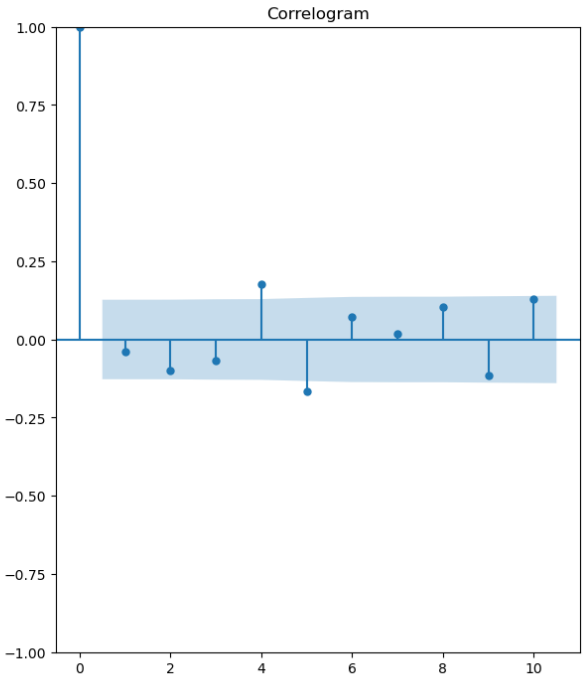


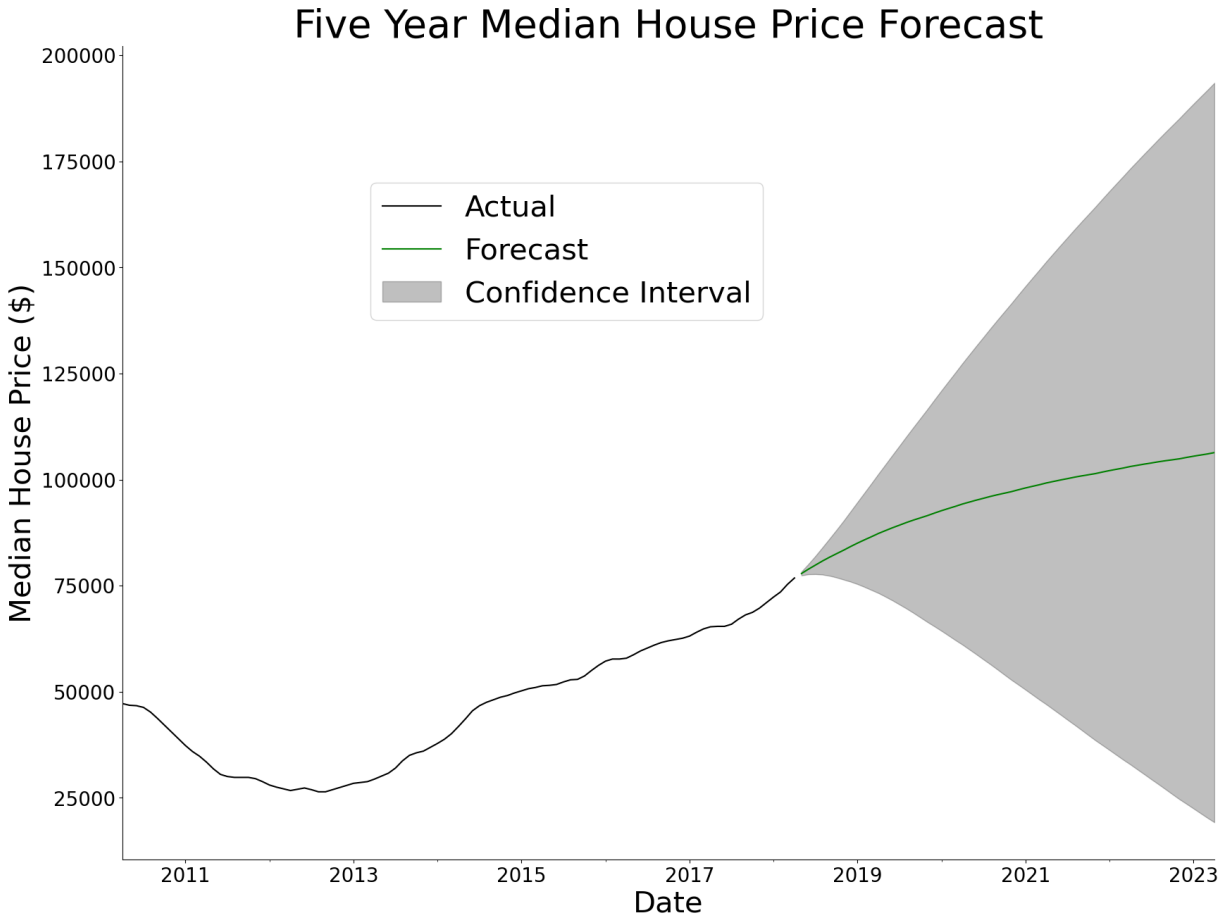
```
['48240']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
ar.L1          0.9519      0.020     47.106      0.000       0.912
0.991
ma.L1          0.5049      0.050     10.011      0.000       0.406
0.604
ma.S.L12      -0.9968      1.096     -0.910      0.363      -3.144
1.150
sigma2      5.313e+04   5.72e+04      0.929      0.353    -5.9e+04
1.65e+05
==============================================================================
```


Standardized residual for "4"
3
2
1
0
−1
−2
−3
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017


Histogram plus estimated density
KDE
N(0,1)
Hist
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3

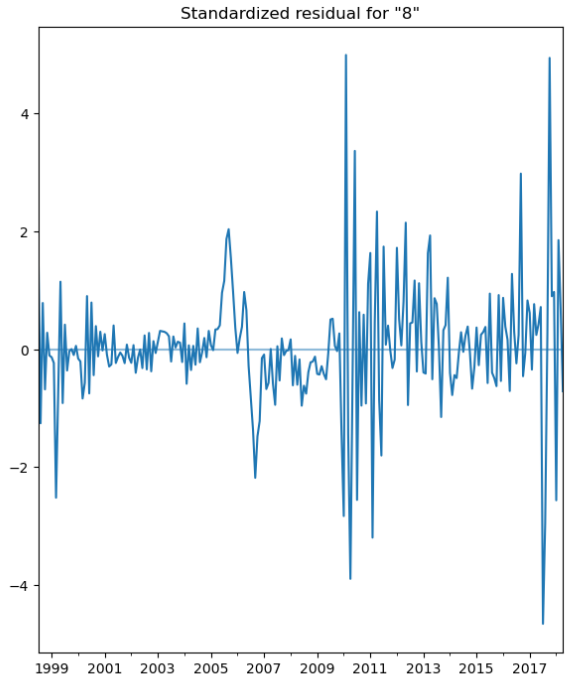

Normal Q-Q
Sample Quantiles
Theoretical Quantiles
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

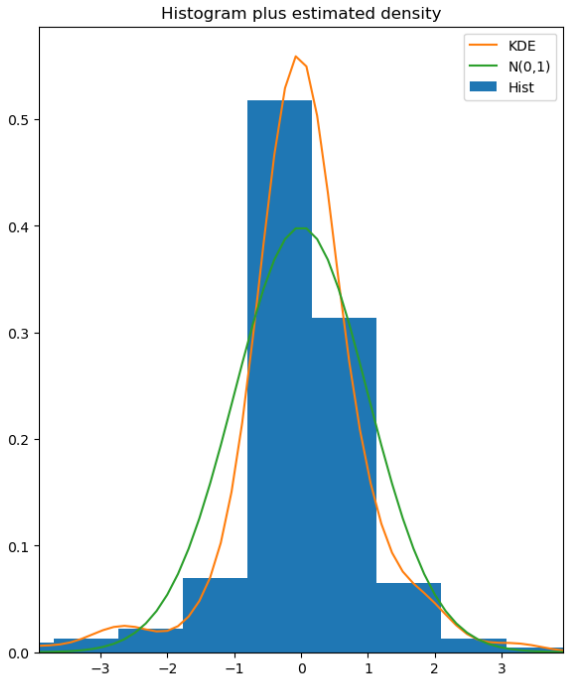

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

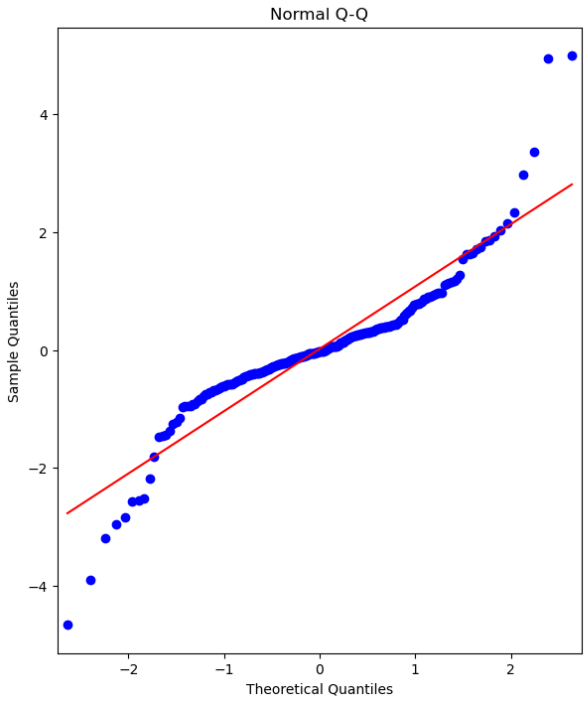


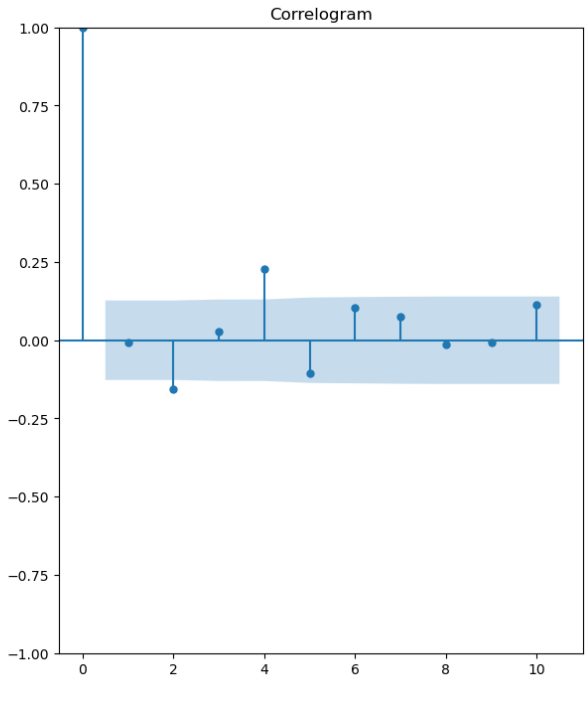

```
['83703']
(1, 1, 1) (0, 1, 1, 12)
==============================================================================
==========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
---------
ar.L1          0.7428      0.038     19.304      0.000       0.667
0.818
ma.L1          0.6268      0.033     18.860      0.000       0.562
0.692
ma.S.L12      -0.4391      0.032    -13.887      0.000      -0.501
-0.377
sigma2      5.895e+05   3.31e+04     17.792      0.000    5.25e+05
6.54e+05
```

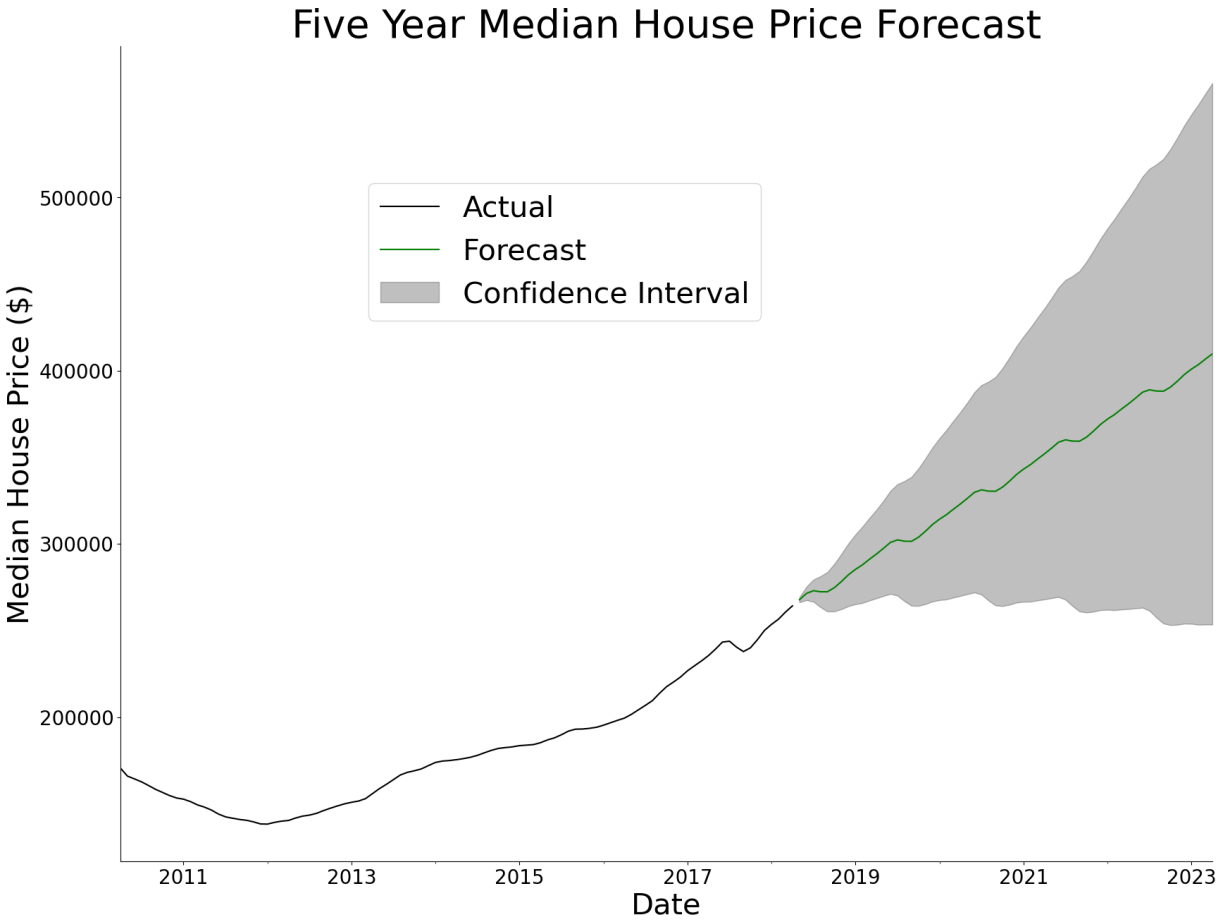
==========


Standardized residual for "8"
4
2
0
−2
−4
1999
2001
2003
2005
2007
2009
2011
2013
2015
2017


Histogram plus estimated density
KDE
N(0,1)
Hist
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3


Normal Q-Q
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−2
−1
0
1
2


Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10


Five Year Median House Price Forecast
Actual
Forecast
Confidence Interval
Median House Price ($)
500000
400000
300000
200000
2011
2013
2015
2017
2019
2021
2023
Date

In [75]:

```
#pred_list.append(pred['2023-04-01'])
pred_list_1
```

executed in 32ms, finished 14:02:41 2022-05-03

Out[75]:

```
[416016.39007328794,
 1171061.0561964822,
 257003.9336061536,
 207692.41758930407,
 106363.34240621062,
 409711.22703533754]
```

7.1 Calculate predicted rate of return after 5 years

In [76]:

```
# Create column for predicted price after 5 years
top_prospects['5_Year_Price'] = pred_list_1

# Calculate Percent Return from time of investment
top_prospects['Rate_Return'] = ((
    top_prospects['5_Year_Price']-
    top_prospects['2018-04'])/
    top_prospects['2018-04'])*100

# Bar Chart of Rate of Return by Zip Code
#plt.barh(top_prospects['Rate_Return'], align='center')

```

executed in 27ms, finished 14:02:43 2022-05-03

In [77]:

```
top_prospects
```

executed in 59ms, finished 14:02:44 2022-05-03

Out[77]:

	ZipCode	City	State	Metro	CountyName	SizeRank	1996-04	1996-05
2627	33460	Lake Worth	FL	Miami-Fort Lauderdale	Palm Beach	2628	59800.0	59900.0
1681	94606	Oakland	CA	San Francisco	Alameda	1682	120400.0	120300.0
4298	95824	Sacramento	CA	Sacramento	Sacramento	4299	73800.0	73400.0
815	89115	Las Vegas	NV	Las Vegas	Clark	816	92500.0	92500.0
5830	48240	Redford	MI	Detroit	Wayne	5831	67800.0	68200.0
5764	83703	Boise	ID	Boise City	Ada	5765	107500.0	107500.0

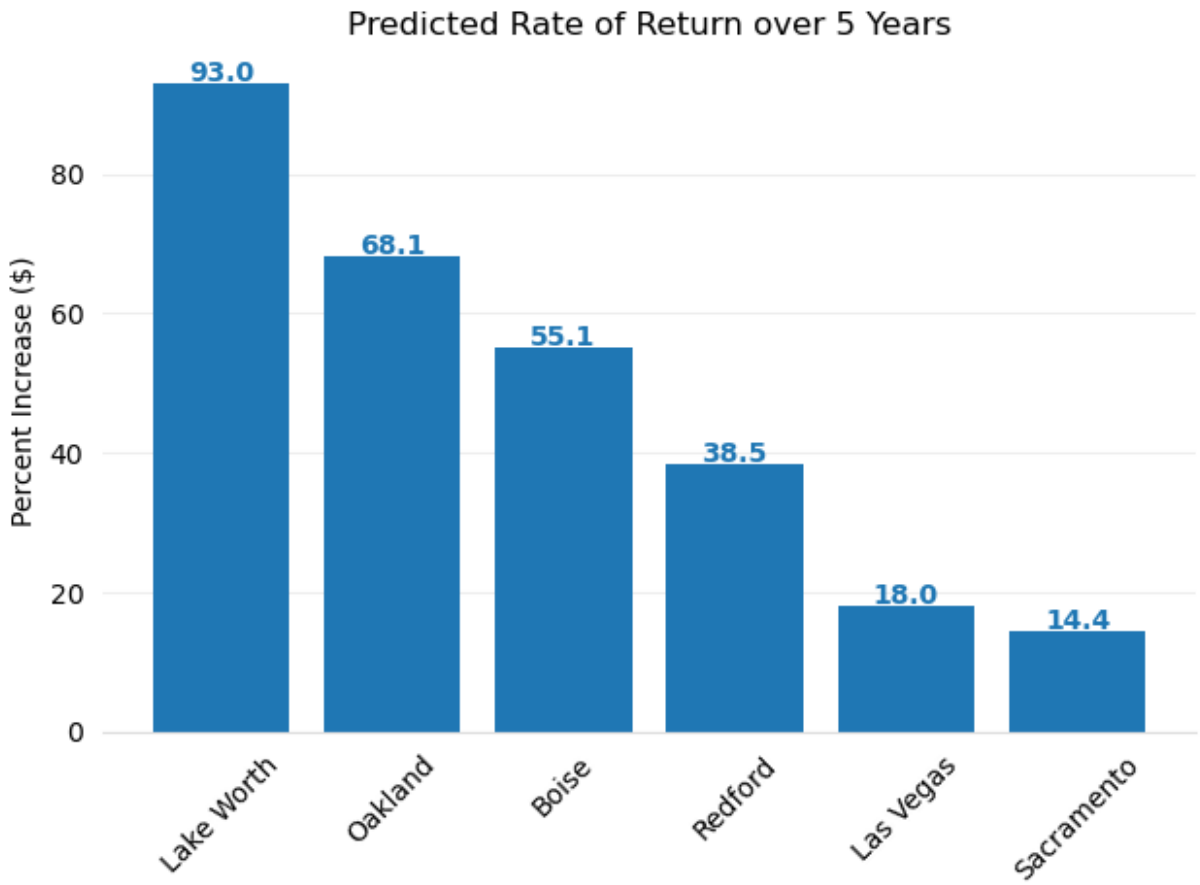
6 rows × 273 columns

7.2 Bar Chart: Rate of Return

In [79]:

```
# create df and sort by Rate_Return in Descending Order
rate_1 = top_prospects[['Rate_Return','City']]
#df.sort_values('importances',inplace=True)
rate_1.sort_values('Rate_Return',inplace=True,ascending=False)

fig, ax = plt.subplots()

# Save the chart so we can loop through the bars below.
bars = ax.bar(
    x=np.arange(rate_1['Rate_Return'].size),
    height=rate_1['Rate_Return'],
    tick_label=rate_1['City']
)

# Axis formatting.
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.spines['bottom'].set_color('#DDDDDD')
ax.tick_params(bottom=False, left=False)
ax.set_axisbelow(True)
ax.yaxis.grid(True, color='#EEEEEE')
ax.xaxis.grid(False)

# Grab the color of the bars so we can make the
# text the same color.
bar_color = bars[0].get_facecolor()

# Add text annotations to the top of the bars.
# Note, you'll have to adjust this slightly (the 0.3)
# with different data.
for bar in bars:
  ax.text(
      bar.get_x() + bar.get_width() / 2,
      bar.get_height() + 0.3,
      round(bar.get_height(), 1),
      horizontalalignment='center',
      color=bar_color,
      weight='bold'
  )

plt.xticks(rotation = 45)
plt.title('Predicted Rate of Return over 5 Years')
plt.ylabel('Percent Increase ($)')

fig.tight_layout()
```

executed in 330ms, finished 14:03:20 2022-05-03



8 Recommendations

8.1 Oakland, CA 94606

- Has a predicted rate of return of 68.1% over the next 5 years (assuming we are in April of 2018 when this dataset was current)
- Has the lowest margin of risk - with the lowest confidence interval value being substantially higher than the models for other zip codes.
- High level of model accuracy:
  - Predictions within 3% of median value on average.

8.2 Lake Worth, FL 33460

- Highest predicted return of 93.0%
- Much higher risk with wide range and large negative potential for the confidence interval price at 5 years
- High level of model accuracy:
  - Predictions within 1.3% of median value on average.

8.3 Boise, ID 83703

- Fairly high predicted return of 55.1%
- Low risk, with lowest confidence interval value being substantially higher than the models for other zip codes.
- Moderate level of model accuracy:
  - Predictions within 8% of median value on average.

9 Further Study

9.1 Adding to the model

- Need more recent data
- Add external influences, such as:
  - Average interest rates on mortgages
  - Overall employment, nationally or locally
  - Percent Increase or decrease in jobs in the county of that zip code

9.2 Ongoing testing

- Need monitoring and testing of price changes in real-time
- Weekly updates for price changes

9.3 Additional zip codes

- Most big cities, such as New York, Los Angeles, and Chicago aren't represented.
  - These big markets should be tracked alongside the top returning markets.

9.4 Top returning zip codes

- Find a variety of time periods to qualify the top returning zip codes for investment options.