FORECASTING REAL ESTATE PRICES USING TIME SERIES ANALYSIS

Final Project Phase 4 Group 2 Members

- Emmanuel Kiprotich
- Winnie Onduru
- Denis Sang
- Phelix Okumu
- Hellen Mwangi
- Ismail Ibrahim
- Sheila Machaha

Executive Summary

The project's goal was to provide actionable insights for U.S. property investments across different zip codes using data-driven methods. It utilized a comprehensive dataset from Zillow with 14,723 records spanning April 1996 to April 2018, showing property value trends from $118,299 to 288,040$. The dataset covered a wide value range, supporting trend analysis, regional benchmarking, and informed investment choices.

Data preparation involved meticulous checks and transformations using Python libraries like pandas, numpy, and pmdarima. It encompassed data type validation, filling missing values, and eliminating duplicates. Key metrics like Return on Investment (ROI) were engineered, and the dataset was optimized for analysis. Thorough exploratory data analysis (EDA), including univariate, bivariate, and multivariate analyses, was conducted. Data stationarity was evaluated via the Dickey-Fuller test, and non-stationary data were made stationary through differencing.

For modeling, the ARIMA model was chosen due to its ability to capture time-dependent patterns, handle nonlinearity and stationarity. The model underwent rigorous training and validation. Evaluation centered on Root Mean Square Error (RMSE) metrics, comparing forecasted and actual test data. Visualizations were created for intuitive comparison. RMSE values for each zip code quantified the model's predictive accuracy, aiding potential real estate investors. By combining meticulous data preparation, advanced modeling, and thorough evaluation, the project aimed to be a definitive guide for data-driven property investment

Step 1.Business Understanding

1.1 Background

In a constantly evolving real estate landscape, various elements, including economic shifts, population dynamics, market emotions, and regulatory shifts, converge to shape market trends. Acknowledging this complexity, the Kar-Dak Investment Group acknowledges the strategic advantage of harnessing data science methodologies to proactively uncover lucrative investment prospects.

1.2 Business Problem

Property investment offers multiple revenue streams including leasing income and asset growth, and provides benefits such as passive income and portfolio diversification, which makes it an attractive option for many investors. However, the critical issue at hand is identifying optimal locations for real estate investment. The primary objective of this project is to facilitate informed real estate investment decisions for the Kar-Dak Investment Group. By leveraging the extensive Zillow housing dataset, which spans from April 1996 to April 2018, the project aims to identify the top 5 most favorable zip codes for potential investment opportunities.

1.3 Project Question

This Project aims at answering the question:

1. What are the top 5 best zipcode for the Kar-Dak Investments to consider?

1.4 Objectives

1. To identify the top 5 zip codes with the highest ROI.
2. To develop time series models to forecast real estate prices for different zip codes over various time horizons.
3. To establish cities that are optimal for both short-term and long-term investment

Step 2.Data Understanding

In this section, we will do the following to get more insights about our dataset before proceeding to subsequent steps.

1. Import the Libraries
2. Load and Explore the Time Series Data
3. Inspect the Data Types
4. Inspect the column values

2.1 Summary Of The Dataset

The key columns in the dataset are as follows:

RegionID - This is unique Id for the Regions.

SizeRank - This is the ranking done based on the size of the Region.

RegionName - This field contains the zip code of the Region.

RegionType - Type of Region is Zip.

StateName - State.

City - This column provide the specific City Name of Housing Data.

Metro - This provide the name of the metro city around that Region.

County Name - This is the County Name for that Region.

Months Column - These Columns contains the prices of Region for every month.

By analyzing this dataset, we can gain a comprehensive understanding of the top five most promising real estate investment zip code areas.

The dataset covers a significant time period (April 1996 to April 2018), allowing for the exploration of long-term trends and capturing various market conditions. It provides a valuable resource for conducting time series analysis and developing predictive models to forecast

future prices

2.2 Importing Necessary Packages

In the cell below import various libraries.

In [1]:
```
#Importing the libraries to use
import pandas as pd
import numpy as np
import pmdarima as pm
import matplotlib.pyplot as plt
from statsmodels.tsa.holtwinters import ExponentialSmoothing
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
import folium
import numpy as np
from sklearn.metrics import mean_squared_error
import statsmodels.api as sm
from statsmodels.tsa.stattools import adfuller
import matplotlib.dates as mdates
import math
from statsmodels.tsa.arima_model import ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.graphics.tsaplots import plot_pacf
```

2.3 Loading and Exploring the Dataset

Below cell loads the csv file of zillow dataset, using a read_csv method, as a dataframe and saving it in a variable df, for purposes of subsequent work.

In [2]:
```
#Loading dataset into pandas DataFrame
df = pd.read_csv('zillow_data.csv')
```

The following code cell does a general view of the df, checking on top5 and bottom 5 of the df with a general indication of number of rows and columns at the bottom

In [3]:
```
# viewing df
df.head()
```

Out[3]:

	RegionID	RegionName	City	State	Metro	CountyName	SizeRank	1996-04	1996-0
0	84654	60657	Chicago	IL	Chicago	Cook	1	334200.0	335400.
1	90668	75070	McKinney	TX	Dallas-Fort Worth	Collin	2	235700.0	236900.
2	91982	77494	Katy	TX	Houston	Harris	3	210400.0	212200.
3	84616	60614	Chicago	IL	Chicago	Cook	4	498100.0	500900.
4	93144	79936	El Paso	TX	El Paso	El Paso	5	77300.0	77300.

5 rows × 272 columns

In [4]:
```
df.tail()
```

Out[4]:

	RegionID	RegionName	City	State	Metro	CountyName	SizeRank	1996-04
14718	58333	1338	Ashfield	MA	Greenfield Town	Franklin	14719	94600.0
14719	59107	3293	Woodstock	NH	Claremont	Grafton	14720	92700.0
14720	75672	40404	Berea	KY	Richmond	Madison	14721	57100.0
14721	93733	81225	Mount Crested Butte	CO	NaN	Gunnison	14722	191100.0
14722	95851	89155	Mesquite	NV	Las Vegas	Clark	14723	176400.0

5 rows × 272 columns

In [5]:
```
#Checking on the shape of the dataset
df.shape
```

Out[5]:
```
(14723, 272)
```

From above preview of df DataFrame, it is evident that the df has 14723 rows and 272 columns. The data is in a wide Format, evident by having columns 8 onwards being with dates. We will need to convert this to a long format where we will have time represented by a single column with another column representing value.

In [6]:
```
# Checking the info of the data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14723 entries, 0 to 14722
Columns: 272 entries, RegionID to 2018-04
dtypes: float64(219), int64(49), object(4)
memory usage: 30.6+ MB
```

In general, the dataset we are using has the below attributes:

The dataframe has a total of 14723 entries, indexed from 0 to 14722 and a total of 272 Columns. The dataset contains three main data types:

- 219 columns with the floating point numbers data type.
- 49 columns with the integer data type.
- 4 columns with the object data type. And it consumes approximately 30.6 megabytes of memory.

In [7]:
```
# checking descriptive statistics of the df dataset

df.describe()
```

Out[7]:

	RegionID	RegionName	SizeRank	1996-04	1996-05	1996-06	1
count	14723.000000	14723.000000	14723.000000	1.368400e+04	1.368400e+04	1.368400e+04	1
mean	81075.010052	48222.348706	7362.000000	1.182991e+05	1.184190e+05	1.185374e+05	1
std	31934.118525	29359.325439	4250.308342	8.600251e+04	8.615567e+04	8.630923e+04	8
min	58196.000000	1001.000000	1.000000	1.130000e+04	1.150000e+04	1.160000e+04	1
25%	67174.500000	22101.500000	3681.500000	6.880000e+04	6.890000e+04	6.910000e+04	6
50%	78007.000000	46106.000000	7362.000000	9.950000e+04	9.950000e+04	9.970000e+04	9
75%	90920.500000	75205.500000	11042.500000	1.432000e+05	1.433000e+05	1.432250e+05	1
max	753844.000000	99901.000000	14723.000000	3.676700e+06	3.704200e+06	3.729600e+06	3

8 rows × 268 columns

In [8]:
```
# Checking the null values in the dataset by percentage of total values
# Calculate the percentage of null values for each column and get the top 30 l


top_null_columns = (df.isnull().sum() / df.shape[0] * 100).sort_values(ascend

# Display the top 20 columns with the highest percentage of null values
print("Top 20 Columns with Highest Percentage of Null Values:")
for column, percentage in top_null_columns.items():
    print(f"{column}: {percentage:.2f}%")
```

```
Top 20 Columns with Highest Percentage of Null Values:
Metro: 7.08%
1997-04: 7.06%
1996-08: 7.06%
1997-06: 7.06%
1997-05: 7.06%
1997-03: 7.06%
1997-02: 7.06%
1997-01: 7.06%
1996-11: 7.06%
1996-10: 7.06%
1996-09: 7.06%
1996-12: 7.06%
1996-07: 7.06%
1996-05: 7.06%
1996-04: 7.06%
1996-06: 7.06%
1997-08: 7.05%
1997-12: 7.05%
1997-11: 7.05%
1997-10: 7.05%
```

- Based on the above analysis Metro column has the highest percentage of null values at 7.08% with the rest betweeen 7.06% and 7.05%

In [9]:
```
# Checking for null values in the first 7 columns
# Assuming df is your DataFrame

null_counts = df.iloc[:, :7].isnull().sum()

print("Null Value Counts in the First 7 Columns:")
print(null_counts)
```

```
Null Value Counts in the First 7 Columns:
RegionID         0
RegionName       0
City             0
State            0
Metro         1043
CountyName       0
SizeRank         0
dtype: int64
```

- It is noted that the Metro column has the highest number of null values with 1,043, which was previously captured to be 7.08%

In [10]:
```
#Checking on the duplicates
df.duplicated().sum()
```

Out[10]:
```
0
```

It is noted that the dataset have no duplicated datapoints

3 Reshape from Wide to Long Format

Within this section, a transformation is executed on the data, transitioning it from a wide format to a long format. Furthermore, supplementary columns are incorporated into the dataset. These newly added columns encompass two vital metrics: percentage Return on Investment (ROI) and ROI in terms of price

In [11]:
```
# Creating a Column of %ROI

df["%ROI"] = ((df["2018-04"] / df["2009-01"]) ** (1 / (2018-2009)) - 1) * 100
df["%ROI"]
```

Out[11]:
```
0        2.596098
1        5.287075
2        3.281773
3        2.296917
4       -0.009141
            ...
14718    1.039444
14719    0.921918
14720    2.416872
14721    0.329810
14722    2.923535
Name: %ROI, Length: 14723, dtype: float64
```

In [12]:
```
# Creating a Column of Actual ROI
df['ROIPrice'] = df["2018-04"] - df["2009-01"] - 1
df.ROIPrice
```

Out[12]:
```
0        212299.0
1        119399.0
2         83199.0
3        241599.0
4          -101.0
            ...
14718     18599.0
14719     17899.0
14720     25799.0
14721     19399.0
14722     81599.0
Name: ROIPrice, Length: 14723, dtype: float64
```

The newly created %ROI

In [13]:
```
# Convert Wide Format to Long format and checking on the first five rows
data = pd.melt(df,
               id_vars=['RegionID','RegionName', 'SizeRank', 'City', 'State',
               var_name='Date')

# rename RegionID to zipcode
data = data.rename(columns={'RegionName': 'Zipcode','value':'Price'})

#convert zipcode to categorical datatype
data['Zipcode'] = data['Zipcode'].astype('str')

# convert date to datetime
data['Date'] = pd.to_datetime(data['Date'],format='%Y-%m')

data.head()
```

Out[13]:

	RegionID	Zipcode	SizeRank	City	State	Metro	CountyName	%ROI	ROIPrice	
0	84654	60657	1	Chicago	IL	Chicago	Cook	2.596098	212299.0	1
1	90668	75070	2	McKinney	TX	Dallas-Fort Worth	Collin	5.287075	119399.0	1
2	91982	77494	3	Katy	TX	Houston	Harris	3.281773	83199.0	1
3	84616	60614	4	Chicago	IL	Chicago	Cook	2.296917	241599.0	1
4	93144	79936	5	El Paso	TX	El Paso	El Paso	-0.009141	-101.0	1

In [14]:
```
#Checking On the Shape of the Long Format
data.shape
```

Out[14]:
```
(3901595, 11)
```

The applied process led to a substantial transformation in the dataset's structure. The original row count of 14,723 experienced a remarkable expansion, reaching approximately 3.9 million rows.

Simultaneously, the dataset's width underwent a significant reduction, with the number of columns shrinking from 272 to just 11.

In [15]:
```
#Checking for Duplicates Within the Dataset
print(f'The number of duplicates within the dataset is : {data.duplicated().s
```

```
The number of duplicates within the dataset is : 0
```

The above cell shows there is No Duplicated Value in the Dataset.

In [16]:
```
# Checking For Missing Values in Terms of Percentage
data.isna().sum()/len(data)*100
```

Out[16]:
```
RegionID      0.000000
Zipcode       0.000000
SizeRank      0.000000
City          0.000000
State         0.000000
Metro         7.084154
CountyName    0.000000
%ROI          3.986959
ROIPrice      3.986959
Date          0.000000
Price         4.021202
dtype: float64
```

The analysis revealed that the "Metro" column stands out with the highest occurrence of null values, constituting 7% of the total dataset. In contrast, other columns exhibit null values of less than 4%, suggesting a relatively minor impact on our analytical process.

Nonetheless, our chosen approach involves remedying the null values in the "Metro" column by filling them with the string "missing." This strategy is elaborated further in the following cell.

In [17]:
```
# Filling Missing Values in the metro column with "missing"
data['Metro'].fillna('Missing',inplace = True)
```

In [18]:
```
# Recheck for Missing Values
data.isna().sum()/len(data)*100
```

Out[18]:
```
RegionID      0.000000
Zipcode       0.000000
SizeRank      0.000000
City          0.000000
State         0.000000
Metro         0.000000
CountyName    0.000000
%ROI          3.986959
ROIPrice      3.986959
Date          0.000000
Price         4.021202
dtype: float64
```

In [19]:
```
# Checking on Descriptive Statistics of the Dataset
data.describe()
```

Out[19]:

	RegionID	SizeRank	%ROI	ROIPrice	Price
count	3.901595e+06	3.901595e+06	3.746040e+06	3.746040e+06	3.744704e+06
mean	8.107501e+04	7.362000e+03	2.078260e+00	6.260119e+04	2.076064e+05
std	3.193304e+04	4.250165e+03	2.060273e+00	1.624183e+05	2.400207e+05
min	5.819600e+04	1.000000e+00	-6.769992e+00	-5.328010e+05	1.130000e+04
25%	6.717400e+04	3.681000e+03	6.961308e-01	9.199000e+03	9.790000e+04
50%	7.800700e+04	7.362000e+03	1.989658e+00	2.829900e+04	1.476000e+05
75%	9.092100e+04	1.104300e+04	3.306184e+00	6.579900e+04	2.372000e+05
max	7.538440e+05	1.472300e+04	1.207663e+01	6.938599e+06	1.931490e+07

In [20]:
```
#statistical description of categorical variables
data.describe(include=['object'])
```

Out[20]:

	Zipcode	City	State	Metro	CountyName
count	3901595	3901595	3901595	3901595	3901595
unique	14723	7554	51	702	1212
top	60657	New York	CA	Missing	Los Angeles
freq	265	30210	324360	276395	69960

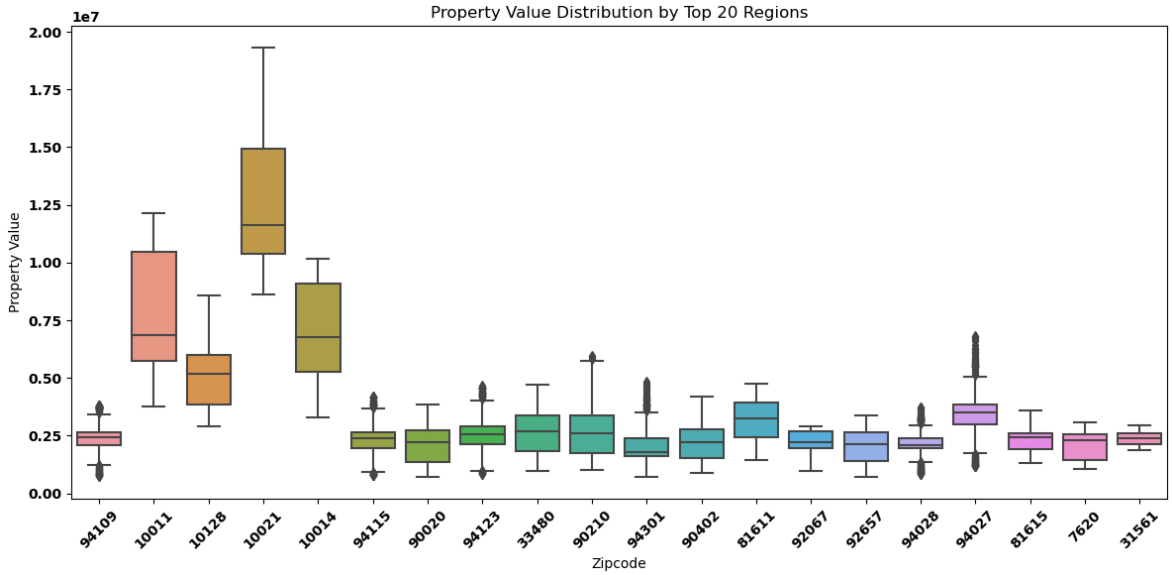
In [21]:
```
#Checking on the state column

data["State"].unique()
```

Out[21]:
```
array(['IL', 'TX', 'NY', 'CA', 'FL', 'TN', 'NC', 'GA', 'DC', 'MO', 'OK',
       'AZ', 'NJ', 'MD', 'VA', 'WA', 'OH', 'MI', 'MA', 'KS', 'NM', 'CT',
       'NV', 'PA', 'CO', 'OR', 'IN', 'SC', 'KY', 'AR', 'ND', 'MN', 'AL',
       'DE', 'LA', 'MS', 'ID', 'MT', 'HI', 'WI', 'UT', 'ME', 'SD', 'WV',
       'IA', 'RI', 'NE', 'WY', 'AK', 'NH', 'VT'], dtype=object)
```

In [22]:
```
font = {'family' : 'normal',
        'weight' : 'bold',
        'size'   : 22}
plt.rc('font', **font)
# NOTE: if you visualizations are too cluttered to read, try calling 'plt.gcf
```

In [23]:
```
import matplotlib.pyplot as plt

# Define a font configuration
font_config = {
    'family': 'sans-serif',  # You can try other font families as well
    'size': 10
}

# Update the font configuration for Matplotlib
plt.rc('font', **font_config)

# Your plotting code here
# ...

# Show the plot
plt.show()
```

In [25]:
```
# Checking For The Outlies

# Box plot
plt.figure(figsize=(12, 6))
top_regions = data.groupby('Zipcode')['Price'].mean().nlargest(20).index
top_data = data[data['Zipcode'].isin(top_regions)]
sns.boxplot(x='Zipcode', y='Price', data=top_data)
plt.title("Property Value Distribution by Top 20 Regions")
plt.xlabel("Zipcode")
plt.ylabel("Property Value")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



The dataset potrays presence of outliers in some zipcodes however this may be a true representation of the prices in the affected locality

Step 4.Explotary Data Analysis and Visualization

This section entails examining the relationship between variables using;

- Univariate analysis
- Bivariate analysis
- Multivariate analysis.

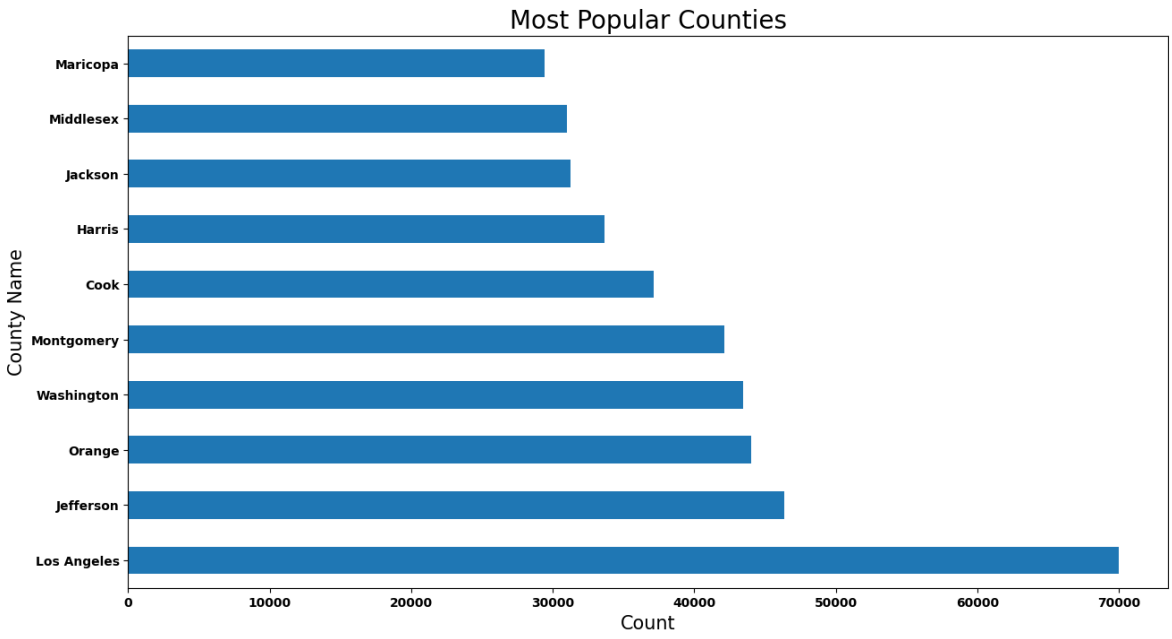
4.1 Univariate Analysis

Univariate Analysis involves the study of individual variables in isolation, focusing on their distribution and properties without considering the influence of other variables. This analytical approach provides a foundational understanding of the characteristics and behavior of a single variable.

4.1.1 Top 10 Most Popular Counties

The code below shows the top 10 most popular counties in the dataset

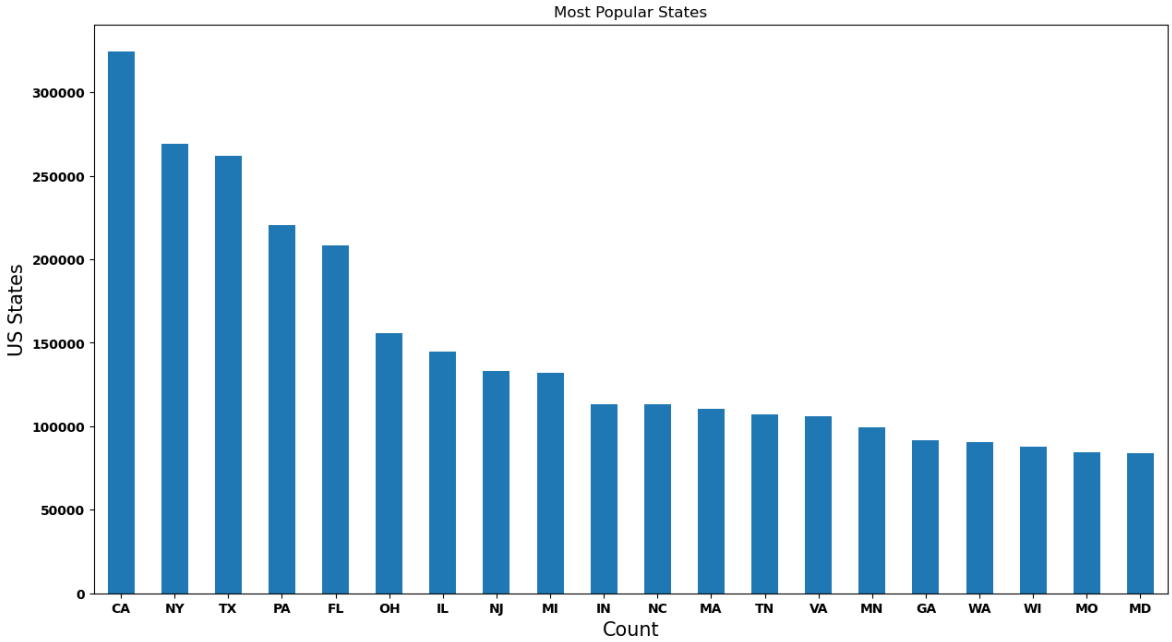
In [26]:
```
# # plotting the most popular counties in the dataset
plt.figure(figsize=(15,8))
data.CountyName.value_counts()[:10].plot(kind="barh")
plt.xlabel("Count",fontsize=15)
plt.ylabel("County Name",fontsize=15)
plt.title("Most Popular Counties",fontsize=20);
```



Based on the bar graph above, this shows that Los Angeles is the most popular county with around 70000 zipcodes.Most of the other counties range between 30000 and 50000 zipcodes.

4.1.2 Top 10 Most Popular States

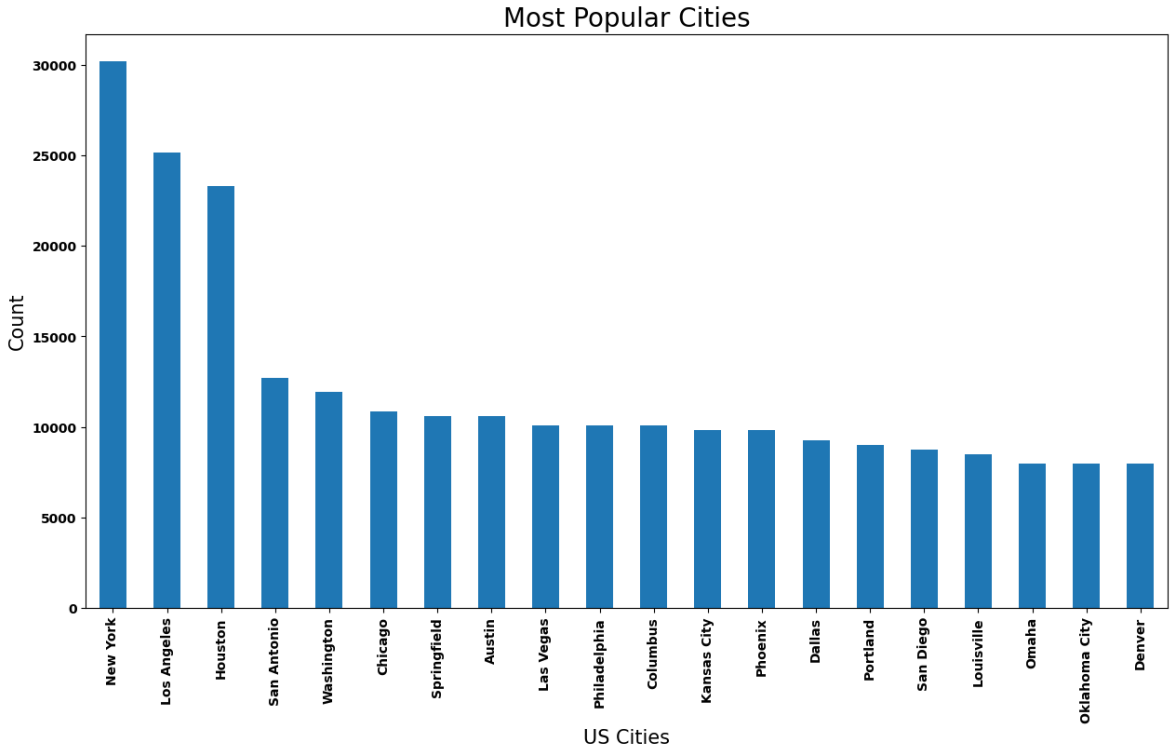
In [27]:
```
# plotting the most popular states in the dataset
plt.figure(figsize=(15,8))
data.State.value_counts()[:20].plot(kind="bar")
plt.xlabel("Count",fontsize=15)
plt.ylabel("US States",fontsize=15)
plt.xticks(rotation=0)
plt.title("Most Popular States",fontsize=12);
```



- The above graph shows that CA is the most popular with above 300000 zipcodes followed NY with aound 270000, TA with aound 260000 and others ranges below 250000 zipcodes.

4.1.3 The Most Popular Cities

In [28]:
```
# plotting the most popular cities in the dataset
plt.figure(figsize=(15,8))
data.City.value_counts()[:20].plot(kind="bar")
plt.xlabel("US Cities",fontsize=15)
plt.ylabel("Count",fontsize=15)
plt.title("Most Popular Cities",fontsize=20);
```



- From the dataset we can deduce that New York City, Los Angeles, and Houston are the most popular cities with above 20000 zipcodes. Other cities lie below 15000 zipcodes

4.1.4 The Distribution of Price

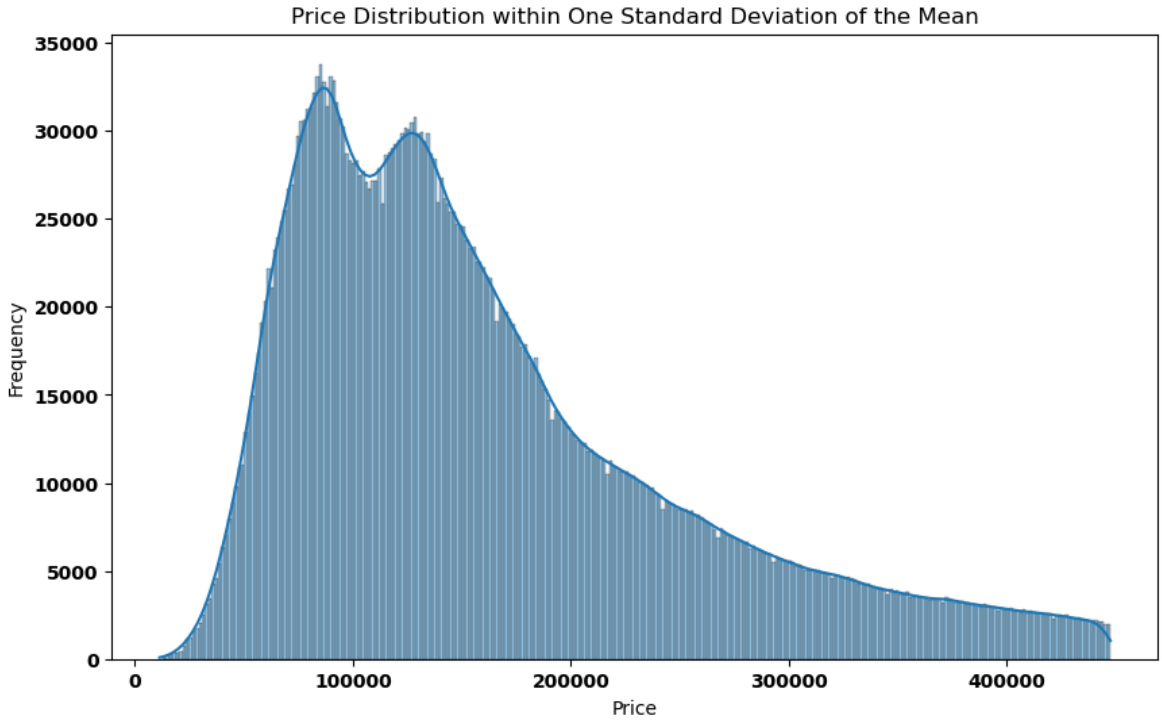
The cell below generates an histogram to check on the distribution of the price dataset

In [29]:
```
# calculate the average profit margin for each city
df_avg_value = data.groupby('State').mean().reset_index()

# sort the cities by average profit margin in descending order
df_sorted_states = df_avg_value.sort_values('Price', ascending=False)

# select the top 20 cities by average profit margin
top_states = df_sorted_states['State'].head(20)

# filter the DataFrame to include only the top 20 cities
df_top_states = data[data['State'].isin(top_states)].groupby("State").mean().
```

In [30]:
```
#Checking on price descriptive statistics
data["Price"].describe()
```

Out[30]:
```
count    3.744704e+06
mean     2.076064e+05
std      2.400207e+05
min      1.130000e+04
25%      9.790000e+04
50%      1.476000e+05
75%      2.372000e+05
max      1.931490e+07
Name: Price, dtype: float64
```

In [31]:
```
# Calculate mean and standard deviation
mean_price = data['Price'].mean()
std_price = data['Price'].std()

# Filter data within one standard deviation of the mean
filtered_data = data[(data['Price'] >= mean_price - std_price) & (data['Price

# Distribution plot
plt.figure(figsize=(10, 6))
sns.histplot(filtered_data['Price'], kde=True)
plt.title('Price Distribution within One Standard Deviation of the Mean')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()
```



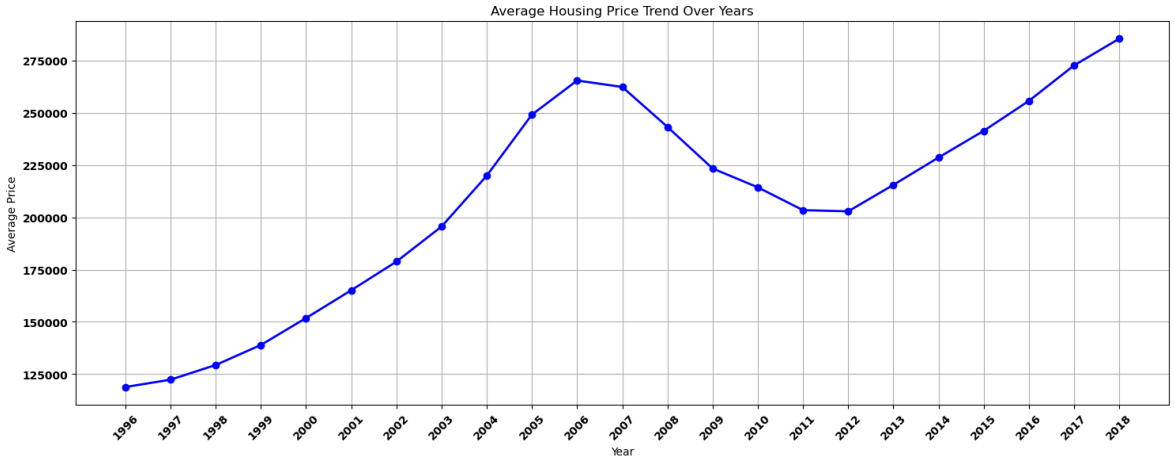
- Based on the above histogram the price dataset is right skewed and most of the prices are below mean_price.

4.2 Bivariate Analysis

- Under Bivariate analysis, we are checking the relationship between two variables within the dataset. Aiming to understand how changes in one variable might affect changes in another.This will also enable identification of patterns, correlation trends, or dependencies between the variables. Which will provide valuable insights for making decisions.

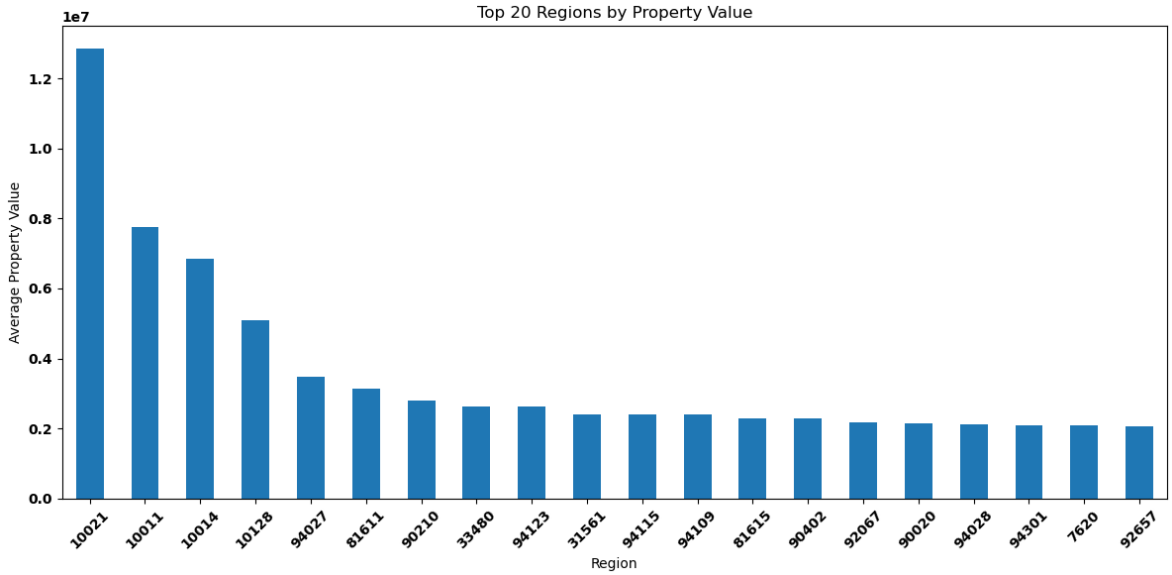
- Below cell visualize the average housing price trend from year 1996 to 2018

In [32]:
```
#Average price over the years

# Calculating average price per year

data['Year'] = data['Date'].dt.year
average_prices = data.groupby('Year')['Price'].mean()
# Creating a figure and axis
fig, ax = plt.subplots(figsize=(15, 6))
# Ploting the data
ax.plot(average_prices.index, average_prices.values, label='Average Price', c
# Adding labels and title
ax.set_xlabel('Year')
ax.set_ylabel('Average Price')
ax.set_title('Average Housing Price Trend Over Years')
# Adding grid lines
ax.grid(True)
# Customizing x-axis ticks for each year
plt.xticks(average_prices.index, rotation=45)
# Tight layout
plt.tight_layout()
# Show the plot
plt.show()
```



- The graph shows that the average housing price has been increasing steadily over the years, with a deep between 2007 and 2012. Since then, the average housing price has rebounded and continued to rise

In [33]:
```
# Checking on top 20 regions by property prices

plt.figure(figsize=(12, 6))
top_regions = data.groupby('Zipcode')['Price'].mean().nlargest(20)
top_regions.plot(kind='bar')
plt.title("Top 20 Regions by Property Value")
plt.xlabel("Region")
plt.ylabel("Average Property Value")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

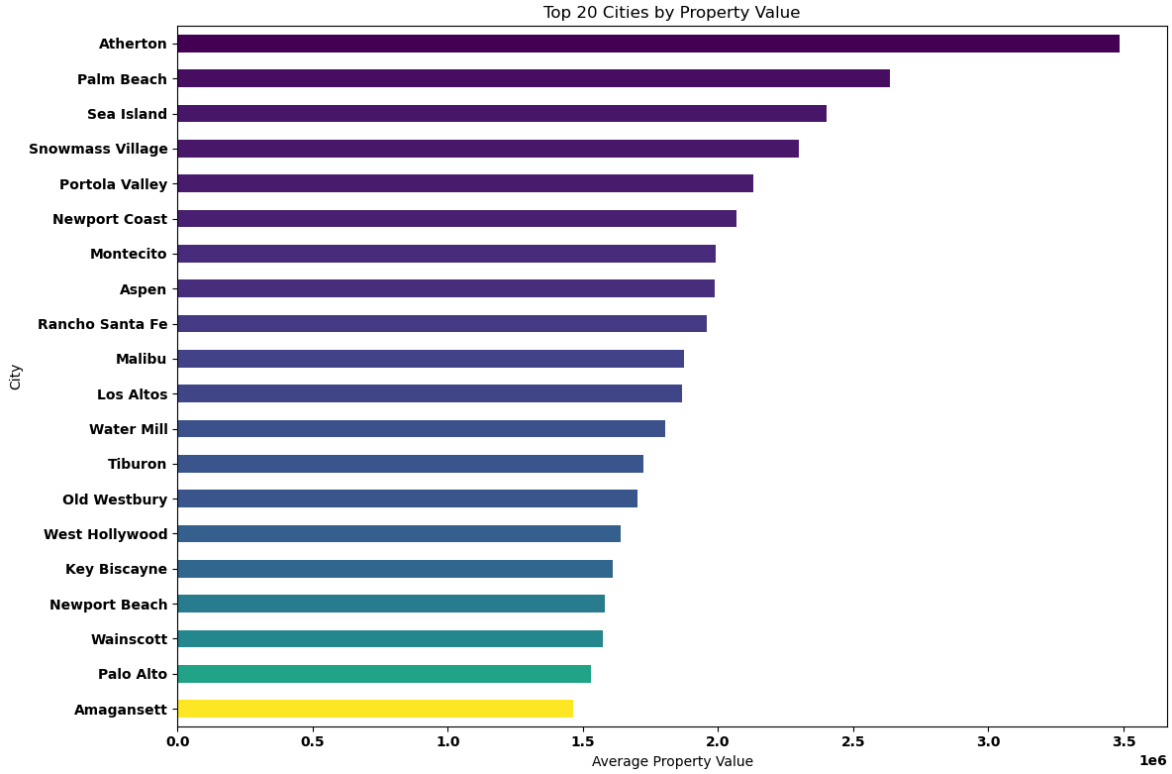


- Zipcode 10021,10011, 10014, 10128 in New York has properties with the highest value, followed by 94027 in California and 81611 in Colorado, and the other remaining Zipcodes range between 2 to 3 Million.

4.2.1 Top 20 cities by by property value

The cell below compute the mean of price and visualizes top 20 cities with highest mean value.

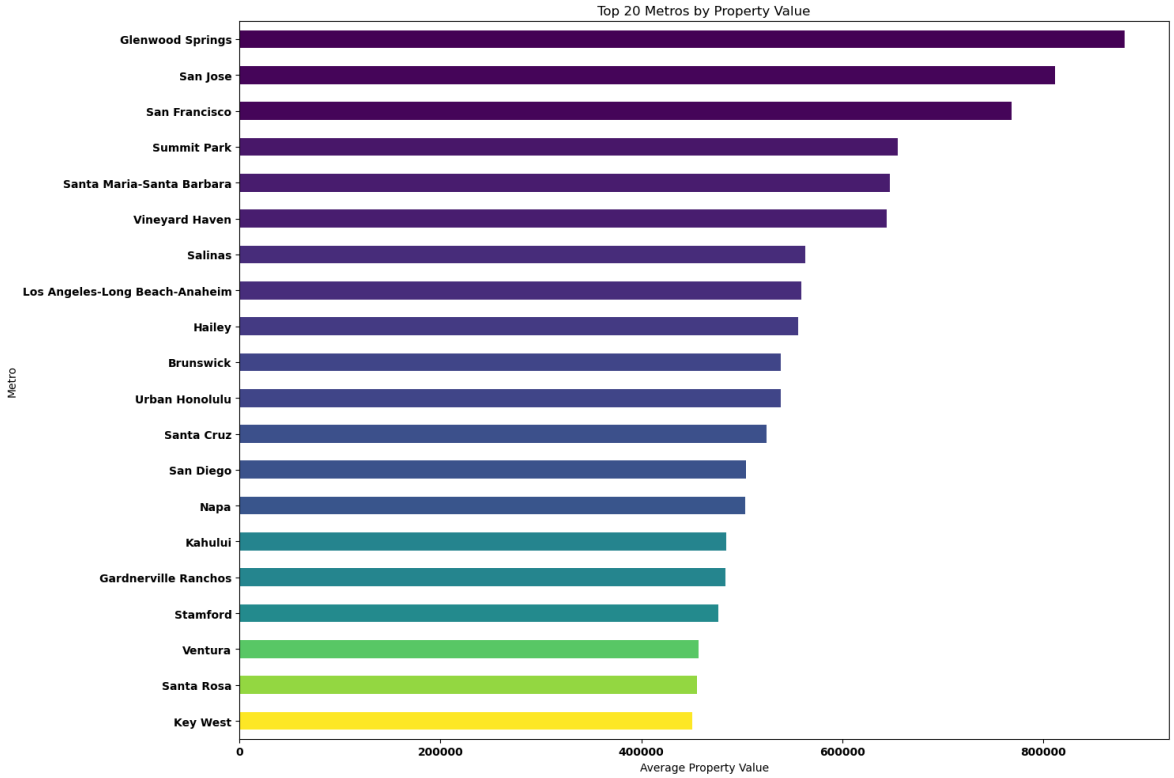
In [34]:
```
# #Top 10 Cities by property value
# Calculate the top 20 cities by average property value
top_regions = data.groupby('City')['Price'].mean().nlargest(20)

# Normalize values for coloring
normalized_values = (top_regions - top_regions.min()) / (top_regions.max() -

# Define a colormap
colormap = plt.cm.get_cmap('viridis')  # You can choose any colormap you pref

# Generate colors based on normalized values
colors = colormap(normalized_values)

# Create a bar plot
plt.figure(figsize=(12, 8))
top_regions.sort_values().plot(kind='barh', color=colors)
plt.title("Top 20 Cities by Property Value")
plt.xlabel("Average Property Value")
plt.ylabel("City")
plt.tight_layout()
plt.show()
```



- From the horizontal graph, Palm Beach, Malibu and Los Altos are the properties with the highest property value. Palm Beach with an average price of above 2.5 million. Other cities operates below price of 2 million

4.2.2 Top 20 Metropolitant cities

In [35]:
```
#Top 10 Metropolitant by price

# Flipped Axes Bar Plot: Property Value vs. Region
plt.figure(figsize=(15, 10))
top_regions = data.groupby('Metro')['Price'].mean().nlargest(20)
normalized_values = (top_regions - top_regions.min()) / (top_regions.max() -
colors = colormap(normalized_values)

top_regions.sort_values().plot(kind='barh', color=colors)
plt.title("Top 20 Metros by Property Value")
plt.xlabel("Average Property Value")
plt.ylabel("Metro")
plt.tight_layout()
plt.show()
```

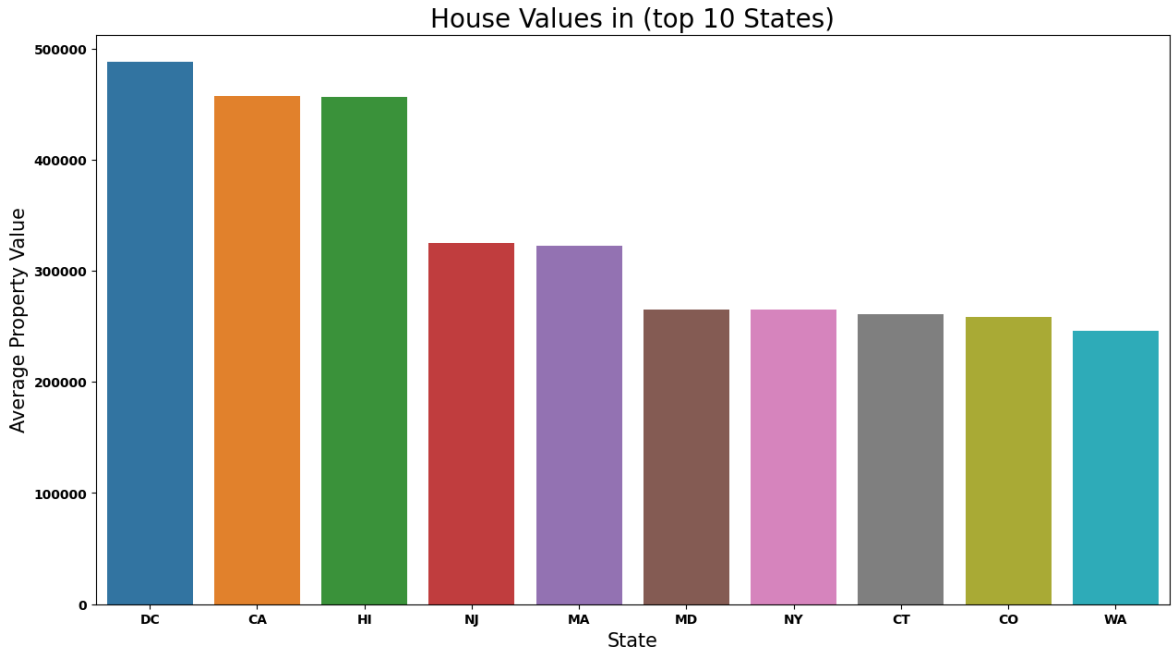


- from the bar graph San Jose and San Francisco are the most metropolitant counties with the highest value of the property with an average price above 600000.

4.2.3 Houses values in top 10 states

In [36]:
```
# calculate the average profit margin for each city
df_avg_value = data.groupby('State').mean().reset_index()

# sort the cities by average profit margin in descending order
df_sorted_states = df_avg_value.sort_values('Price', ascending=False)

# select the top 20 cities by average profit margin
top_states = df_sorted_states['State'].head(10)

# filter the DataFrame to include only the top 20 cities
df_top_states = data[data['State'].isin(top_states)].groupby("State").mean().
```

In [37]:
```
# create a bar plot of city vs. profit margin
fig,ax = plt.subplots(figsize=(15,8))
sns_plot = sns.barplot(x=df_top_states.index, y=df_top_states.Price,ax=ax)

# set the title and axis labels
ax.set_title('House Values in (top 10 States)',fontsize=20)
ax.set_xlabel('State',fontsize=15)
ax.set_ylabel('Average Property Value',fontsize=15)

# display the plot
plt.show();
```



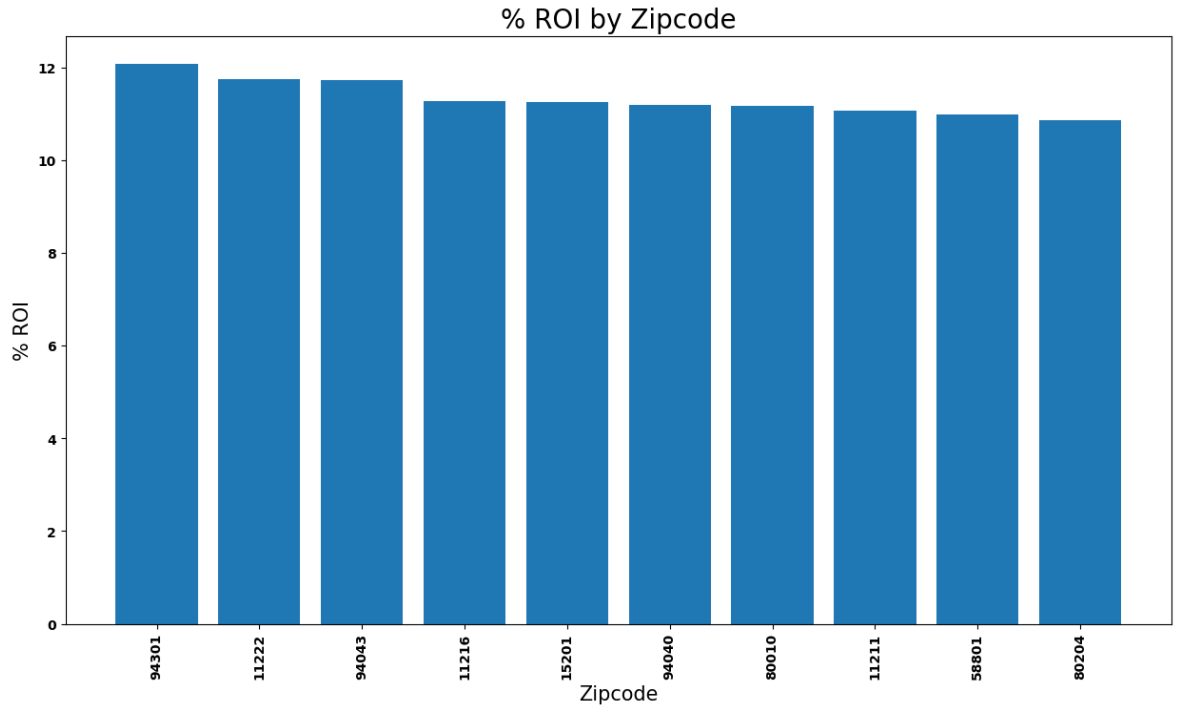
- from the bar graph, its noted that HI, DC and CA arethe top 3 states with the house property value above 4000000. Most of the others states operates with house value of below 4000000

4.2.4 Return On Investement By Zipcodes

The code belowgroup the zipcodes by Return On Investment percentage, picks the zipcodes with the highest property value and visualize.

In [38]:
```
# grouping data by mean %ROI and selecting top 30 zipcodes
grouped1 = data.groupby('Zipcode')
state_values1 = grouped1['%ROI'].mean()
state_values_df1 = state_values1.reset_index(name='% ROI')
state_values_df1 = state_values_df1.sort_values(by='% ROI', ascending=False)
top_thirty_zipcodes_df_roi = state_values_df1.head(10)
```

In [39]:
```
# plotting the %ROI by zipcode
plt.figure(figsize=(15,8))
plt.bar(top_thirty_zipcodes_df_roi['Zipcode'], top_thirty_zipcodes_df_roi['% 
plt.xlabel('Zipcode',fontsize=15)
plt.ylabel('% ROI',fontsize=15)
plt.title('% ROI by Zipcode',fontsize=20)
plt.xticks(rotation=90)
plt.show()
```



- From this graph, zipcode 94301 seems to be the most profitable zipcode at 12% ROI from 2009 to 2018.

4.3 Multivariate Analysis

This section seeks to understand relationships among three or more variables within a dataset.It also aims to uncover patterns, trends, and underlying structures that might not be evident when considering variables in isolation. By considering multiple variables together,we gain a more comprehensive understanding of complex systems and can make more nuanced predictions and informed decisions.

In [40]:
```
data.head()
```

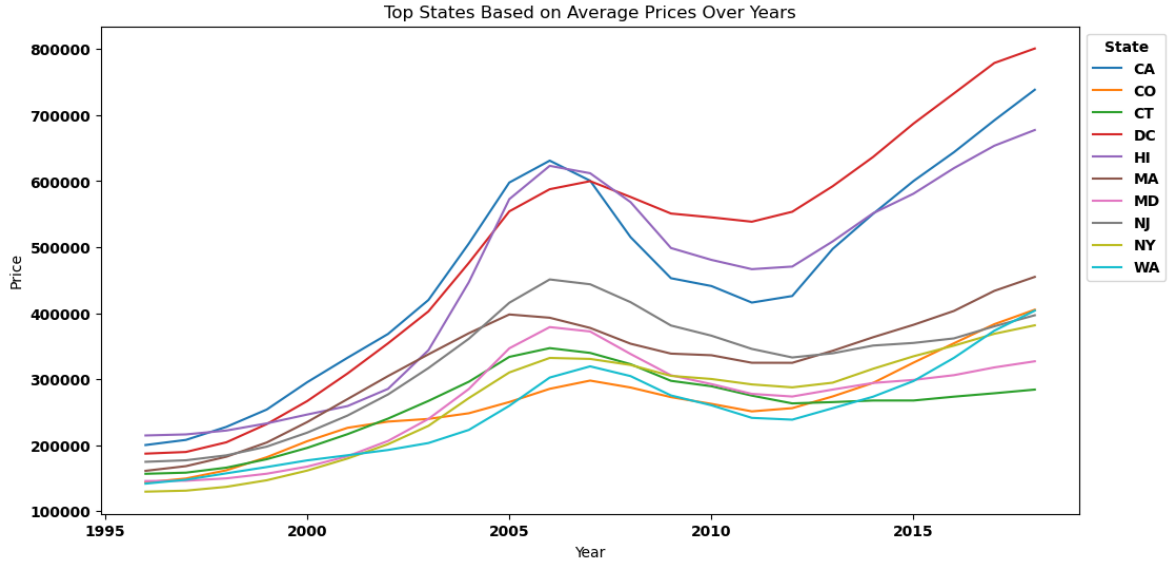
Out[40]:

	RegionID	Zipcode	SizeRank	City	State	Metro	CountyName	%ROI	ROIPrice	
0	84654	60657	1	Chicago	IL	Chicago	Cook	2.596098	212299.0	1
1	90668	75070	2	McKinney	TX	Dallas-Fort Worth	Collin	5.287075	119399.0	1
2	91982	77494	3	Katy	TX	Houston	Harris	3.281773	83199.0	1
3	84616	60614	4	Chicago	IL	Chicago	Cook	2.296917	241599.0	1
4	93144	79936	5	El Paso	TX	El Paso	El Paso	-0.009141	-101.0	1

4.3.1 Top states based on average prices over years

The cell below seeks to visualize top states with the highest property valuve over a period of time.

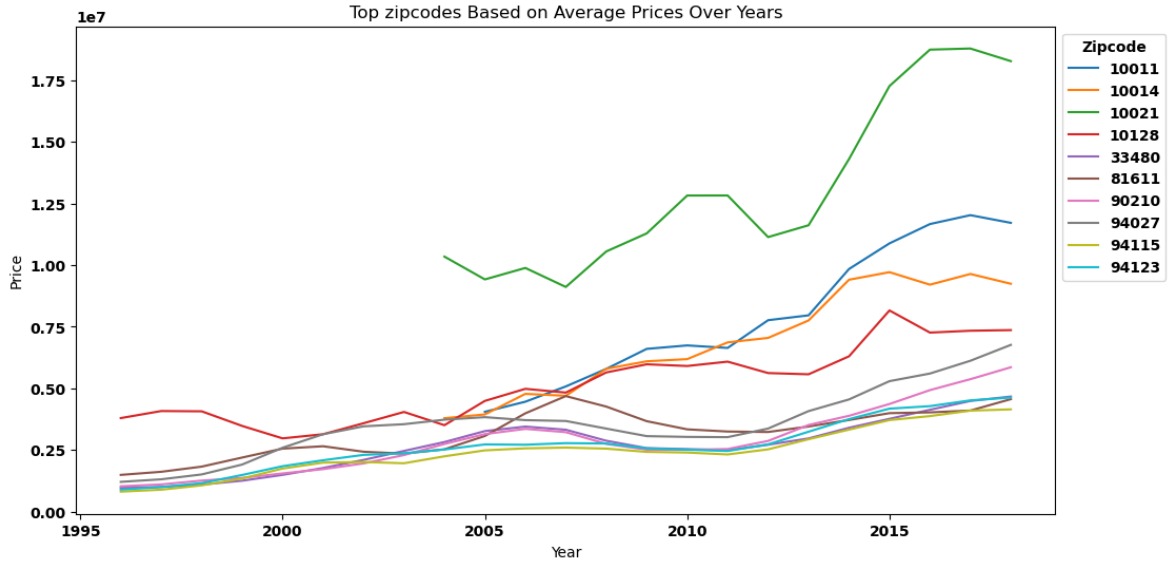
In [41]:
```
#Checking top 10 states based on average prices per year

# Calculate average prices by state
average_prices_by_state = data.groupby(['Year', 'State'])['Price'].mean().res

# Get top states based on average prices
top_states = average_prices_by_state.groupby('State')['Price'].mean().sort_va

# Filter the DataFrame to include only top states
df_top_states = average_prices_by_state[average_prices_by_state['State'].isin

# Plotting the data
plt.figure(figsize=(12, 6))
sns.lineplot(x='Year', y='Price', hue='State', data=df_top_states, ci=None)
plt.title('Top States Based on Average Prices Over Years')
plt.legend(title='State', bbox_to_anchor=(1, 1))
plt.show()
```



- As observed, the percentage of return on investents for the top ten states had a crash in 2008 to 2012 and then from there they all have an upward linear trend. This also depicts the rising trent has been from 2012 onwards meaning 2018 remains highest in terms of investments returns.

4.3.2 Top zipcodes Based on Average Prices Over Years

In [42]:
```
#Checking top 10 Zipcodes based on average prices per year

# Calculate average prices by state
average_prices_by_zipcode = data.groupby(['Year', 'Zipcode'])['Price'].mean()

# Get top states based on average prices
top_zipcodes = average_prices_by_zipcode.groupby('Zipcode')['Price'].mean().s

# Filter the DataFrame to include only top states
df_top_zipcodes = average_prices_by_zipcode[average_prices_by_zipcode['Zipcod

# Plotting the data
plt.figure(figsize=(12, 6))
sns.lineplot(x='Year', y='Price', hue='Zipcode', data=df_top_zipcodes, ci=Non
plt.title('Top zipcodes Based on Average Prices Over Years')
plt.legend(title='Zipcode', bbox_to_anchor=(1, 1))
plt.show()
```

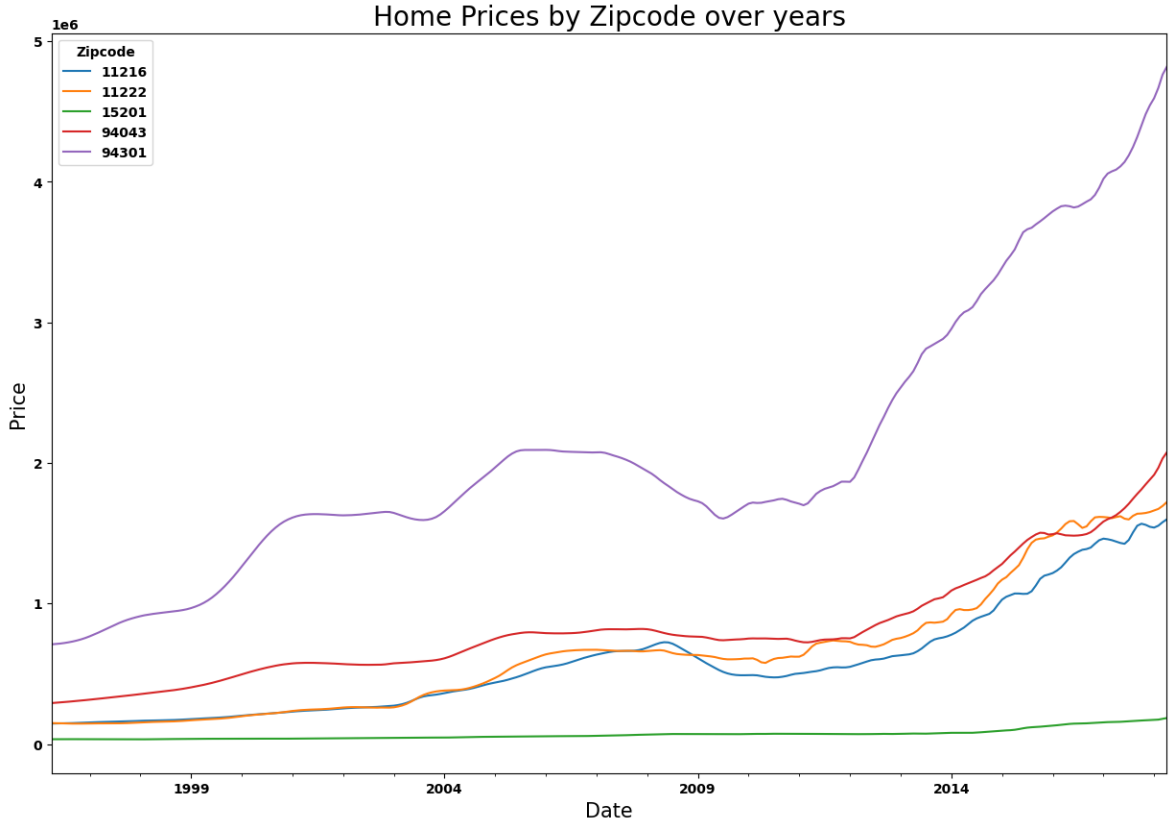


- From the above trend curve the zip code 10021 remains high in terms of valuation. over time there has been slight rising of prices in other zipcodes as from 2012 upto year 2018

4.3.3 Home Prices by Zipcode over years

- The code below filters the best 5 zipcodes based on return on investment and visualize the result from 1996 to 2018 tp show the trend.

In [43]:
```
# Filter data for the top 5 zipcodes based on %ROI
zipcodes = data.sort_values('%ROI', ascending=False)['Zipcode'].unique()[:5]
top_5 = data[data['Zipcode'].isin(zipcodes)]

# Group data by date and zipcode, and calculate the mean price for each group
grouped = top_5.groupby(['Date', 'Zipcode']).mean().reset_index()

# Pivot the data to get the zipcodes as columns and the dates as rows
pivoted = grouped.pivot(index='Date', columns='Zipcode', values='Price')

# Plot the data as a line graph
pivoted.plot(kind='line', figsize=(15,10))
plt.title('Home Prices by Zipcode over years',fontsize=20)
plt.xlabel('Date',fontsize=15)
plt.ylabel('Price',fontsize=15)
plt.show()
```



- Based on the observation above, its notice that zipcodes potrays steady rise from 1996 with low fluctuations. from 2008 there has been decline and from 2012 steady rise is witnessed. 94301 remains the zipcode with higher reurn than the other 5 zipcodes. .

Step 5: Data Preprocessing

When working with time series models, it's crucial to assume that the data is stationary. This assumption implies that the mean, variance, and autocorrelation of the data remain constant over time for each lag.

Stationary time series data is essential for efficient model development. Prior to modeling, a thorough assessment of data stationarity will be conducted using the following methods:

1. Dickey-Fuller Test: The Dickey-Fuller test will be employed to assess the stationarity of the data. This statistical test helps determine if a unit root is present in the series, which is indicative of non-stationarity.
2. Rolling Mean Analysis: Additionally, a rolling mean analysis will be performed. This involves calculating the mean over a sliding window of observations. Fluctuations in the rolling mean indicate non-stationarity.

In cases where the data is identified as non-stationary, a differencing technique will be applied. Differencing involves computing the difference between consecutive observations. This process helps transform the data into a stationary form, enabling more accurate modeling and analysis.

5.1 Checking for Stationarity

In [44]:
```
# Filter data for the top 5 zipcodes based on %ROI
zipcode = data.sort_values('%ROI', ascending=False)['Zipcode'].unique()[:5]
top_5 = data[data['Zipcode'].isin(zipcode)]

# Group data by date and zipcode, and calculate the mean price for each group
grouped_5 = top_5.groupby(['Date', 'Zipcode']).mean().reset_index()
final_df = grouped_5[grouped_5.Date >= "2005-01-01"]

final_df.head()
```

Out[44]:

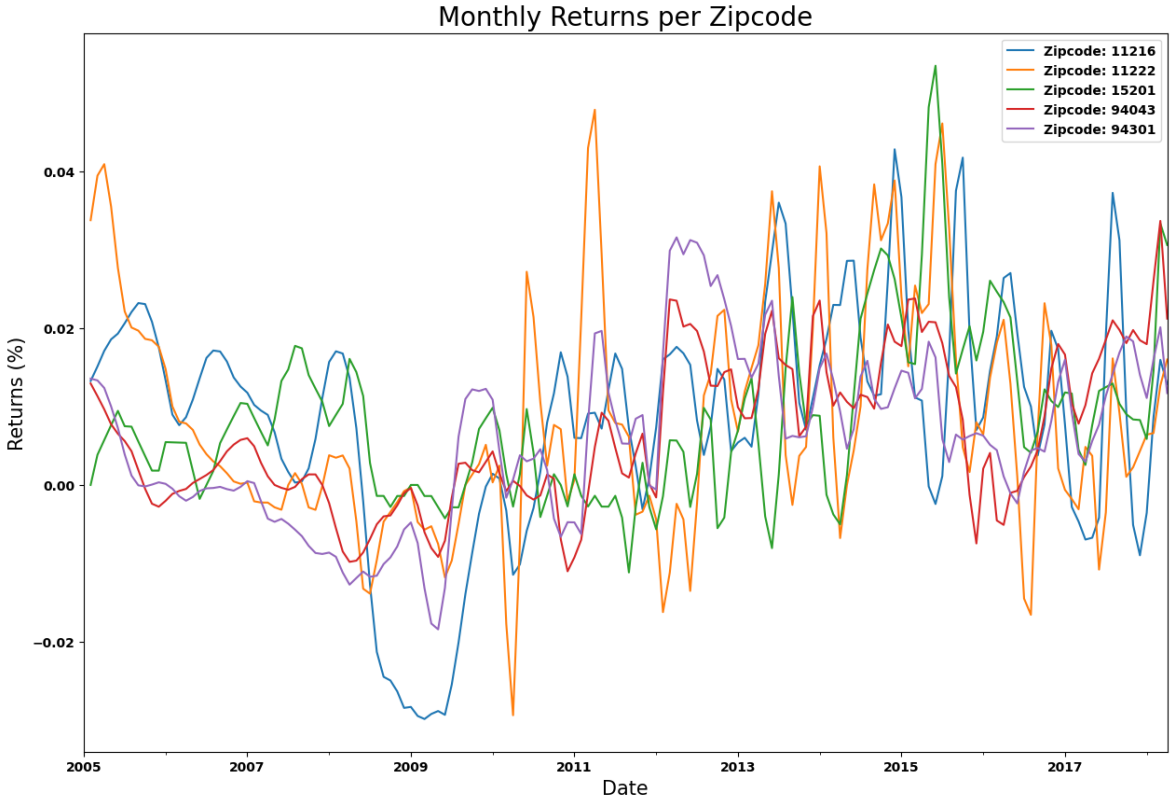
	Date	Zipcode	RegionID	SizeRank	%ROI	ROIPrice	Price	Year
525	2005-01-01	11216	62027.0	476.0	11.281218	987799.0	437500.0	2005.0
526	2005-01-01	11222	62033.0	1156.0	11.743229	1087499.0	470800.0	2005.0
527	2005-01-01	15201	63932.0	6564.0	11.258385	114299.0	52000.0	2005.0
528	2005-01-01	94043	97530.0	2581.0	11.729368	1310199.0	749200.0	2005.0
529	2005-01-01	94301	97691.0	5739.0	12.076632	3091499.0	1967900.0	2005.0

In [45]:
```
TS_zc5 = final_df.drop(['RegionID', 'SizeRank', '%ROI', 'ROIPrice'],axis=1)

TS_zc5 = TS_zc5.set_index('Date')
print('Time series data for the 5 zipcodes:\n',TS_zc5.head())

#Create individualized time series for each zipcode.
#List containing the 5 different time series.
df_ts = []
for zipcode in TS_zc5.Zipcode.unique():
    # Create separate dataframes for each zipcode with a monthly frequency.
    df_zip = TS_zc5[TS_zc5['Zipcode']==zipcode].asfreq('MS')
    df_ts.append(df_zip)
    print('\nZipcode 48894 time series:')
    df_ts[0].head()
```

```
Time series data for the 5 zipcodes:
            Zipcode      Price    Year
Date
2005-01-01   11216   437500.0  2005.0
2005-01-01   11222   470800.0  2005.0
2005-01-01   15201    52000.0  2005.0
2005-01-01   94043   749200.0  2005.0
2005-01-01   94301  1967900.0  2005.0

Zipcode 48894 time series:

Zipcode 48894 time series:

Zipcode 48894 time series:

Zipcode 48894 time series:

Zipcode 48894 time series:
```

- It can be observed like before that there was a dip in the prices due to the 2008 market crash and then the prices continued to rise over time from 2012 all the way to 2018.
- A new column called ret is created to check the returns per month

In [46]:
```
# creating a column called "ret" representing monthly returns on investment
for zc in range(len(df_ts)):
    df_ts[zc]['ret']=np.nan*len(df_ts[zc])
    for i in range(len(df_ts[zc])-1):
        df_ts[zc]['ret'][i+1]= (df_ts[zc].Price.iloc[i+1] / df_ts[zc].Price.i

#Plot the monthly returns of each zipcode
for i in range(len(df_ts)):
    df_ts[i].ret.plot(figsize=(15,10),label=f"Zipcode: {df_ts[i].Zipcode[0]}"
    plt.title(f'Monthly Returns per Zipcode',fontsize=20)
    plt.xlabel('Date',fontsize=15)
    plt.ylabel('Returns (%)',fontsize=15)
    plt.legend(loc='best')
```



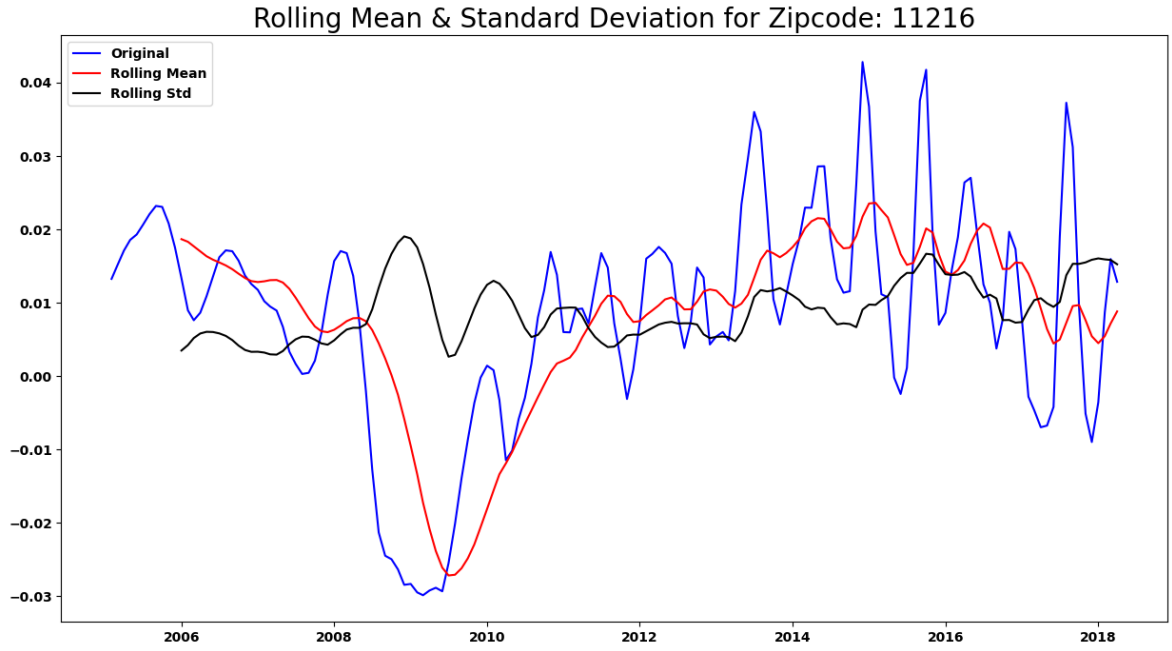
- From the obesrvation there is no clear trend which could indicate stationarity. The data seemes to noo_stationary.This calls for further tests to be carried out to determine this.

5.1.1 Rolling Mean and standard deviation to check for stationarity

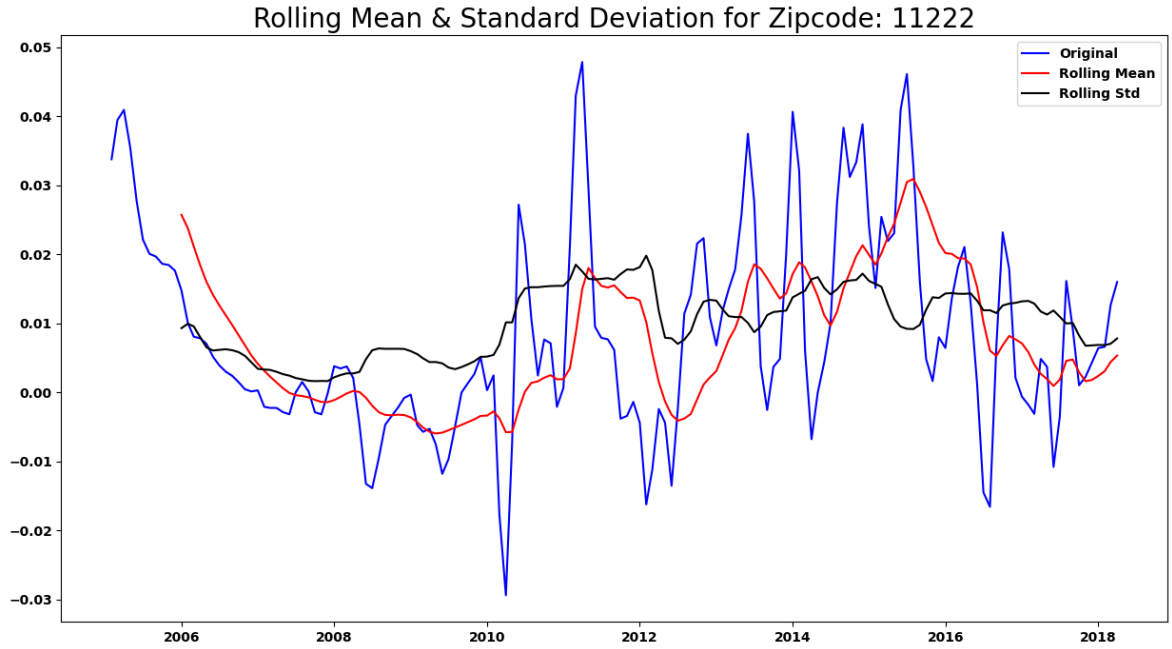
- Below code generates visualization to check for stationarity of the five zipcodes

In [47]:
```
# Testing for stationarity of the zipcodes

# Plot each of the zipcodes' returns with their respective rolling mean and r
for i in range(len(df_ts)):
    rollingmean = df_ts[i].ret.rolling(window=12, center=False).mean()
    rollingstd = df_ts[i].ret.rolling(window=12, center=False).std()
    fig = plt.figure(figsize=(15, 8))
    original = plt.plot(df_ts[i].ret, color="blue", label="Original")
    mean = plt.plot(rollingmean, color="red", label="Rolling Mean")
    std = plt.plot(rollingstd, color="black", label="Rolling Std")
    plt.legend(loc="best")
    plt.title(f'Rolling Mean & Standard Deviation for Zipcode: {df_ts[i].Zipc

    # Perform ADF test
    adf_result = adfuller(df_ts[i].ret.dropna())
    print('ADF Test Results for Zipcode:', df_ts[i].Zipcode[0])
    print('Test Statistic:', adf_result[0])
    print('p-Value:', adf_result[1])
    print('Used Lags:', adf_result[2])
    print('Observations:', adf_result[3])
    print('Critical Values:')
    for key, value in adf_result[4].items():
        print(f'   {key}: {value}')

    plt.show()
```

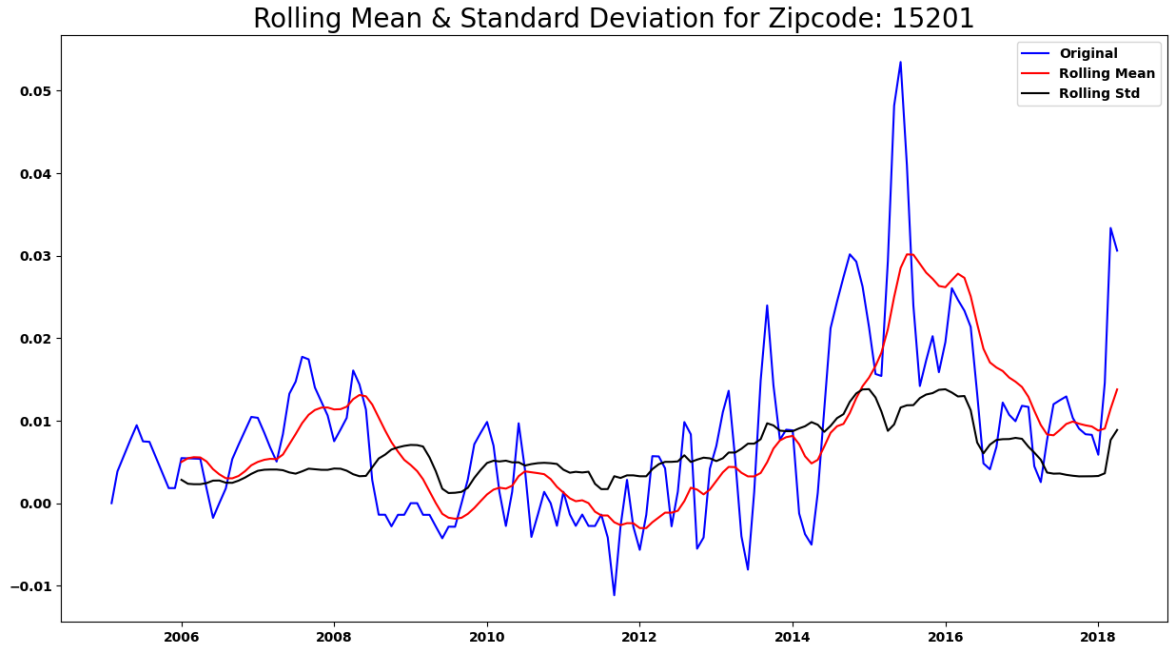
```
ADF Test Results for Zipcode: 11216
Test Statistic: -2.3928540231791966
p-Value: 0.14376347414448093
Used Lags: 8
Observations: 150
Critical Values:
   1%: -3.474714913481481
   5%: -2.881008708148148
   10%: -2.5771508444444446
```



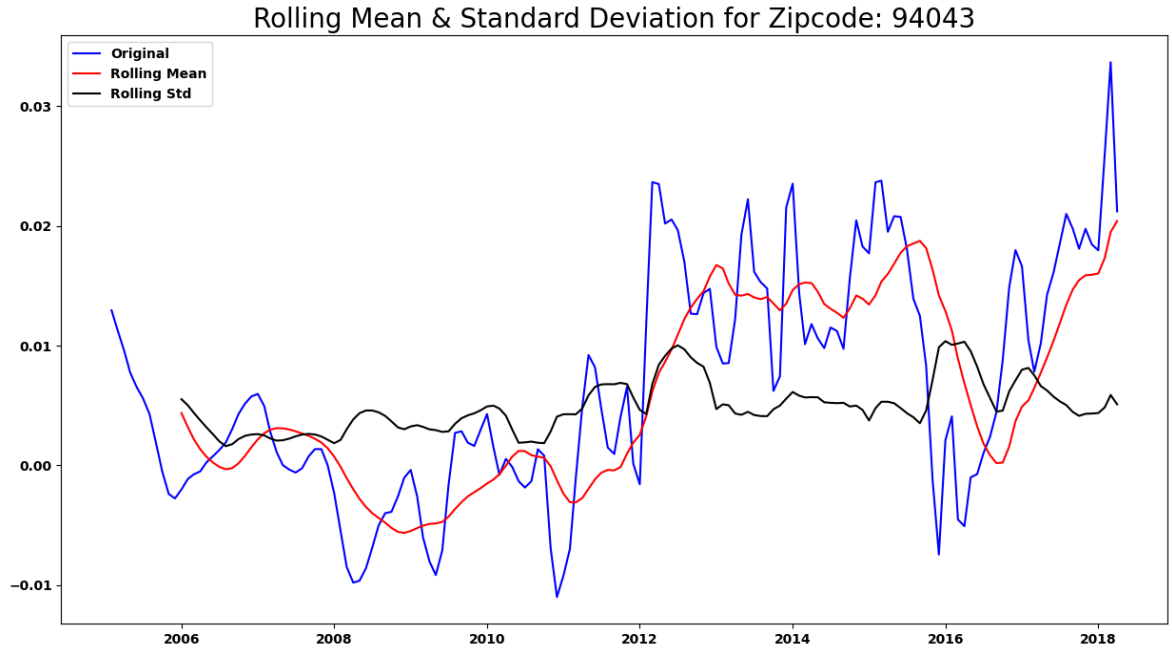
```
ADF Test Results for Zipcode: 11222
Test Statistic: -2.5337723538387196
p-Value: 0.10747185863643116
Used Lags: 14
Observations: 144
Critical Values:
   1%: -3.476597917537401
   5%: -2.8818291230495543
   10%: -2.5775887982253085
```



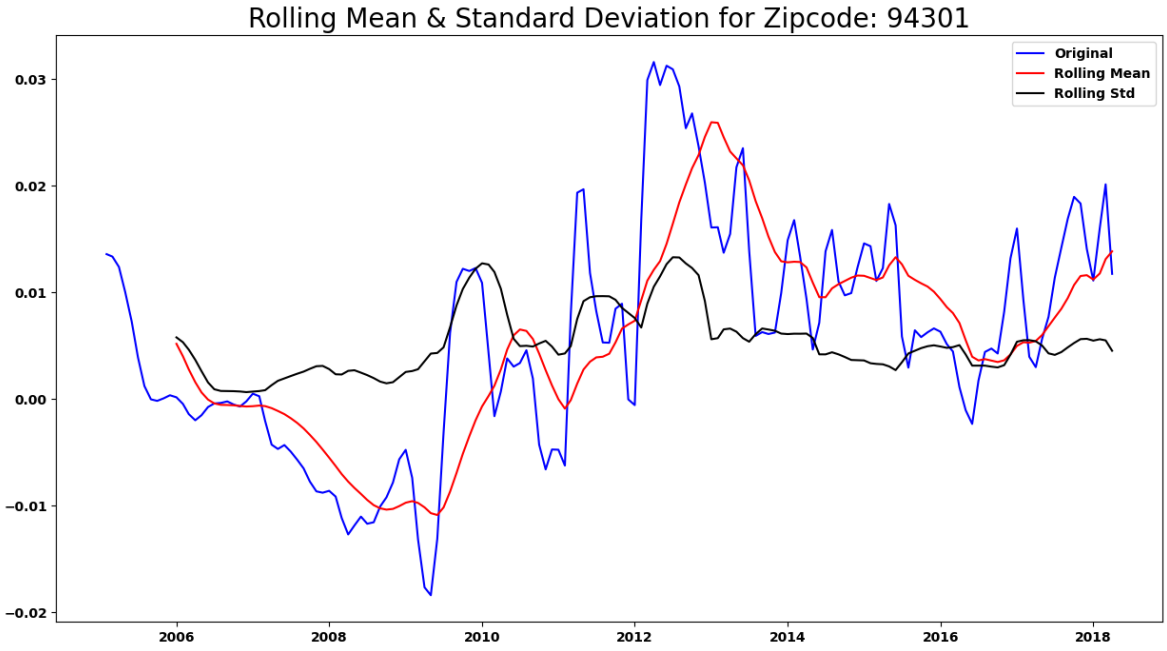

```
ADF Test Results for Zipcode: 15201
Test Statistic: -2.5970509916760363
p-Value: 0.09360830058314773
Used Lags: 12
Observations: 146
Critical Values:
   1%: -3.4759527332353084
   5%: -2.881548071241103
   10%: -2.577438765246763
```




```
ADF Test Results for Zipcode: 94043
Test Statistic: -1.764657775467016
p-Value: 0.39814384859424407
Used Lags: 13
Observations: 145
Critical Values:
   1%: -3.476273058920005
   5%: -2.881687616548444
   10%: -2.5775132580261593
```



```
ADF Test Results for Zipcode: 94301
Test Statistic: -1.9796174752787588
p-Value: 0.29558100750588956
Used Lags: 13
Observations: 145
Critical Values:
   1%: -3.476273058920005
   5%: -2.881687616548444
   10%: -2.5775132580261593
```



- From the graphs above there are some states that exhibit non-stationarity but to be certain, a Dickey Fuller test is performed to make a clear picture of it.

5.1.2 Dicky-fuller Test

- Testing for stationerty using dicky-fuller test

In [48]:
```
# performing Dicky-fuller test for stationarity

for i in range(5):
    results = adfuller(df_ts[i].ret.dropna())
    print(f'ADFuller test p-value for zipcode: {df_ts[i].Zipcode[0]}')
    print('p-value:',results[1])

    if results[1]>0.05:
        print('Fail to reject the null hypothesis. Data is not stationary.\n'
    else:
        print('Reject the null hypothesis. Data is stationary.\n')
```

```
ADFuller test p-value for zipcode: 11216
p-value: 0.14376347414448093
Fail to reject the null hypothesis. Data is not stationary.

ADFuller test p-value for zipcode: 11222
p-value: 0.10747185863643116
Fail to reject the null hypothesis. Data is not stationary.

ADFuller test p-value for zipcode: 15201
p-value: 0.09360830058314773
Fail to reject the null hypothesis. Data is not stationary.

ADFuller test p-value for zipcode: 94043
p-value: 0.39814384859424407
Fail to reject the null hypothesis. Data is not stationary.

ADFuller test p-value for zipcode: 94301
p-value: 0.29558100750588956
Fail to reject the null hypothesis. Data is not stationary.
```

- From the above analysis, the Dickey Fuller test shows that **11216** has p_value of 0.1437, **11222** has p_value of 0.1074, **15201** has p_value of 0.093608, **94043** has a p_value of 0.3981 and **94301** has a p_value of 0.2955 which indicates that non of the zipcode has meet the criteria of stationarity(P_value < 0.05) and this may validate the procedure of differencing to bring the points to stationarity for further analysis.

5.1.3 Differencing of the dataset

The code below will perform differencing process to bring data to stationarity. It will conduct the iterations according to the number of zipcodes

In [49]:
```
# differencing the non stationary zip codes

for i in [0,1,2,3,4]:
    #Perform adfuller test and drop NaN values created when calculating month
    results = adfuller(df_ts[i].ret.diff().dropna())
    print(f'ADFuller test p-value for zipcode: {df_ts[i].Zipcode[0]}')
    print('p-value:',results[1])

    if results[1]>0.05:
        print('Fail to reject the null hypothesis. Data is not stationary.\n'
    else:
        print('Reject the null hypothesis. Data is stationary.\n')
```

```
ADFuller test p-value for zipcode: 11216
p-value: 0.04242671744182503
Reject the null hypothesis. Data is stationary.

ADFuller test p-value for zipcode: 11222
p-value: 0.0004172981075063474
Reject the null hypothesis. Data is stationary.

ADFuller test p-value for zipcode: 15201
p-value: 0.014703632964944808
Reject the null hypothesis. Data is stationary.

ADFuller test p-value for zipcode: 94043
p-value: 2.403066325520023e-05
Reject the null hypothesis. Data is stationary.

ADFuller test p-value for zipcode: 94301
p-value: 0.005821932027030528
Reject the null hypothesis. Data is stationary.
```

- From the differencing process we can observe that all the zipcodes p_values are less than alpha_value of 0.05 hence indicating that the dataset has met stationarity aspect.

Step 6: Modelling

- Since the aim is to identify the top five zipcodes to invest in, there will be five different models for each of the top five zipcodes to forecast their prices and thus give the investors clear path to make informed decision.

- The code below calculates the differences between consecutive values then creating individual time series for each set of data. Any null values are also dropped to ensure the data for modeling and visualization is complete and suitable for the intended purpose.

In [50]:
```
# creating individual time series
ts_11216 = df_ts[1].ret.dropna()
ts_11222 = df_ts[0].ret.diff().dropna()
ts_15201 = df_ts[2].ret.diff().dropna()
ts_94043 = df_ts[3].ret.diff().dropna()
ts_94301 = df_ts[4].ret.diff().dropna()
```

6.1 Baseline Model - ARIMA

ARIMA is a popular time series forecasting model that combines autoregressive, differencing, and moving average components.

Its use for price prediction is justified due to its ability to capture time-dependent patterns, handle nonlinear relationships, address stationarity, provide interpretability, and its well-established framework in time series analysis.

- Below function acf_pacf creates and displays Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots for a given time series dataset using the plot_acf and plot_pacf functions. These plots are used to determine the appropriate values of the ARIMA model parameters ("p" "d"and "q").

6.1.1 ACF & PACF plots per Zipcode

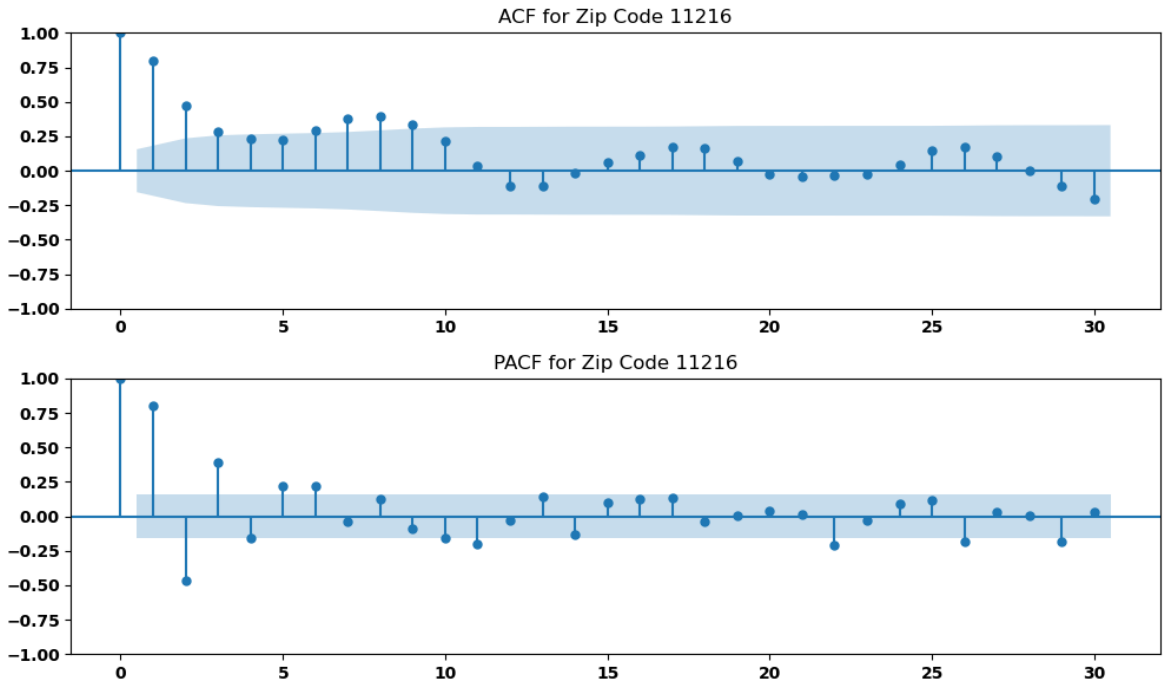
Through plotting the ACF and PACF plots, we identify potential patterns, trends, and the order of autoregressive (AR) and moving average (MA) components in our time series data. The figures we get from this plots can be utilised in the order parameter of our ARIMA model. ACF measures the correlation between a time series and its lagged values. It shows how each observation in the series is correlated with its previous observations at various lags.

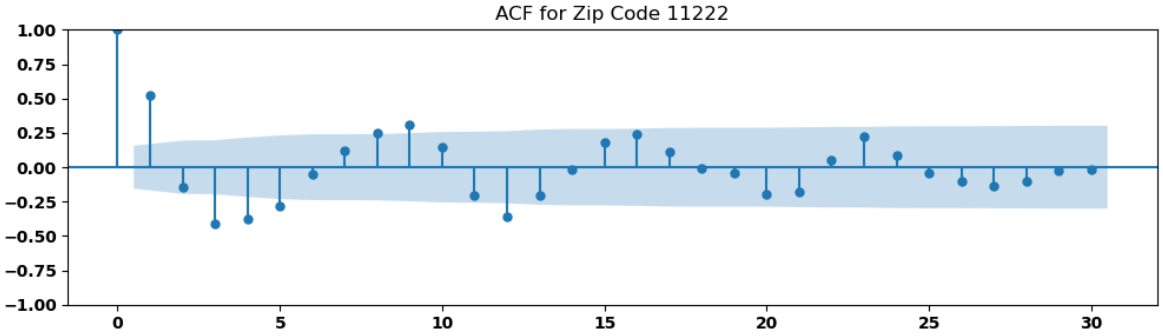
If the ACF plot shows a sharp drop after a certain lag, it indicates that the values beyond that lag are not significantly correlated with the current value. If the ACF plot shows a slow decay, it suggests a high degree of autocorrelation and indicates the presence of a possible autoregressive (AR) component. PACF measures the correlation between a time series and its lagged values after removing the effects of intervening lags. It gives the direct relationship between an observation and its lagged values without the influence of the intermediate observations.

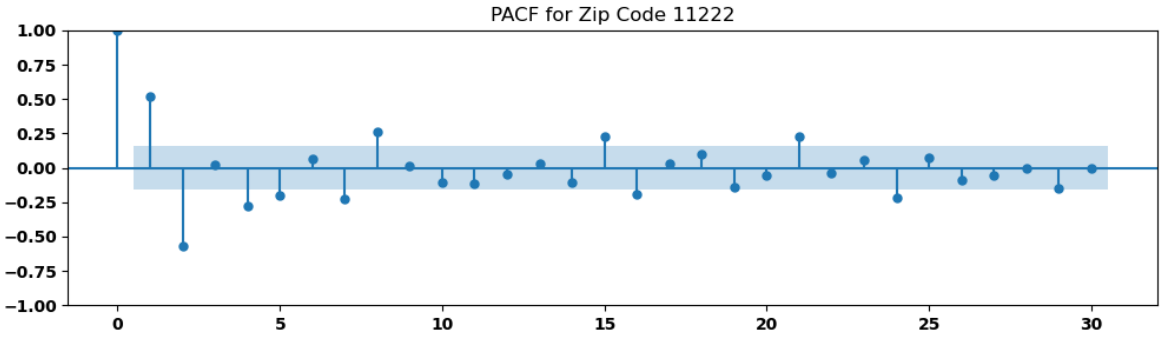
If the PACF plot shows a sharp drop after a certain lag, it indicates that there is no significant correlation beyond that lag. If the PACF plot shows a slow decay, it suggests a high degree of partial autocorrelation and indicates the presence of a possible moving average (MA) component.

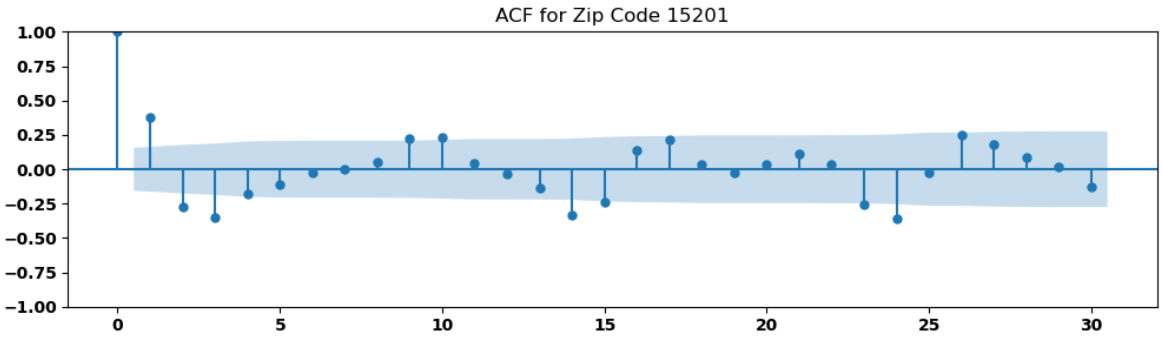
In [51]:
```
# defining a function that plots acf and pacf plots
def acf_pacf(df,alags=40,plags=40):
    #Create figure
    fig,(ax1,ax2) = plt.subplots(2,1,figsize=(15,10))
    #Make ACF plot
    plot_acf(df,lags=alags, zero=False,ax=ax1)
    #Make PACF plot
    plot_pacf(df,lags=plags, ax=ax2)
    plt.show()
```

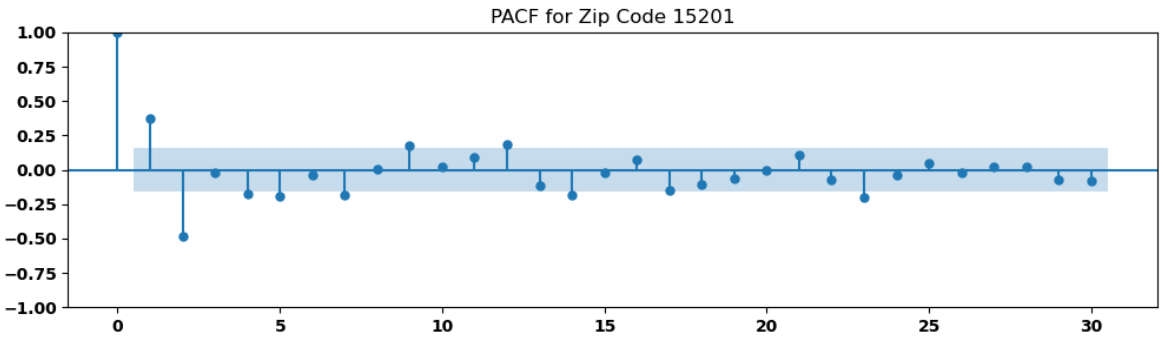
In [52]:
```
# # plotting acf and pacf for zipcode 85035
# from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Plot ACF and PACF for each zip code
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data
    plt.figure(figsize=(10, 6))

    # ACF plot
    plt.subplot(2, 1, 1)
    plot_acf(ts, lags=30, ax=plt.gca())
    plt.title(f'ACF for Zip Code {zipcode}')

    # PACF plot
    plt.subplot(2, 1, 2)
    plot_pacf(ts, lags=30, ax=plt.gca())
    plt.title(f'PACF for Zip Code {zipcode}')

    plt.tight_layout()
    plt.show()
```

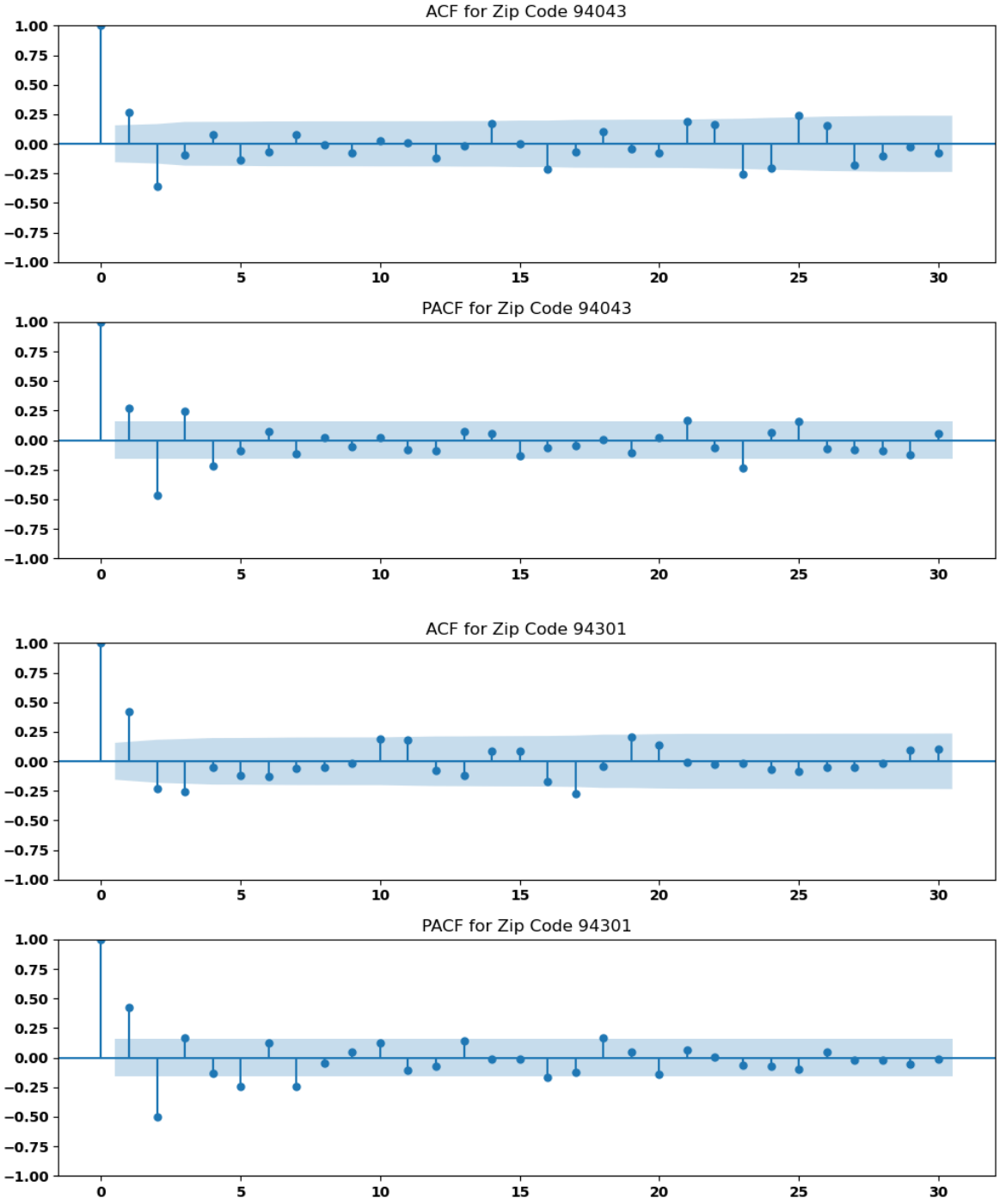



ACF for Zip Code 11222
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30


PACF for Zip Code 11222
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30


ACF for Zip Code 15201
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30


PACF for Zip Code 15201
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30



- The code below fits an ARIMA model to multiple time series data corresponding to different zip codes. It uses the pmdarima library, which provides an interface for fitting ARIMA models with various configurations.

In [53]:
```
#Fitting the model into the zipcodes


# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']


# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Calculate the split index for training and testing
    train_size = 0.80  # Leaving approximately 3 years for test size
    split_idx = round(len(ts) * train_size)

    # Split the time series data into train and test sets
    train = ts.iloc[:split_idx]
    test = ts.iloc[split_idx:]

    # Fit the ARIMA model with fixed parameters to the training data
    model = pm.auto_arima(ts, trace=True, error_action='ignore', suppress_war
    model.fit(train)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1157.995, Time=0.72 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-889.320, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1075.783, Time=0.27 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1048.416, Time=0.30 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-846.771, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1168.942, Time=0.52 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1123.269, Time=0.30 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-1170.572, Time=1.05 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=inf, Time=1.15 sec
 ARIMA(1,0,2)(0,0,2)[12] intercept   : AIC=-1159.154, Time=0.95 sec
 ARIMA(2,0,1)(0,0,2)[12] intercept   : AIC=inf, Time=0.89 sec
 ARIMA(3,0,2)(0,0,2)[12] intercept   : AIC=-1181.565, Time=1.12 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-1181.422, Time=0.60 sec
 ARIMA(3,0,2)(1,0,2)[12] intercept   : AIC=inf, Time=1.30 sec
 ARIMA(3,0,2)(1,0,1)[12] intercept   : AIC=-1170.854, Time=0.70 sec
 ARIMA(3,0,1)(0,0,2)[12] intercept   : AIC=inf, Time=1.04 sec
 ARIMA(4,0,2)(0,0,2)[12] intercept   : AIC=-1184.421, Time=1.40 sec
 ARIMA(4,0,2)(0,0,1)[12] intercept   : AIC=-1191.108, Time=0.79 sec
 ARIMA(4,0,2)(0,0,0)[12] intercept   : AIC=-1129.372, Time=0.48 sec
 ARIMA(4,0,2)(1,0,1)[12] intercept   : AIC=-1168.681, Time=0.98 sec
 ARIMA(4,0,2)(1,0,0)[12] intercept   : AIC=-1147.064, Time=0.83 sec
 ARIMA(4,0,2)(1,0,2)[12] intercept   : AIC=-1183.013, Time=1.67 sec
 ARIMA(4,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.72 sec
 ARIMA(5,0,2)(0,0,1)[12] intercept   : AIC=-1179.743, Time=0.89 sec
 ARIMA(4,0,3)(0,0,1)[12] intercept   : AIC=-1183.974, Time=0.74 sec
 ARIMA(3,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.58 sec
 ARIMA(3,0,3)(0,0,1)[12] intercept   : AIC=-1190.467, Time=0.66 sec
 ARIMA(5,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.75 sec
 ARIMA(5,0,3)(0,0,1)[12] intercept   : AIC=-1188.879, Time=0.79 sec
 ARIMA(4,0,2)(0,0,1)[12]             : AIC=-1187.420, Time=0.71 sec

Best model:  ARIMA(4,0,2)(0,0,1)[12] intercept
Total fit time: 22.994 seconds
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1231.465, Time=0.51 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1128.108, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1194.595, Time=0.17 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.38 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1130.108, Time=0.02 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1255.026, Time=0.30 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1244.087, Time=0.23 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-1227.897, Time=0.99 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1234.600, Time=0.29 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-1244.515, Time=0.69 sec
 ARIMA(1,0,2)(0,0,1)[12] intercept   : AIC=-1222.946, Time=0.43 sec
 ARIMA(2,0,1)(0,0,1)[12] intercept   : AIC=-1253.637, Time=0.28 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-1233.106, Time=0.46 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1250.943, Time=0.25 sec
 ARIMA(1,0,1)(0,0,1)[12] intercept   : AIC=-1243.025, Time=0.26 sec
 ARIMA(1,0,3)(0,0,1)[12] intercept   : AIC=-1262.693, Time=0.41 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1257.503, Time=0.49 sec
 ARIMA(1,0,3)(1,0,1)[12] intercept   : AIC=-1264.552, Time=0.68 sec
 ARIMA(1,0,3)(1,0,0)[12] intercept   : AIC=-1266.225, Time=0.57 sec
 ARIMA(1,0,3)(2,0,0)[12] intercept   : AIC=-1271.653, Time=1.05 sec
 ARIMA(1,0,3)(2,0,1)[12] intercept   : AIC=-1268.744, Time=1.22 sec
 ARIMA(0,0,3)(2,0,0)[12] intercept   : AIC=-1280.559, Time=0.87 sec
```

```
 ARIMA(0,0,3)(1,0,0)[12] intercept   : AIC=-1278.059, Time=0.44 sec
 ARIMA(0,0,3)(2,0,1)[12] intercept   : AIC=-1275.152, Time=0.91 sec
 ARIMA(0,0,3)(1,0,1)[12] intercept   : AIC=-1283.417, Time=0.63 sec
 ARIMA(0,0,3)(0,0,1)[12] intercept   : AIC=-1280.626, Time=0.31 sec
 ARIMA(0,0,3)(1,0,2)[12] intercept   : AIC=-1277.138, Time=0.49 sec
 ARIMA(0,0,3)(0,0,0)[12] intercept   : AIC=-1263.635, Time=0.32 sec
 ARIMA(0,0,3)(0,0,2)[12] intercept   : AIC=-1297.068, Time=1.02 sec
 ARIMA(0,0,2)(0,0,2)[12] intercept   : AIC=-1273.969, Time=0.95 sec
 ARIMA(1,0,3)(0,0,2)[12] intercept   : AIC=-1284.070, Time=1.69 sec
 ARIMA(0,0,4)(0,0,2)[12] intercept   : AIC=-1291.672, Time=1.29 sec
 ARIMA(1,0,2)(0,0,2)[12] intercept   : AIC=-1201.418, Time=0.63 sec
 ARIMA(1,0,4)(0,0,2)[12] intercept   : AIC=-1252.014, Time=0.81 sec
 ARIMA(0,0,3)(0,0,2)[12]             : AIC=-1301.067, Time=0.93 sec
 ARIMA(0,0,3)(0,0,1)[12]             : AIC=-1282.627, Time=0.28 sec
 ARIMA(0,0,3)(1,0,2)[12]             : AIC=-1294.587, Time=1.22 sec
 ARIMA(0,0,3)(1,0,1)[12]             : AIC=-1286.180, Time=0.56 sec
 ARIMA(0,0,2)(0,0,2)[12]             : AIC=-1276.076, Time=0.79 sec
 ARIMA(1,0,3)(0,0,2)[12]             : AIC=-1294.180, Time=1.13 sec
 ARIMA(0,0,4)(0,0,2)[12]             : AIC=-1299.146, Time=1.22 sec
 ARIMA(1,0,2)(0,0,2)[12]             : AIC=-1250.085, Time=0.87 sec
 ARIMA(1,0,4)(0,0,2)[12]             : AIC=inf, Time=1.13 sec

Best model:  ARIMA(0,0,3)(0,0,2)[12]
Total fit time: 28.196 seconds
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1277.258, Time=0.66 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1203.702, Time=0.05 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1223.715, Time=0.17 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1251.875, Time=0.46 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1205.491, Time=0.04 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1277.380, Time=0.92 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1279.175, Time=0.35 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1277.323, Time=0.61 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1237.125, Time=0.30 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1271.182, Time=0.29 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1269.039, Time=0.33 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1274.829, Time=0.38 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-1251.991, Time=0.22 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=inf, Time=0.30 sec
 ARIMA(3,0,1)(0,0,0)[12] intercept   : AIC=-1269.536, Time=0.31 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1272.866, Time=0.21 sec
 ARIMA(2,0,2)(0,0,0)[12]             : AIC=-1279.733, Time=0.11 sec
 ARIMA(2,0,2)(1,0,0)[12]             : AIC=-1277.713, Time=0.23 sec
 ARIMA(2,0,2)(0,0,1)[12]             : AIC=-1279.502, Time=0.66 sec
 ARIMA(2,0,2)(1,0,1)[12]             : AIC=-1275.240, Time=0.16 sec
 ARIMA(1,0,2)(0,0,0)[12]             : AIC=-1268.122, Time=0.18 sec
 ARIMA(2,0,1)(0,0,0)[12]             : AIC=-1273.066, Time=0.09 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-1273.476, Time=0.32 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1272.585, Time=0.11 sec
 ARIMA(1,0,1)(0,0,0)[12]             : AIC=-1253.917, Time=0.13 sec
 ARIMA(1,0,3)(0,0,0)[12]             : AIC=inf, Time=0.14 sec
 ARIMA(3,0,1)(0,0,0)[12]             : AIC=-1271.190, Time=0.22 sec
 ARIMA(3,0,3)(0,0,0)[12]             : AIC=-1271.645, Time=0.15 sec

Best model:  ARIMA(2,0,2)(0,0,0)[12]
Total fit time: 8.125 seconds
Performing stepwise search to minimize aic
```

```
ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1377.512, Time=0.45 sec
ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1309.933, Time=0.05 sec
ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1322.291, Time=0.13 sec
ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1368.182, Time=0.20 sec
ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1311.903, Time=0.04 sec
ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1399.110, Time=0.44 sec
ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1409.732, Time=0.38 sec
ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1383.004, Time=0.25 sec
ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1371.324, Time=0.28 sec
ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1397.531, Time=0.37 sec
ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1396.316, Time=0.43 sec
ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1426.367, Time=0.47 sec
ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-1421.228, Time=0.37 sec
ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1421.426, Time=0.67 sec
ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-1423.383, Time=0.88 sec
```

```
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1413.846, Time=0.43 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1422.369, Time=0.42 sec
 ARIMA(2,0,4)(0,0,0)[12] intercept   : AIC=-1421.801, Time=0.62 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-1423.410, Time=0.49 sec
 ARIMA(3,0,4)(0,0,0)[12] intercept   : AIC=-1418.904, Time=0.51 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1428.240, Time=0.28 sec
 ARIMA(2,0,3)(1,0,0)[12]             : AIC=-1426.822, Time=0.65 sec
 ARIMA(2,0,3)(0,0,1)[12]             : AIC=-1423.416, Time=0.32 sec
 ARIMA(2,0,3)(1,0,1)[12]             : AIC=-1425.821, Time=0.86 sec
 ARIMA(1,0,3)(0,0,0)[12]             : AIC=-1418.236, Time=0.23 sec
 ARIMA(2,0,2)(0,0,0)[12]             : AIC=-1389.958, Time=0.13 sec
 ARIMA(3,0,3)(0,0,0)[12]             : AIC=-1430.322, Time=0.48 sec
 ARIMA(3,0,3)(1,0,0)[12]             : AIC=-1428.126, Time=0.67 sec
 ARIMA(3,0,3)(0,0,1)[12]             : AIC=-1420.785, Time=0.31 sec
 ARIMA(3,0,3)(1,0,1)[12]             : AIC=-1425.764, Time=0.76 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-1390.203, Time=0.10 sec
 ARIMA(4,0,3)(0,0,0)[12]             : AIC=-1412.109, Time=0.33 sec
 ARIMA(3,0,4)(0,0,0)[12]             : AIC=-1407.779, Time=0.18 sec
 ARIMA(2,0,4)(0,0,0)[12]             : AIC=-1425.922, Time=0.42 sec
 ARIMA(4,0,2)(0,0,0)[12]             : AIC=-1407.389, Time=0.12 sec
 ARIMA(4,0,4)(0,0,0)[12]             : AIC=-1412.332, Time=0.29 sec

Best model:  ARIMA(3,0,3)(0,0,0)[12]
Total fit time: 14.049 seconds
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1394.761, Time=0.69 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1294.426, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1325.440, Time=0.17 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1359.938, Time=0.44 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1296.425, Time=0.04 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1385.821, Time=0.50 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1383.008, Time=0.56 sec
 ARIMA(2,0,2)(2,0,1)[12] intercept   : AIC=-1382.589, Time=1.91 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-1388.468, Time=1.59 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1396.087, Time=0.37 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1356.165, Time=0.26 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1380.651, Time=0.32 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1377.369, Time=0.33 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1416.075, Time=0.56 sec
 ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-1412.250, Time=0.39 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1391.027, Time=0.26 sec
 ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-1387.383, Time=0.58 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1409.292, Time=0.13 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1391.863, Time=0.50 sec
 ARIMA(2,0,4)(0,0,0)[12] intercept   : AIC=-1406.863, Time=0.54 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-1411.258, Time=0.35 sec
 ARIMA(3,0,4)(0,0,0)[12] intercept   : AIC=-1394.669, Time=0.77 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1412.966, Time=0.15 sec

Best model:  ARIMA(2,0,3)(0,0,0)[12] intercept
Total fit time: 11.496 seconds
```

In [54]:
```
# Checking for the best p,d,q for each zipcode

# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Using auto_arima for each zip code
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Using auto_arima to find the best p, d, q for the model
    model = pm.auto_arima(ts, trace=True, error_action='ignore', suppress_war

    # Print model summary
    print(f"Summary for Zip Code {zipcode}:")
    print(model.summary())
    print()
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1157.995, Time=0.72 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-889.320, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1075.783, Time=0.29 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1048.416, Time=0.30 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-846.771, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1168.942, Time=0.65 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1123.269, Time=0.38 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-1170.572, Time=1.11 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=inf, Time=1.22 sec
 ARIMA(1,0,2)(0,0,2)[12] intercept   : AIC=-1159.154, Time=1.01 sec
 ARIMA(2,0,1)(0,0,2)[12] intercept   : AIC=inf, Time=0.90 sec
 ARIMA(3,0,2)(0,0,2)[12] intercept   : AIC=-1181.565, Time=1.21 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-1181.422, Time=0.63 sec
 ARIMA(3,0,2)(1,0,2)[12] intercept   : AIC=inf, Time=1.38 sec
 ARIMA(3,0,2)(1,0,1)[12] intercept   : AIC=-1170.854, Time=0.78 sec
 ARIMA(3,0,1)(0,0,2)[12] intercept   : AIC=inf, Time=1.10 sec
 ARIMA(4,0,2)(0,0,2)[12] intercept   : AIC=-1184.421, Time=1.50 sec
 ARIMA(4,0,2)(0,0,1)[12] intercept   : AIC=-1191.108, Time=0.81 sec
 ARIMA(4,0,2)(0,0,0)[12] intercept   : AIC=-1129.372, Time=0.51 sec
 ARIMA(4,0,2)(1,0,1)[12] intercept   : AIC=-1168.681, Time=0.95 sec
 ARIMA(4,0,2)(1,0,0)[12] intercept   : AIC=-1147.064, Time=0.79 sec
 ARIMA(4,0,2)(1,0,2)[12] intercept   : AIC=-1183.013, Time=1.60 sec
 ARIMA(4,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.71 sec
 ARIMA(5,0,2)(0,0,1)[12] intercept   : AIC=-1179.743, Time=0.86 sec
 ARIMA(4,0,3)(0,0,1)[12] intercept   : AIC=-1183.974, Time=0.71 sec
 ARIMA(3,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.58 sec
 ARIMA(3,0,3)(0,0,1)[12] intercept   : AIC=-1190.467, Time=0.67 sec
 ARIMA(5,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.75 sec
 ARIMA(5,0,3)(0,0,1)[12] intercept   : AIC=-1188.879, Time=0.79 sec
 ARIMA(4,0,2)(0,0,1)[12]             : AIC=-1187.420, Time=0.77 sec

Best model:  ARIMA(4,0,2)(0,0,1)[12] intercept
Total fit time: 23.805 seconds
Summary for Zip Code 11216:
                                     SARIMAX Results
==========================================================================
================
Dep. Variable:                                     y   No. Observations:
159
Model:             SARIMAX(4, 0, 2)x(0, 0, [1], 12)   Log Likelihood
604.554
Date:                              Fri, 01 Sep 2023   AIC
-1191.108
Time:                                      00:34:26   BIC
-1163.488
Sample:                                  02-01-2005   HQIC
-1179.891
                                       - 04-01-2018
Covariance Type:                               opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
intercept      0.0007      0.001      0.925      0.355      -0.001       0.0
```

```
02
ar.L1          0.2801      0.138      2.032      0.042       0.010       0.5
50
ar.L2          0.2860      0.158      1.812      0.070      -0.023       0.5
95
ar.L3          0.3376      0.130      2.599      0.009       0.083       0.5
92
ar.L4          0.0198      0.121      0.163      0.871      -0.218       0.2
58
ma.L1          1.4245      0.132     10.794      0.000       1.166       1.6
83
ma.L2          0.7424      0.098      7.559      0.000       0.550       0.9
35
ma.S.L12      -0.8808      0.107     -8.214      0.000      -1.091      -0.6
71
sigma2      2.558e-05   2.83e-06      9.024      0.000       2e-05    3.11e-
05
===================================================================================
=======
Ljung-Box (L1) (Q):                   0.14   Jarque-Bera (JB):
33.36
Prob(Q):                              0.71   Prob(JB):
0.00
Heteroskedasticity (H):               4.30   Skew:
-0.38
Prob(H) (two-sided):                  0.00   Kurtosis:
5.11
===================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).

Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1231.465, Time=0.54 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1128.108, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1194.595, Time=0.17 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.39 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1130.108, Time=0.02 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1255.026, Time=0.28 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1244.087, Time=0.25 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-1227.897, Time=0.88 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1234.600, Time=0.29 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-1244.515, Time=0.70 sec
 ARIMA(1,0,2)(0,0,1)[12] intercept   : AIC=-1222.946, Time=0.45 sec
 ARIMA(2,0,1)(0,0,1)[12] intercept   : AIC=-1253.637, Time=0.30 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-1233.106, Time=0.48 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1250.943, Time=0.24 sec
 ARIMA(1,0,1)(0,0,1)[12] intercept   : AIC=-1243.025, Time=0.30 sec
 ARIMA(1,0,3)(0,0,1)[12] intercept   : AIC=-1262.693, Time=0.48 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1257.503, Time=0.54 sec
 ARIMA(1,0,3)(1,0,1)[12] intercept   : AIC=-1264.552, Time=0.67 sec
 ARIMA(1,0,3)(1,0,0)[12] intercept   : AIC=-1266.225, Time=0.52 sec
 ARIMA(1,0,3)(2,0,0)[12] intercept   : AIC=-1271.653, Time=1.27 sec
 ARIMA(1,0,3)(2,0,1)[12] intercept   : AIC=-1268.744, Time=1.42 sec
 ARIMA(0,0,3)(2,0,0)[12] intercept   : AIC=-1280.559, Time=0.89 sec
```

```
 ARIMA(0,0,3)(1,0,0)[12] intercept   : AIC=-1278.059, Time=0.52 sec
 ARIMA(0,0,3)(2,0,1)[12] intercept   : AIC=-1275.152, Time=0.91 sec
 ARIMA(0,0,3)(1,0,1)[12] intercept   : AIC=-1283.417, Time=0.66 sec
 ARIMA(0,0,3)(0,0,1)[12] intercept   : AIC=-1280.626, Time=0.31 sec
 ARIMA(0,0,3)(1,0,2)[12] intercept   : AIC=-1277.138, Time=0.54 sec
 ARIMA(0,0,3)(0,0,0)[12] intercept   : AIC=-1263.635, Time=0.32 sec
 ARIMA(0,0,3)(0,0,2)[12] intercept   : AIC=-1297.068, Time=1.19 sec
 ARIMA(0,0,2)(0,0,2)[12] intercept   : AIC=-1273.969, Time=1.05 sec
 ARIMA(1,0,3)(0,0,2)[12] intercept   : AIC=-1284.070, Time=1.79 sec
 ARIMA(0,0,4)(0,0,2)[12] intercept   : AIC=-1291.672, Time=1.38 sec
 ARIMA(1,0,2)(0,0,2)[12] intercept   : AIC=-1201.418, Time=0.68 sec
 ARIMA(1,0,4)(0,0,2)[12] intercept   : AIC=-1252.014, Time=0.81 sec
 ARIMA(0,0,3)(0,0,2)[12]             : AIC=-1301.067, Time=0.89 sec
 ARIMA(0,0,3)(0,0,1)[12]             : AIC=-1282.627, Time=0.27 sec
 ARIMA(0,0,3)(1,0,2)[12]             : AIC=-1294.587, Time=1.17 sec
 ARIMA(0,0,3)(1,0,1)[12]             : AIC=-1286.180, Time=0.51 sec
 ARIMA(0,0,2)(0,0,2)[12]             : AIC=-1276.076, Time=0.84 sec
 ARIMA(1,0,3)(0,0,2)[12]             : AIC=-1294.180, Time=0.99 sec
 ARIMA(0,0,4)(0,0,2)[12]             : AIC=-1299.146, Time=1.20 sec
 ARIMA(1,0,2)(0,0,2)[12]             : AIC=-1250.085, Time=1.07 sec
 ARIMA(1,0,4)(0,0,2)[12]             : AIC=inf, Time=1.41 sec

Best model:  ARIMA(0,0,3)(0,0,2)[12]
Total fit time: 29.673 seconds
Summary for Zip Code 11222:
                                     SARIMAX Results
==========================================================================
==================
Dep. Variable:                                        y   No. Observations:
158
Model:             SARIMAX(0, 0, 3)x(0, 0, [1, 2], 12)   Log Likelihood
656.533
Date:                                 Fri, 01 Sep 2023   AIC
-1301.067
Time:                                         00:34:56   BIC
-1282.691
Sample:                                     03-01-2005   HQIC
-1293.604
                                          - 04-01-2018
Covariance Type:                                  opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
ma.L1          0.8911      0.050     17.652      0.000       0.792       0.9
90
ma.L2         -0.2174      0.079     -2.741      0.006      -0.373      -0.0
62
ma.L3         -0.6598      0.054    -12.309      0.000      -0.765      -0.5
55
ma.S.L12      -0.6132      0.111     -5.501      0.000      -0.832      -0.3
95
ma.S.L24      -0.2189      0.110     -1.988      0.047      -0.435      -0.0
03
sigma2      1.285e-05   1.45e-06      8.885      0.000       1e-05    1.57e-
```

```
05
================================================================================
=======
Ljung-Box (L1) (Q):                   3.51   Jarque-Bera (JB):
23.97
Prob(Q):                              0.06   Prob(JB):
0.00
Heteroskedasticity (H):               4.56   Skew:
0.70
Prob(H) (two-sided):                  0.00   Kurtosis:
4.30
================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).

Performing stepwise search to minimize aic
```

```
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1277.258, Time=0.59 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1203.702, Time=0.05 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1223.715, Time=0.21 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1251.875, Time=0.43 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1205.491, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1277.380, Time=0.95 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1279.175, Time=0.37 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1277.323, Time=0.67 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1237.125, Time=0.28 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1271.182, Time=0.27 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1269.039, Time=0.35 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1274.829, Time=0.29 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-1251.991, Time=0.24 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=inf, Time=0.31 sec
 ARIMA(3,0,1)(0,0,0)[12] intercept   : AIC=-1269.536, Time=0.30 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1272.866, Time=0.27 sec
 ARIMA(2,0,2)(0,0,0)[12]             : AIC=-1279.733, Time=0.14 sec
 ARIMA(2,0,2)(1,0,0)[12]             : AIC=-1277.713, Time=0.19 sec
 ARIMA(2,0,2)(0,0,1)[12]             : AIC=-1279.502, Time=0.66 sec
 ARIMA(2,0,2)(1,0,1)[12]             : AIC=-1275.240, Time=0.18 sec
 ARIMA(1,0,2)(0,0,0)[12]             : AIC=-1268.122, Time=0.20 sec
 ARIMA(2,0,1)(0,0,0)[12]             : AIC=-1273.066, Time=0.08 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-1273.476, Time=0.34 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1272.585, Time=0.13 sec
 ARIMA(1,0,1)(0,0,0)[12]             : AIC=-1253.917, Time=0.17 sec
 ARIMA(1,0,3)(0,0,0)[12]             : AIC=inf, Time=0.14 sec
 ARIMA(3,0,1)(0,0,0)[12]             : AIC=-1271.190, Time=0.21 sec
 ARIMA(3,0,3)(0,0,0)[12]             : AIC=-1271.645, Time=0.19 sec

Best model:  ARIMA(2,0,2)(0,0,0)[12]
Total fit time: 8.238 seconds
Summary for Zip Code 15201:
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                        y   No. Observations:                  1
58
Model:               SARIMAX(2, 0, 2)   Log Likelihood                 644.8
67
Date:                Fri, 01 Sep 2023   AIC                           -1279.7
33
Time:                        00:35:05   BIC                           -1264.4
20
Sample:                    03-01-2005   HQIC                          -1273.5
14
                         - 04-01-2018
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
ar.L1          0.8666      0.101      8.547      0.000       0.668       1.0
65
ar.L2         -0.4078      0.100     -4.097      0.000      -0.603      -0.2
13
```

```
ma.L1         -0.3319      0.106     -3.116      0.002      -0.541      -0.1
23
ma.L2         -0.4236      0.101     -4.175      0.000      -0.623      -0.2
25
sigma2      1.655e-05   1.74e-06      9.506      0.000    1.31e-05       2e-
05
================================================================================
=======
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
4.35
Prob(Q):                              0.98   Prob(JB):
0.11
Heteroskedasticity (H):               3.79   Skew:
0.32
Prob(H) (two-sided):                  0.00   Kurtosis:
3.50
================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).

Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1377.512, Time=0.46 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1309.933, Time=0.06 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1322.291, Time=0.13 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1368.182, Time=0.26 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1311.903, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1399.110, Time=0.39 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1409.732, Time=0.33 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1383.004, Time=0.27 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1371.324, Time=0.29 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1397.531, Time=0.40 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1396.316, Time=0.42 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1426.367, Time=0.50 sec
 ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-1421.228, Time=0.45 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1421.426, Time=0.71 sec
 ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-1423.383, Time=0.89 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1413.846, Time=0.47 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1422.369, Time=0.34 sec
 ARIMA(2,0,4)(0,0,0)[12] intercept   : AIC=-1421.801, Time=0.66 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-1423.410, Time=0.50 sec
 ARIMA(3,0,4)(0,0,0)[12] intercept   : AIC=-1418.904, Time=0.46 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1428.240, Time=0.35 sec
 ARIMA(2,0,3)(1,0,0)[12]             : AIC=-1426.822, Time=0.63 sec
 ARIMA(2,0,3)(0,0,1)[12]             : AIC=-1423.416, Time=0.26 sec
 ARIMA(2,0,3)(1,0,1)[12]             : AIC=-1425.821, Time=0.90 sec
 ARIMA(1,0,3)(0,0,0)[12]             : AIC=-1418.236, Time=0.27 sec
 ARIMA(2,0,2)(0,0,0)[12]             : AIC=-1389.958, Time=0.16 sec
 ARIMA(3,0,3)(0,0,0)[12]             : AIC=-1430.322, Time=0.49 sec
 ARIMA(3,0,3)(1,0,0)[12]             : AIC=-1428.126, Time=0.64 sec
 ARIMA(3,0,3)(0,0,1)[12]             : AIC=-1420.785, Time=0.29 sec
 ARIMA(3,0,3)(1,0,1)[12]             : AIC=-1425.764, Time=0.66 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-1390.203, Time=0.06 sec
 ARIMA(4,0,3)(0,0,0)[12]             : AIC=-1412.109, Time=0.27 sec
 ARIMA(3,0,4)(0,0,0)[12]             : AIC=-1407.779, Time=0.15 sec
```

```
 ARIMA(2,0,4)(0,0,0)[12]             : AIC=-1425.922, Time=0.35 sec
 ARIMA(4,0,2)(0,0,0)[12]             : AIC=-1407.389, Time=0.15 sec
 ARIMA(4,0,4)(0,0,0)[12]             : AIC=-1412.332, Time=0.30 sec

Best model:  ARIMA(3,0,3)(0,0,0)[12]
Total fit time: 13.955 seconds
Summary for Zip Code 94043:
                                SARIMAX Results
==========================================================================
==
Dep. Variable:                        y   No. Observations:                  1
58
Model:               SARIMAX(3, 0, 3)   Log Likelihood                722.1
61
Date:                Fri, 01 Sep 2023   AIC                          -1430.3
22
Time:                        00:35:19   BIC                          -1408.8
84
Sample:                    03-01-2005   HQIC                         -1421.6
16
                         - 04-01-2018
Covariance Type:                 opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
ar.L1          0.3526      0.091      3.855      0.000       0.173       0.5
32
ar.L2         -0.0106      0.086     -0.123      0.902      -0.180       0.1
59
ar.L3          0.3251      0.098      3.306      0.001       0.132       0.5
18
ma.L1          0.6406      0.064     10.083      0.000       0.516       0.7
65
ma.L2         -0.6169      0.062     -9.991      0.000      -0.738      -0.4
96
ma.L3         -0.8657      0.051    -17.077      0.000      -0.965      -0.7
66
sigma2      6.073e-06   5.92e-07     10.259      0.000    4.91e-06    7.23e-
06
==========================================================================
=======
Ljung-Box (L1) (Q):                   0.16   Jarque-Bera (JB):
21.53
Prob(Q):                              0.69   Prob(JB):
0.00
Heteroskedasticity (H):               7.48   Skew:
-0.30
Prob(H) (two-sided):                  0.00   Kurtosis:
4.71
==========================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
```

```
ex-step).

Performing stepwise search to minimize aic
```

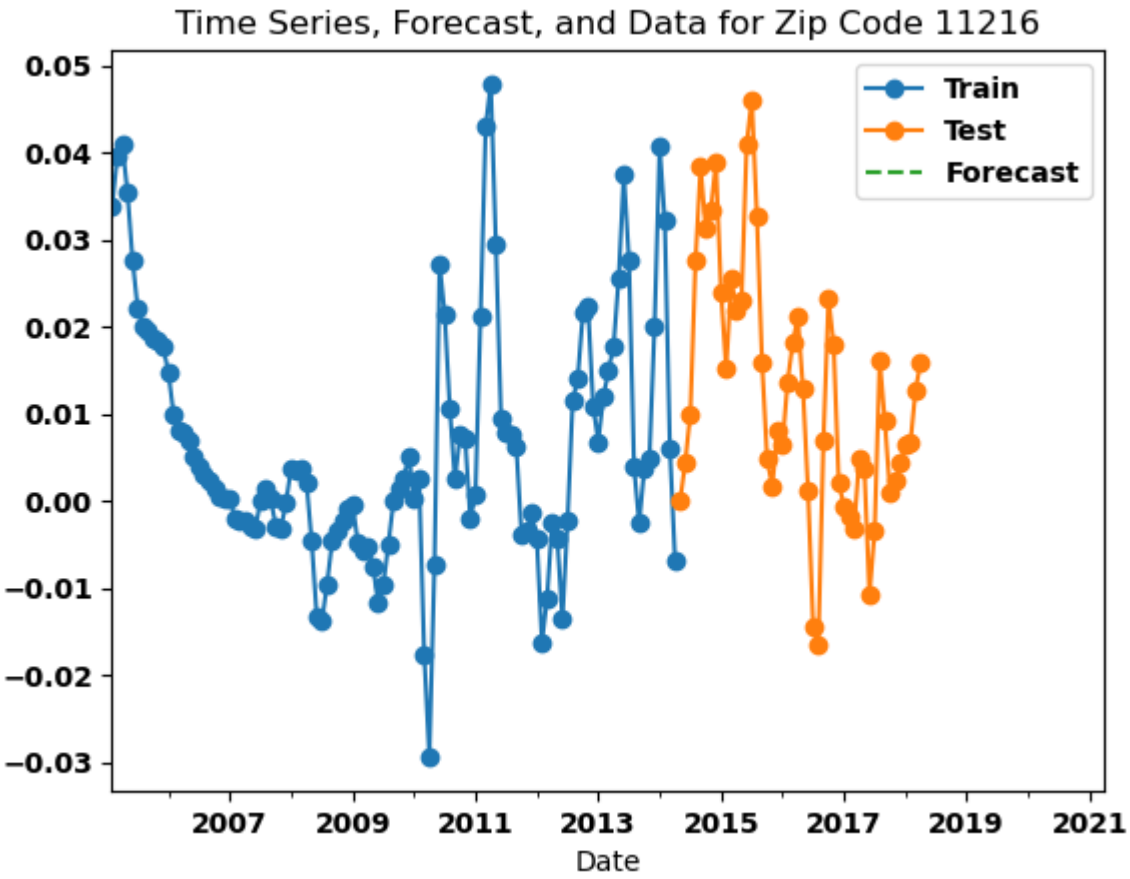
```
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1394.761, Time=0.68 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1294.426, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1325.440, Time=0.16 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1359.938, Time=0.42 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1296.425, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1385.821, Time=0.51 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1383.008, Time=0.49 sec
 ARIMA(2,0,2)(2,0,1)[12] intercept   : AIC=-1382.589, Time=1.79 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-1388.468, Time=1.38 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1396.087, Time=0.34 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1356.165, Time=0.23 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1380.651, Time=0.28 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1377.369, Time=0.29 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1416.075, Time=0.52 sec
 ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-1412.250, Time=0.32 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1391.027, Time=0.22 sec
 ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-1387.383, Time=0.42 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1409.292, Time=0.11 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1391.863, Time=0.47 sec
 ARIMA(2,0,4)(0,0,0)[12] intercept   : AIC=-1406.863, Time=0.46 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-1411.258, Time=0.33 sec
 ARIMA(3,0,4)(0,0,0)[12] intercept   : AIC=-1394.669, Time=0.66 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1412.966, Time=0.13 sec

Best model:  ARIMA(2,0,3)(0,0,0)[12] intercept
Total fit time: 10.279 seconds
Summary for Zip Code 94301:
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                      y   No. Observations:                 1
58
Model:               SARIMAX(2, 0, 3)   Log Likelihood                715.0
38
Date:                Fri, 01 Sep 2023   AIC                           -1416.0
75
Time:                        00:35:29   BIC                           -1394.6
37
Sample:                    03-01-2005   HQIC                          -1407.3
69
                         - 04-01-2018
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
intercept  -5.216e-06      0.000     -0.039      0.969      -0.000       0.0
00
ar.L1          0.3178      0.150      2.117      0.034       0.024       0.6
12
ar.L2          0.1305      0.102      1.283      0.200      -0.069       0.3
30
ma.L1          0.7194      0.124      5.822      0.000       0.477       0.9
62
ma.L2         -0.4331      0.186     -2.330      0.020      -0.797      -0.0
```

```
69
ma.L3         -0.7402      0.103     -7.182      0.000      -0.942      -0.5
38
sigma2       6.72e-06   6.27e-07     10.719      0.000    5.49e-06    7.95e-
06
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.44   Jarque-Bera (JB):
104.47
Prob(Q):                              0.51   Prob(JB):
0.00
Heteroskedasticity (H):               4.19   Skew:
0.82
Prob(H) (two-sided):                  0.00   Kurtosis:
6.63
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

In [55]:
```
#This code split the data into train and test split and visualize the focasti

# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Calculate the split index for training and testing
    train_size = 0.70  # Leaving approximately 3 years for test size
    split_idx = round(len(ts) * train_size)

    # Split the time series data into train and test sets
    train = ts.iloc[:split_idx]
    test = ts.iloc[split_idx:]

    # Fit the ARIMA model with fixed parameters to the training data
    model = pm.auto_arima(ts, trace=True, error_action='ignore', suppress_war
    model.fit(train)

    # Perform forecasting for the desired period (April 2018 to 2021)
    forecast_steps = 36  # Number of forecast steps (months from April 2018 t
    forecast = model.predict(n_periods=forecast_steps)

    # Print model summary
    print(f"Summary for Zip Code {zipcode}:")
    print(model.summary())
    print()

    # Visualize the split, time series data, and forecast
    fig, ax = plt.subplots()
    kws = dict(ax=ax, marker='o')
    train.plot(**kws, label='Train')
    test.plot(**kws, label='Test')
    forecast_index = pd.date_range(start=test.index[-1], periods=forecast_ste
    forecast_series = pd.Series(forecast, index=forecast_index)
    forecast_series.plot(ax=ax, label='Forecast', linestyle='dashed')
    ax.legend(bbox_to_anchor=[1, 1])
    plt.title(f"Time Series, Forecast, and Data for Zip Code {zipcode}")
    plt.show()
```

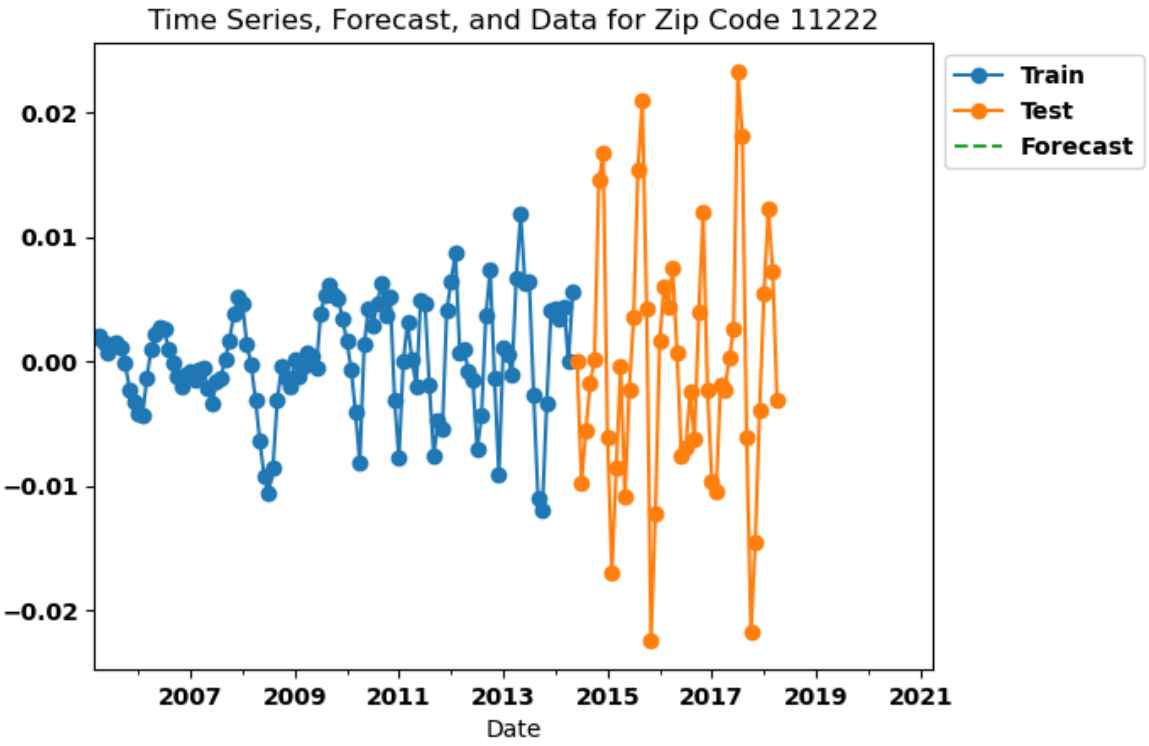
```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1157.995, Time=0.69 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-889.320, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1075.783, Time=0.29 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1048.416, Time=0.30 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-846.771, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1168.942, Time=0.57 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1123.269, Time=0.35 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-1170.572, Time=1.13 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=inf, Time=1.27 sec
 ARIMA(1,0,2)(0,0,2)[12] intercept   : AIC=-1159.154, Time=1.04 sec
 ARIMA(2,0,1)(0,0,2)[12] intercept   : AIC=inf, Time=0.98 sec
 ARIMA(3,0,2)(0,0,2)[12] intercept   : AIC=-1181.565, Time=1.18 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-1181.422, Time=0.63 sec
 ARIMA(3,0,2)(1,0,2)[12] intercept   : AIC=inf, Time=1.34 sec
 ARIMA(3,0,2)(1,0,1)[12] intercept   : AIC=-1170.854, Time=0.82 sec
 ARIMA(3,0,1)(0,0,2)[12] intercept   : AIC=inf, Time=1.17 sec
 ARIMA(4,0,2)(0,0,2)[12] intercept   : AIC=-1184.421, Time=1.74 sec
 ARIMA(4,0,2)(0,0,1)[12] intercept   : AIC=-1191.108, Time=0.86 sec
 ARIMA(4,0,2)(0,0,0)[12] intercept   : AIC=-1129.372, Time=0.45 sec
 ARIMA(4,0,2)(1,0,1)[12] intercept   : AIC=-1168.681, Time=0.94 sec
 ARIMA(4,0,2)(1,0,0)[12] intercept   : AIC=-1147.064, Time=0.80 sec
 ARIMA(4,0,2)(1,0,2)[12] intercept   : AIC=-1183.013, Time=1.61 sec
 ARIMA(4,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.66 sec
 ARIMA(5,0,2)(0,0,1)[12] intercept   : AIC=-1179.743, Time=0.82 sec
 ARIMA(4,0,3)(0,0,1)[12] intercept   : AIC=-1183.974, Time=0.70 sec
 ARIMA(3,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.55 sec
 ARIMA(3,0,3)(0,0,1)[12] intercept   : AIC=-1190.467, Time=0.64 sec
 ARIMA(5,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.70 sec
 ARIMA(5,0,3)(0,0,1)[12] intercept   : AIC=-1188.879, Time=0.74 sec
 ARIMA(4,0,2)(0,0,1)[12]             : AIC=-1187.420, Time=0.69 sec

Best model:  ARIMA(4,0,2)(0,0,1)[12] intercept
Total fit time: 23.743 seconds
Summary for Zip Code 11216:
                                     SARIMAX Results
==========================================================================
================
Dep. Variable:                                     y   No. Observations:
111
Model:             SARIMAX(4, 0, 2)x(0, 0, [1], 12)   Log Likelihood
425.615
Date:                              Fri, 01 Sep 2023   AIC
-833.230
Time:                                      00:35:53   BIC
-808.844
Sample:                                  02-01-2005   HQIC
-823.337
                                       - 04-01-2014
Covariance Type:                               opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
intercept      0.0005      0.001      0.467      0.640      -0.001       0.0
```

```
02
ar.L1          0.2912      0.173      1.685      0.092      -0.048       0.6
30
ar.L2          0.3407      0.174      1.960      0.050    5.94e-05       0.6
81
ar.L3          0.1520      0.158      0.964      0.335      -0.157       0.4
61
ar.L4          0.1409      0.145      0.973      0.331      -0.143       0.4
25
ma.L1          1.4347      0.134     10.709      0.000       1.172       1.6
97
ma.L2          0.7160      0.115      6.236      0.000       0.491       0.9
41
ma.S.L12      -0.7635      0.128     -5.964      0.000      -1.014      -0.5
13
sigma2      2.346e-05   3.18e-06      7.370      0.000    1.72e-05    2.97e-
05
===================================================================================
=======
Ljung-Box (L1) (Q):                   0.08   Jarque-Bera (JB):
67.11
Prob(Q):                              0.78   Prob(JB):
0.00
Heteroskedasticity (H):               8.70   Skew:
-0.82
Prob(H) (two-sided):                  0.00   Kurtosis:
6.44
===================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```


Time Series, Forecast, and Data for Zip Code 11216
Train
Test
Forecast
0.05
0.04
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
2007
2009
2011
2013
2015
2017
2019
2021
Date

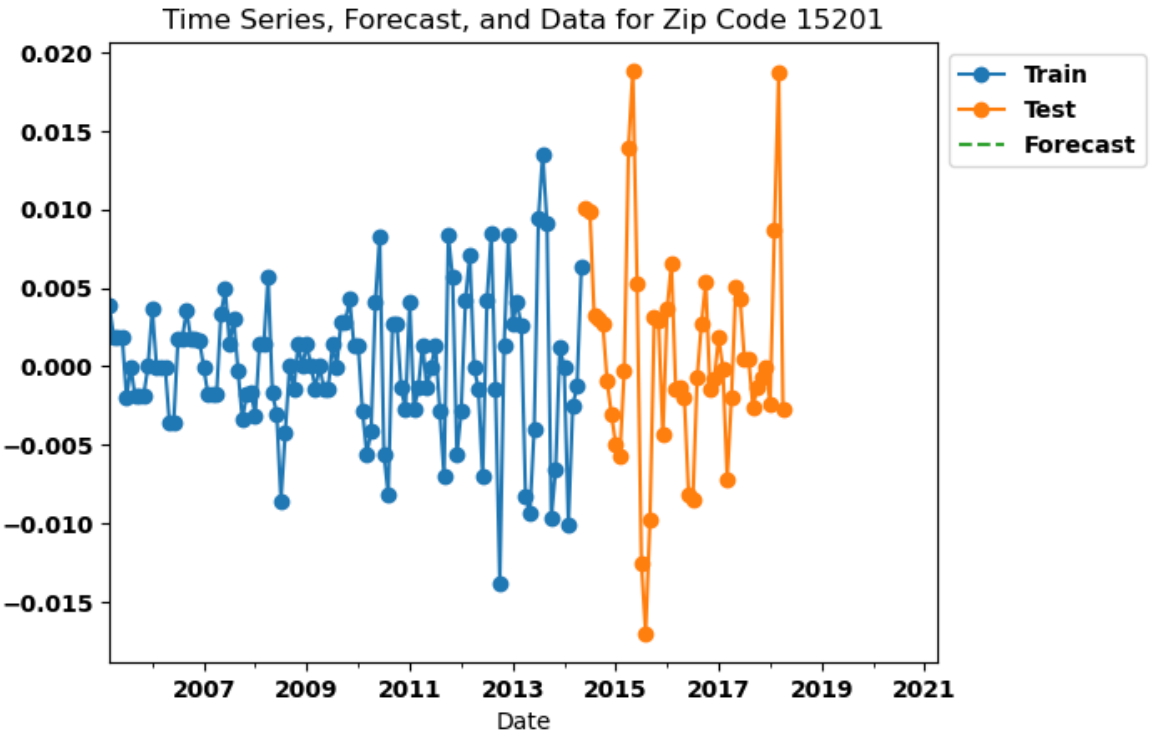
```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1231.465, Time=0.53 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1128.108, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1194.595, Time=0.17 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.35 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1130.108, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1255.026, Time=0.28 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1244.087, Time=0.22 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-1227.897, Time=0.86 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1234.600, Time=0.27 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-1244.515, Time=0.65 sec
 ARIMA(1,0,2)(0,0,1)[12] intercept   : AIC=-1222.946, Time=0.45 sec
 ARIMA(2,0,1)(0,0,1)[12] intercept   : AIC=-1253.637, Time=0.29 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-1233.106, Time=0.46 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1250.943, Time=0.21 sec
 ARIMA(1,0,1)(0,0,1)[12] intercept   : AIC=-1243.025, Time=0.29 sec
 ARIMA(1,0,3)(0,0,1)[12] intercept   : AIC=-1262.693, Time=0.44 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1257.503, Time=0.50 sec
 ARIMA(1,0,3)(1,0,1)[12] intercept   : AIC=-1264.552, Time=0.66 sec
 ARIMA(1,0,3)(1,0,0)[12] intercept   : AIC=-1266.225, Time=0.52 sec
 ARIMA(1,0,3)(2,0,0)[12] intercept   : AIC=-1271.653, Time=1.06 sec
 ARIMA(1,0,3)(2,0,1)[12] intercept   : AIC=-1268.744, Time=1.25 sec
 ARIMA(0,0,3)(2,0,0)[12] intercept   : AIC=-1280.559, Time=0.88 sec
 ARIMA(0,0,3)(1,0,0)[12] intercept   : AIC=-1278.059, Time=0.45 sec
 ARIMA(0,0,3)(2,0,1)[12] intercept   : AIC=-1275.152, Time=0.87 sec
 ARIMA(0,0,3)(1,0,1)[12] intercept   : AIC=-1283.417, Time=0.57 sec
 ARIMA(0,0,3)(0,0,1)[12] intercept   : AIC=-1280.626, Time=0.27 sec
 ARIMA(0,0,3)(1,0,2)[12] intercept   : AIC=-1277.138, Time=0.46 sec
 ARIMA(0,0,3)(0,0,0)[12] intercept   : AIC=-1263.635, Time=0.29 sec
 ARIMA(0,0,3)(0,0,2)[12] intercept   : AIC=-1297.068, Time=1.00 sec
 ARIMA(0,0,2)(0,0,2)[12] intercept   : AIC=-1273.969, Time=0.86 sec
 ARIMA(1,0,3)(0,0,2)[12] intercept   : AIC=-1284.070, Time=1.39 sec
 ARIMA(0,0,4)(0,0,2)[12] intercept   : AIC=-1291.672, Time=1.14 sec
 ARIMA(1,0,2)(0,0,2)[12] intercept   : AIC=-1201.418, Time=0.58 sec
 ARIMA(1,0,4)(0,0,2)[12] intercept   : AIC=-1252.014, Time=0.70 sec
 ARIMA(0,0,3)(0,0,2)[12]             : AIC=-1301.067, Time=0.80 sec
 ARIMA(0,0,3)(0,0,1)[12]             : AIC=-1282.627, Time=0.21 sec
 ARIMA(0,0,3)(1,0,2)[12]             : AIC=-1294.587, Time=1.13 sec
 ARIMA(0,0,3)(1,0,1)[12]             : AIC=-1286.180, Time=0.46 sec
 ARIMA(0,0,2)(0,0,2)[12]             : AIC=-1276.076, Time=0.76 sec
 ARIMA(1,0,3)(0,0,2)[12]             : AIC=-1294.180, Time=0.97 sec
 ARIMA(0,0,4)(0,0,2)[12]             : AIC=-1299.146, Time=1.03 sec
 ARIMA(1,0,2)(0,0,2)[12]             : AIC=-1250.085, Time=0.78 sec
 ARIMA(1,0,4)(0,0,2)[12]             : AIC=inf, Time=1.12 sec

Best model:  ARIMA(0,0,3)(0,0,2)[12]
Total fit time: 26.311 seconds
Summary for Zip Code 11222:
                                     SARIMAX Results
==========================================================================
==================
Dep. Variable:                                        y   No. Observations:
111
Model:             SARIMAX(0, 0, 3)x(0, 0, [1, 2], 12)   Log Likelihood
489.118
Date:                                 Fri, 01 Sep 2023   AIC
-966.237
```

```
Time:                                      00:36:21   BIC
-949.980
Sample:                                  03-01-2005   HQIC
-959.642
                                       - 05-01-2014
Covariance Type:                               opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
ma.L1          1.0403      0.091     11.415      0.000       0.862       1.2
19
ma.L2          0.2991      0.118      2.529      0.011       0.067       0.5
31
ma.L3         -0.1626      0.089     -1.835      0.066      -0.336       0.0
11
ma.S.L12      -0.4669      0.117     -3.976      0.000      -0.697      -0.2
37
ma.S.L24      -0.2448      0.117     -2.093      0.036      -0.474      -0.0
16
sigma2      8.026e-06   9.83e-07      8.169      0.000     6.1e-06    9.95e-
06
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
6.04
Prob(Q):                              0.96   Prob(JB):
0.05
Heteroskedasticity (H):              12.28   Skew:
0.24
Prob(H) (two-sided):                  0.00   Kurtosis:
4.03
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```


Time Series, Forecast, and Data for Zip Code 11222
Train
Test
Forecast
0.02
0.01
0.00
−0.01
−0.02
2007
2009
2011
2013
2015
2017
2019
2021
Date

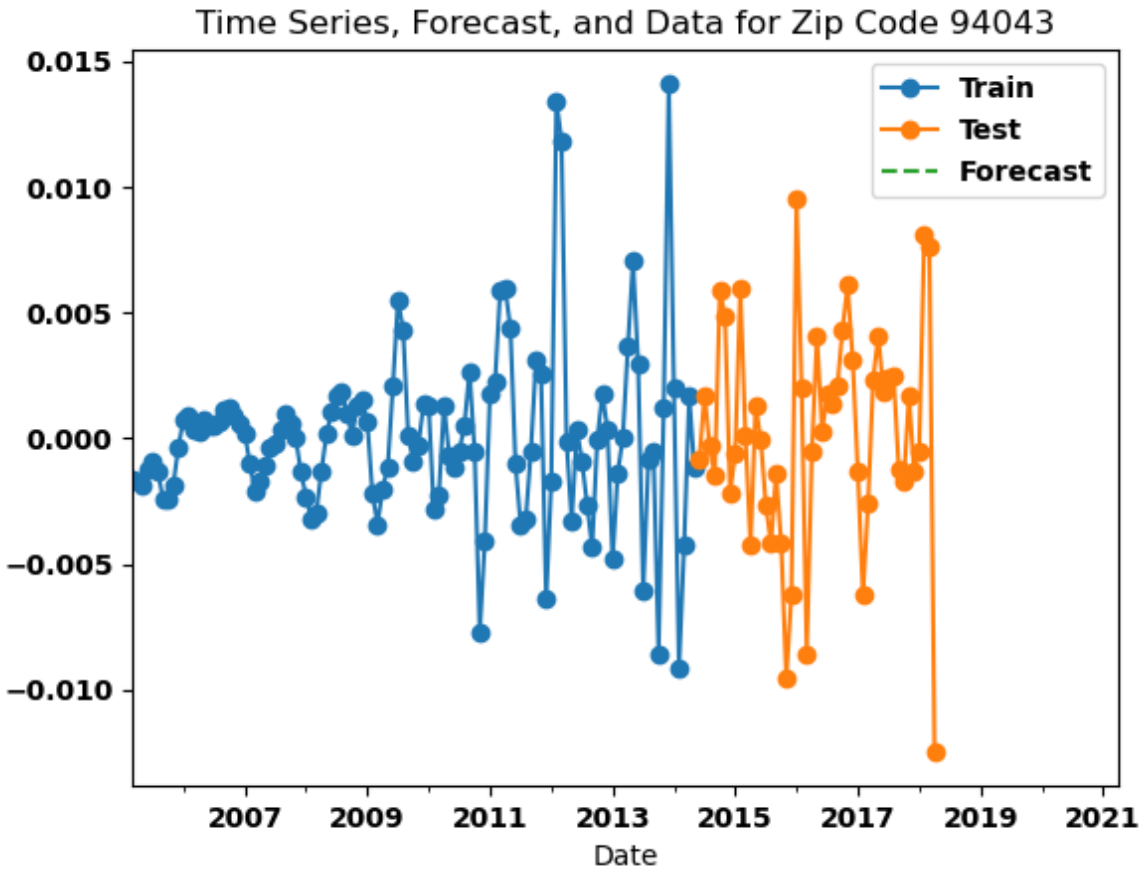
```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1277.258, Time=0.56 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1203.702, Time=0.05 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1223.715, Time=0.16 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1251.875, Time=0.36 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1205.491, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1277.380, Time=0.79 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1279.175, Time=0.30 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1277.323, Time=0.59 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1237.125, Time=0.27 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1271.182, Time=0.25 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1269.039, Time=0.36 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1274.829, Time=0.35 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-1251.991, Time=0.29 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=inf, Time=0.33 sec
 ARIMA(3,0,1)(0,0,0)[12] intercept   : AIC=-1269.536, Time=0.31 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1272.866, Time=0.25 sec
 ARIMA(2,0,2)(0,0,0)[12]             : AIC=-1279.733, Time=0.13 sec
 ARIMA(2,0,2)(1,0,0)[12]             : AIC=-1277.713, Time=0.22 sec
 ARIMA(2,0,2)(0,0,1)[12]             : AIC=-1279.502, Time=0.64 sec
 ARIMA(2,0,2)(1,0,1)[12]             : AIC=-1275.240, Time=0.20 sec
 ARIMA(1,0,2)(0,0,0)[12]             : AIC=-1268.122, Time=0.22 sec
 ARIMA(2,0,1)(0,0,0)[12]             : AIC=-1273.066, Time=0.08 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-1273.476, Time=0.40 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1272.585, Time=0.16 sec
 ARIMA(1,0,1)(0,0,0)[12]             : AIC=-1253.917, Time=0.16 sec
 ARIMA(1,0,3)(0,0,0)[12]             : AIC=inf, Time=0.15 sec
 ARIMA(3,0,1)(0,0,0)[12]             : AIC=-1271.190, Time=0.24 sec
 ARIMA(3,0,3)(0,0,0)[12]             : AIC=-1271.645, Time=0.22 sec

Best model:  ARIMA(2,0,2)(0,0,0)[12]
Total fit time: 8.099 seconds
Summary for Zip Code 15201:
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                        y   No. Observations:                 1
11
Model:               SARIMAX(2, 0, 2)   Log Likelihood                467.1
67
Date:                Fri, 01 Sep 2023   AIC                           -924.3
33
Time:                        00:36:29   BIC                           -910.7
85
Sample:                    03-01-2005   HQIC                          -918.8
37
                         - 05-01-2014
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
ar.L1          0.6564      0.140      4.703      0.000       0.383       0.9
30
ar.L2         -0.4047      0.139     -2.917      0.004      -0.677      -0.1
```

```
33
ma.L1         -0.3494      0.183     -1.910      0.056      -0.708       0.0
09
ma.L2         -0.2820      0.165     -1.705      0.088      -0.606       0.0
42
sigma2      1.283e-05   1.85e-06      6.941      0.000     9.2e-06    1.64e-
05
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.02   Jarque-Bera (JB):
0.44
Prob(Q):                              0.88   Prob(JB):
0.80
Heteroskedasticity (H):               5.24   Skew:
-0.15
Prob(H) (two-sided):                  0.00   Kurtosis:
3.05
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```



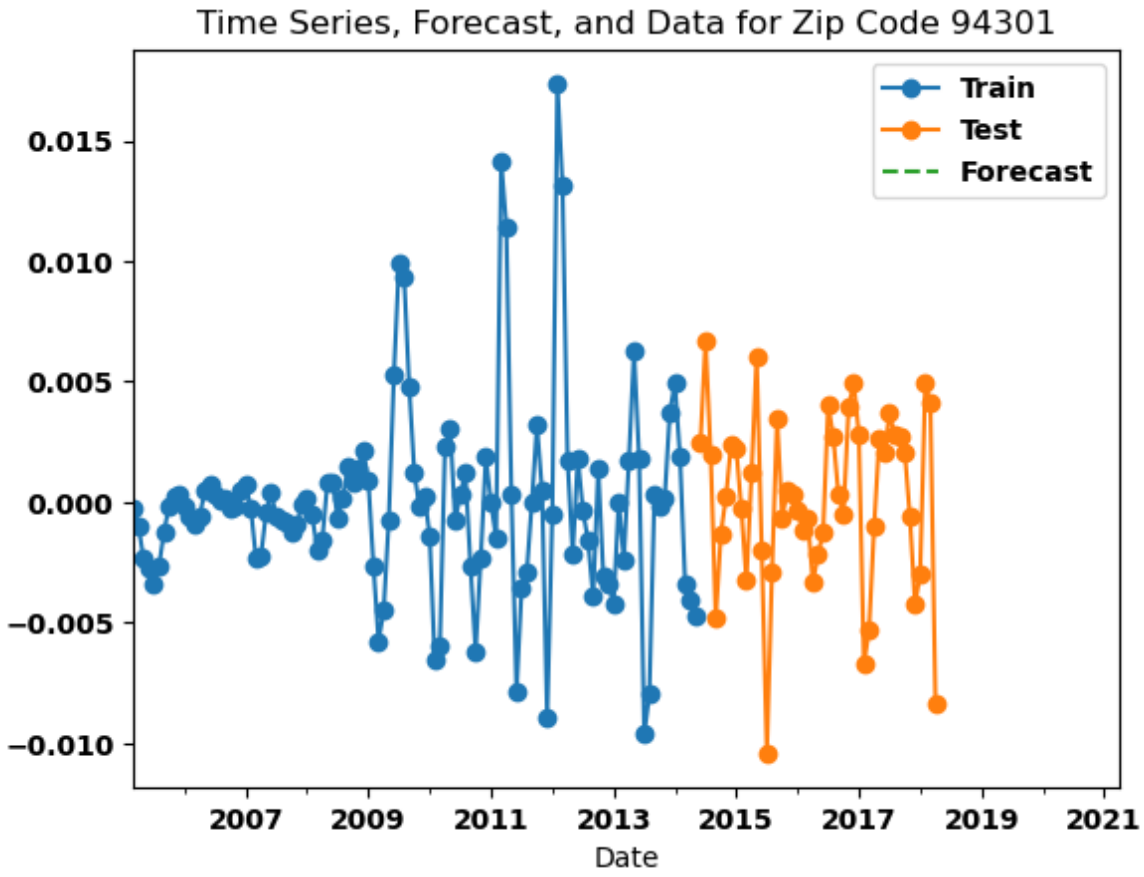
```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1377.512, Time=0.47 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1309.933, Time=0.07 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1322.291, Time=0.20 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1368.182, Time=0.27 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1311.903, Time=0.04 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1399.110, Time=0.51 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1409.732, Time=0.41 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1383.004, Time=0.25 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1371.324, Time=0.31 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1397.531, Time=0.38 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1396.316, Time=0.52 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1426.367, Time=0.58 sec
 ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-1421.228, Time=0.53 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1421.426, Time=0.76 sec
 ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-1423.383, Time=1.08 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1413.846, Time=0.58 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1422.369, Time=0.46 sec
 ARIMA(2,0,4)(0,0,0)[12] intercept   : AIC=-1421.801, Time=0.80 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-1423.410, Time=0.59 sec
 ARIMA(3,0,4)(0,0,0)[12] intercept   : AIC=-1418.904, Time=0.69 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1428.240, Time=0.38 sec
 ARIMA(2,0,3)(1,0,0)[12]             : AIC=-1426.822, Time=0.68 sec
 ARIMA(2,0,3)(0,0,1)[12]             : AIC=-1423.416, Time=0.27 sec
 ARIMA(2,0,3)(1,0,1)[12]             : AIC=-1425.821, Time=0.83 sec
 ARIMA(1,0,3)(0,0,0)[12]             : AIC=-1418.236, Time=0.26 sec
 ARIMA(2,0,2)(0,0,0)[12]             : AIC=-1389.958, Time=0.11 sec
 ARIMA(3,0,3)(0,0,0)[12]             : AIC=-1430.322, Time=0.45 sec
 ARIMA(3,0,3)(1,0,0)[12]             : AIC=-1428.126, Time=0.57 sec
 ARIMA(3,0,3)(0,0,1)[12]             : AIC=-1420.785, Time=0.26 sec
 ARIMA(3,0,3)(1,0,1)[12]             : AIC=-1425.764, Time=0.66 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-1390.203, Time=0.06 sec
 ARIMA(4,0,3)(0,0,0)[12]             : AIC=-1412.109, Time=0.28 sec
 ARIMA(3,0,4)(0,0,0)[12]             : AIC=-1407.779, Time=0.17 sec
 ARIMA(2,0,4)(0,0,0)[12]             : AIC=-1425.922, Time=0.34 sec
 ARIMA(4,0,2)(0,0,0)[12]             : AIC=-1407.389, Time=0.13 sec
 ARIMA(4,0,4)(0,0,0)[12]             : AIC=-1412.332, Time=0.26 sec

Best model:  ARIMA(3,0,3)(0,0,0)[12]
Total fit time: 15.243 seconds
Summary for Zip Code 94043:
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                        y   No. Observations:                  1
11
Model:               SARIMAX(3, 0, 3)   Log Likelihood                 523.5
89
Date:                Fri, 01 Sep 2023   AIC                          -1033.1
79
Time:                        00:36:45   BIC                          -1014.2
12
Sample:                    03-01-2005   HQIC                         -1025.4
85
                         - 05-01-2014
Covariance Type:                  opg
==============================================================================
```

```
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
ar.L1          0.4804      0.098      4.885      0.000       0.288       0.6
73
ar.L2         -0.1044      0.101     -1.032      0.302      -0.303       0.0
94
ar.L3          0.3358      0.104      3.217      0.001       0.131       0.5
40
ma.L1          0.5551      0.094      5.931      0.000       0.372       0.7
39
ma.L2         -0.6310      0.067     -9.427      0.000      -0.762      -0.5
00
ma.L3         -0.8273      0.071    -11.722      0.000      -0.966      -0.6
89
sigma2       4.49e-06   5.57e-07      8.063      0.000     3.4e-06    5.58e-
06
===================================================================================
=======
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
17.53
Prob(Q):                              0.95   Prob(JB):
0.00
Heteroskedasticity (H):              10.63   Skew:
0.85
Prob(H) (two-sided):                  0.00   Kurtosis:
3.95
===================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```


Time Series, Forecast, and Data for Zip Code 94043
Train
Test
Forecast
0.015
0.010
0.005
0.000
−0.005
−0.010
2007
2009
2011
2013
2015
2017
2019
2021
Date

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-1394.761, Time=0.68 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-1294.426, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-1325.440, Time=0.15 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-1359.938, Time=0.36 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-1296.425, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-1385.821, Time=0.46 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-1383.008, Time=0.45 sec
 ARIMA(2,0,2)(2,0,1)[12] intercept   : AIC=-1382.589, Time=1.62 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-1388.468, Time=1.37 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-1396.087, Time=0.40 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-1356.165, Time=0.26 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-1380.651, Time=0.29 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-1377.369, Time=0.35 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-1416.075, Time=0.56 sec
 ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-1412.250, Time=0.38 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-1391.027, Time=0.21 sec
 ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-1387.383, Time=0.48 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-1409.292, Time=0.10 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-1391.863, Time=0.50 sec
 ARIMA(2,0,4)(0,0,0)[12] intercept   : AIC=-1406.863, Time=0.53 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-1411.258, Time=0.36 sec
 ARIMA(3,0,4)(0,0,0)[12] intercept   : AIC=-1394.669, Time=0.71 sec
 ARIMA(2,0,3)(0,0,0)[12]             : AIC=-1412.966, Time=0.13 sec

Best model:  ARIMA(2,0,3)(0,0,0)[12] intercept
Total fit time: 10.440 seconds
Summary for Zip Code 94301:
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                        y   No. Observations:                  1
11
Model:               SARIMAX(2, 0, 3)   Log Likelihood                 497.8
38
Date:                Fri, 01 Sep 2023   AIC                           -981.6
77
Time:                        00:36:56   BIC                           -962.7
10
Sample:                    03-01-2005   HQIC                          -973.9
82
                         - 05-01-2014
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
intercept  -2.028e-05      0.000     -0.098      0.922      -0.000       0.0
00
ar.L1          0.3152      0.213      1.483      0.138      -0.101       0.7
32
ar.L2          0.1336      0.131      1.016      0.310      -0.124       0.3
91
ma.L1          0.6431      0.187      3.433      0.001       0.276       1.0
10
```

```
ma.L2         -0.3359      0.252     -1.332      0.183      -0.830       0.1
59
ma.L3         -0.7040      0.145     -4.861      0.000      -0.988      -0.4
20
sigma2       7.23e-06   8.16e-07      8.856      0.000    5.63e-06    8.83e-
06
===================================================================================
=======
Ljung-Box (L1) (Q):                   0.06   Jarque-Bera (JB):
228.98
Prob(Q):                              0.81   Prob(JB):
0.00
Heteroskedasticity (H):              18.94   Skew:
1.55
Prob(H) (two-sided):                  0.00   Kurtosis:
9.32
===================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

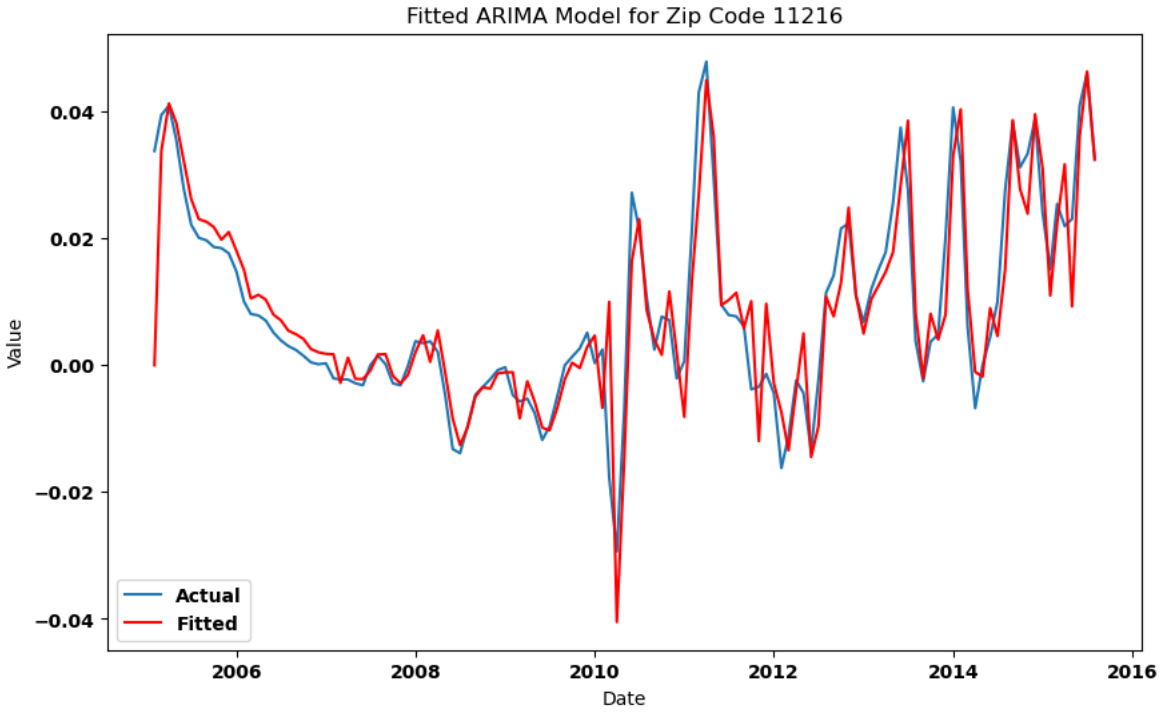


Model Training

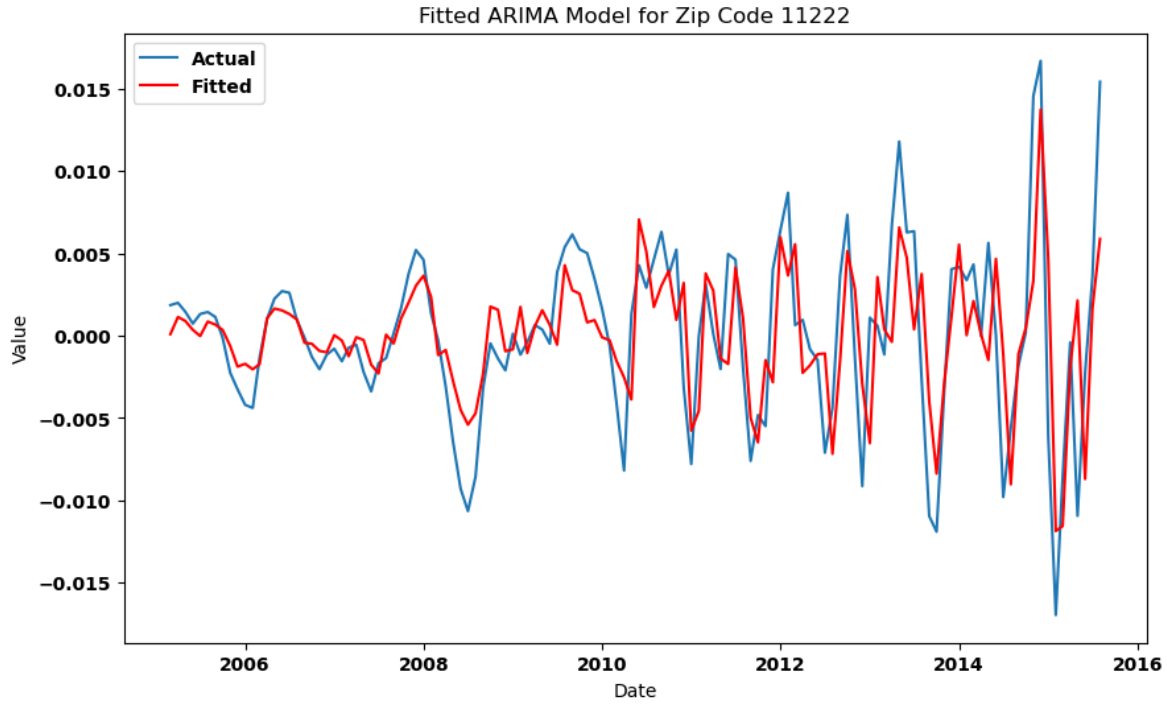
The code below trains an ARIMA model with p,d,q for each zip code using fixed parameters. It then visualizes the fitted model and actual values to give a sense of how well the model captures the training data.

In [56]:
```
# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']


# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Calculate the split index for training and testing
    train_size = 0.80  # Leaving approximately 3 years for test size
    split_idx = round(len(ts) * train_size)

    # Split the time series data into train and test sets
    train = ts.iloc[:split_idx]
    test = ts.iloc[split_idx:]

    # Fit the ARIMA model with fixed parameters to the training data
    model = pm.auto_arima(train, trace=True, error_action='ignore', suppress_
    model.fit(train)

    # Visualize the fitted model and actual values
    plt.figure(figsize=(10, 6))
    plt.plot(train.index, train.values, label='Actual')
    plt.plot(train.index, model.predict_in_sample(), label='Fitted', color='r
    plt.title(f"Fitted ARIMA Model for Zip Code {zipcode}")
    plt.xlabel("Date")
    plt.ylabel("Value")
    plt.legend()
    plt.show()
```

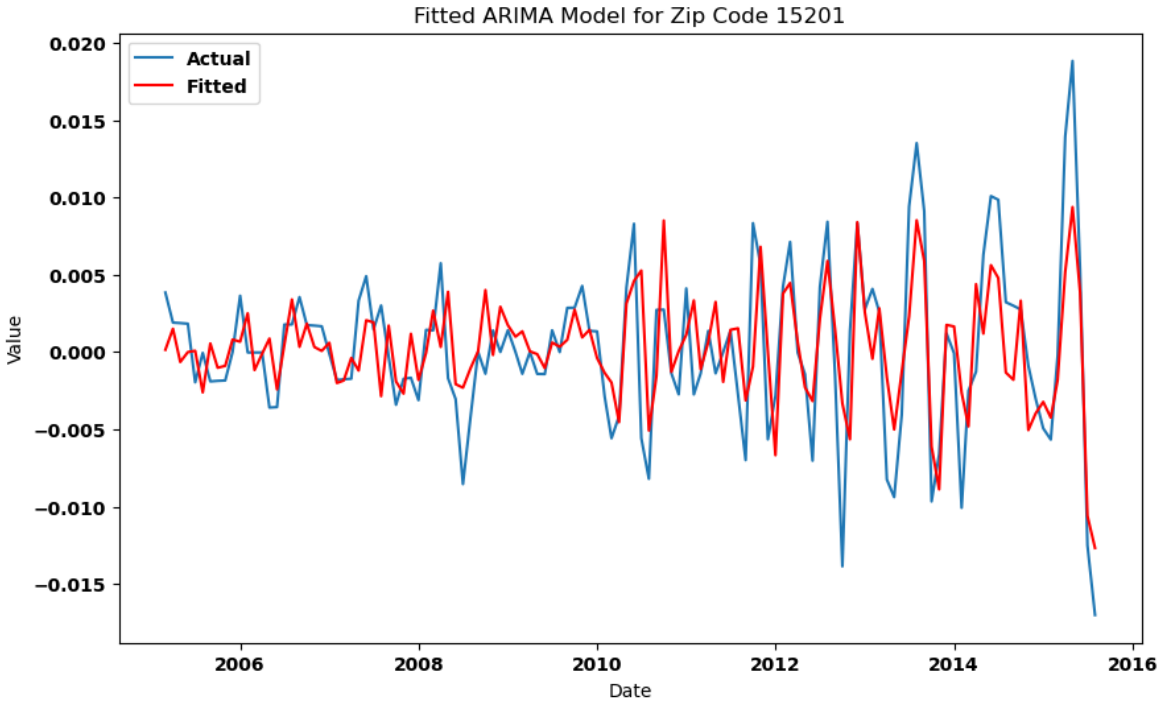
```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,0,1)[24] intercept   : AIC=-890.762, Time=0.84 sec
 ARIMA(0,1,0)(0,0,0)[24] intercept   : AIC=-829.918, Time=0.06 sec
 ARIMA(1,1,0)(1,0,0)[24] intercept   : AIC=-842.023, Time=0.32 sec
 ARIMA(0,1,1)(0,0,1)[24] intercept   : AIC=-873.559, Time=0.51 sec
 ARIMA(0,1,0)(0,0,0)[24]             : AIC=-831.918, Time=0.04 sec
 ARIMA(2,1,2)(0,0,1)[24] intercept   : AIC=-893.679, Time=1.02 sec
 ARIMA(2,1,2)(0,0,0)[24] intercept   : AIC=-893.925, Time=0.10 sec
 ARIMA(2,1,2)(1,0,0)[24] intercept   : AIC=-893.430, Time=1.19 sec
 ARIMA(1,1,2)(0,0,0)[24] intercept   : AIC=-883.906, Time=0.27 sec
 ARIMA(2,1,1)(0,0,0)[24] intercept   : AIC=-889.301, Time=0.28 sec
 ARIMA(3,1,2)(0,0,0)[24] intercept   : AIC=-886.397, Time=0.40 sec
 ARIMA(2,1,3)(0,0,0)[24] intercept   : AIC=-899.703, Time=0.40 sec
 ARIMA(2,1,3)(1,0,0)[24] intercept   : AIC=-901.625, Time=1.11 sec
 ARIMA(2,1,3)(2,0,0)[24] intercept   : AIC=-900.180, Time=4.75 sec
 ARIMA(2,1,3)(1,0,1)[24] intercept   : AIC=-898.805, Time=0.80 sec
 ARIMA(2,1,3)(0,0,1)[24] intercept   : AIC=-906.481, Time=1.06 sec
 ARIMA(2,1,3)(0,0,2)[24] intercept   : AIC=-895.704, Time=4.03 sec
 ARIMA(2,1,3)(1,0,2)[24] intercept   : AIC=-893.704, Time=2.54 sec
 ARIMA(1,1,3)(0,0,1)[24] intercept   : AIC=-912.497, Time=1.00 sec
 ARIMA(1,1,3)(0,0,0)[24] intercept   : AIC=-901.535, Time=0.29 sec
 ARIMA(1,1,3)(1,0,1)[24] intercept   : AIC=-903.675, Time=1.21 sec
 ARIMA(1,1,3)(0,0,2)[24] intercept   : AIC=-897.535, Time=1.64 sec
 ARIMA(1,1,3)(1,0,0)[24] intercept   : AIC=-899.891, Time=0.32 sec
 ARIMA(1,1,3)(1,0,2)[24] intercept   : AIC=-895.536, Time=2.03 sec
 ARIMA(0,1,3)(0,0,1)[24] intercept   : AIC=-914.097, Time=1.13 sec
 ARIMA(0,1,3)(0,0,0)[24] intercept   : AIC=-921.913, Time=0.14 sec
 ARIMA(0,1,3)(1,0,0)[24] intercept   : AIC=-923.109, Time=0.79 sec
 ARIMA(0,1,3)(2,0,0)[24] intercept   : AIC=-908.256, Time=1.01 sec
 ARIMA(0,1,3)(1,0,1)[24] intercept   : AIC=-921.097, Time=1.20 sec
 ARIMA(0,1,3)(2,0,1)[24] intercept   : AIC=-909.029, Time=3.18 sec
 ARIMA(0,1,2)(1,0,0)[24] intercept   : AIC=-871.387, Time=0.78 sec
 ARIMA(0,1,4)(1,0,0)[24] intercept   : AIC=-912.175, Time=0.33 sec
 ARIMA(1,1,2)(1,0,0)[24] intercept   : AIC=-882.840, Time=0.78 sec
 ARIMA(1,1,4)(1,0,0)[24] intercept   : AIC=-911.600, Time=0.79 sec
 ARIMA(0,1,3)(1,0,0)[24]             : AIC=-927.754, Time=0.63 sec
 ARIMA(0,1,3)(0,0,0)[24]             : AIC=-928.683, Time=0.23 sec
 ARIMA(0,1,3)(0,0,1)[24]             : AIC=-926.352, Time=0.74 sec
 ARIMA(0,1,3)(1,0,1)[24]             : AIC=-915.197, Time=0.59 sec
 ARIMA(0,1,2)(0,0,0)[24]             : AIC=-872.365, Time=0.13 sec
 ARIMA(1,1,3)(0,0,0)[24]             : AIC=-922.787, Time=0.29 sec
 ARIMA(0,1,4)(0,0,0)[24]             : AIC=-917.286, Time=0.16 sec
 ARIMA(1,1,2)(0,0,0)[24]             : AIC=-885.925, Time=0.18 sec
 ARIMA(1,1,4)(0,0,0)[24]             : AIC=inf, Time=0.36 sec

Best model:  ARIMA(0,1,3)(0,0,0)[24]
Total fit time: 39.695 seconds
```



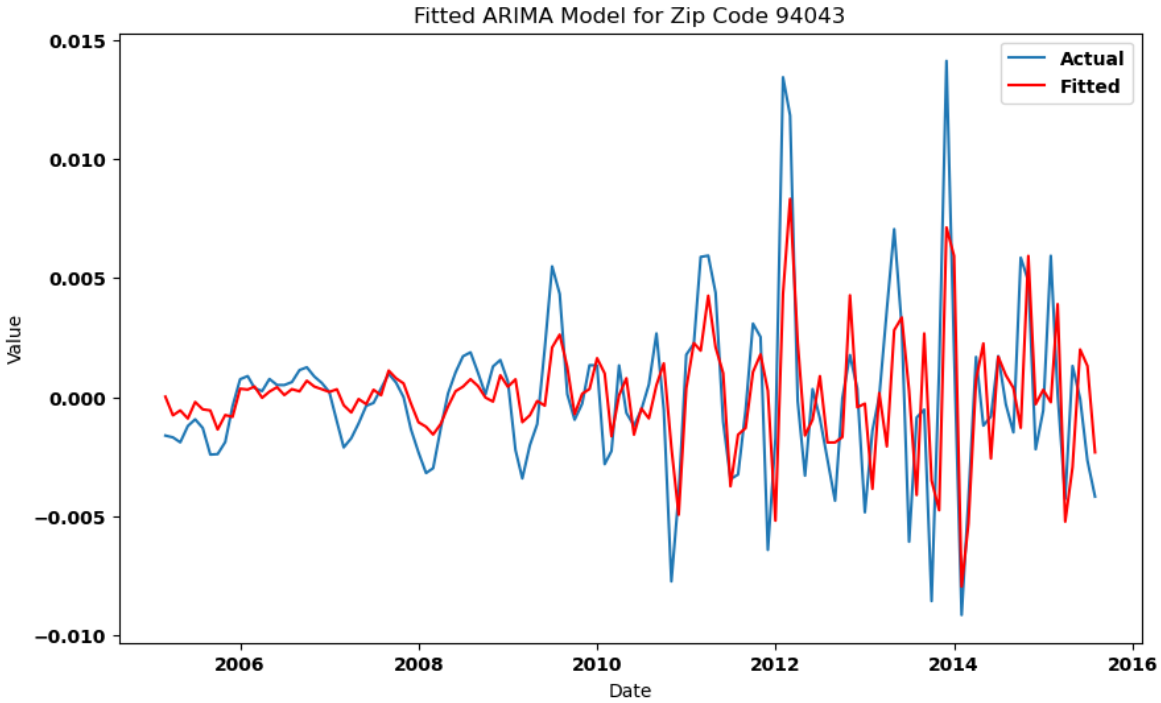

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=-1026.180, Time=1.86 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-958.893, Time=0.05 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1001.676, Time=0.41 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1017.972, Time=0.50 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-960.890, Time=0.05 sec
 ARIMA(2,0,2)(0,0,1)[24] intercept   : AIC=-1015.674, Time=1.03 sec
 ARIMA(2,0,2)(1,0,0)[24] intercept   : AIC=-1011.890, Time=0.76 sec
 ARIMA(2,0,2)(2,0,1)[24] intercept   : AIC=-1016.195, Time=5.04 sec
 ARIMA(2,0,2)(1,0,2)[24] intercept   : AIC=-1003.214, Time=4.47 sec
 ARIMA(2,0,2)(0,0,0)[24] intercept   : AIC=-1010.172, Time=0.33 sec
 ARIMA(2,0,2)(0,0,2)[24] intercept   : AIC=-1023.640, Time=3.48 sec
 ARIMA(2,0,2)(2,0,0)[24] intercept   : AIC=-1017.866, Time=4.07 sec
 ARIMA(2,0,2)(2,0,2)[24] intercept   : AIC=-1002.035, Time=2.90 sec
 ARIMA(1,0,2)(1,0,1)[24] intercept   : AIC=-1010.581, Time=0.70 sec
 ARIMA(2,0,1)(1,0,1)[24] intercept   : AIC=-1028.100, Time=1.12 sec
 ARIMA(2,0,1)(0,0,1)[24] intercept   : AIC=-1029.235, Time=1.15 sec
 ARIMA(2,0,1)(0,0,0)[24] intercept   : AIC=-1023.112, Time=0.16 sec
 ARIMA(2,0,1)(0,0,2)[24] intercept   : AIC=-1029.831, Time=2.52 sec
 ARIMA(2,0,1)(1,0,2)[24] intercept   : AIC=-1028.315, Time=3.43 sec
 ARIMA(1,0,1)(0,0,2)[24] intercept   : AIC=-1023.182, Time=2.93 sec
 ARIMA(2,0,0)(0,0,2)[24] intercept   : AIC=-1031.684, Time=2.30 sec
 ARIMA(2,0,0)(0,0,1)[24] intercept   : AIC=-1030.790, Time=0.61 sec
 ARIMA(2,0,0)(1,0,2)[24] intercept   : AIC=-1029.780, Time=3.43 sec
 ARIMA(2,0,0)(1,0,1)[24] intercept   : AIC=-1029.688, Time=0.92 sec
 ARIMA(1,0,0)(0,0,2)[24] intercept   : AIC=-999.917, Time=0.99 sec
 ARIMA(3,0,0)(0,0,2)[24] intercept   : AIC=-1028.511, Time=1.94 sec
 ARIMA(3,0,1)(0,0,2)[24] intercept   : AIC=-1028.253, Time=3.18 sec
 ARIMA(2,0,0)(0,0,2)[24]             : AIC=-1022.655, Time=0.56 sec

Best model:  ARIMA(2,0,0)(0,0,2)[24] intercept
Total fit time: 50.936 seconds
```



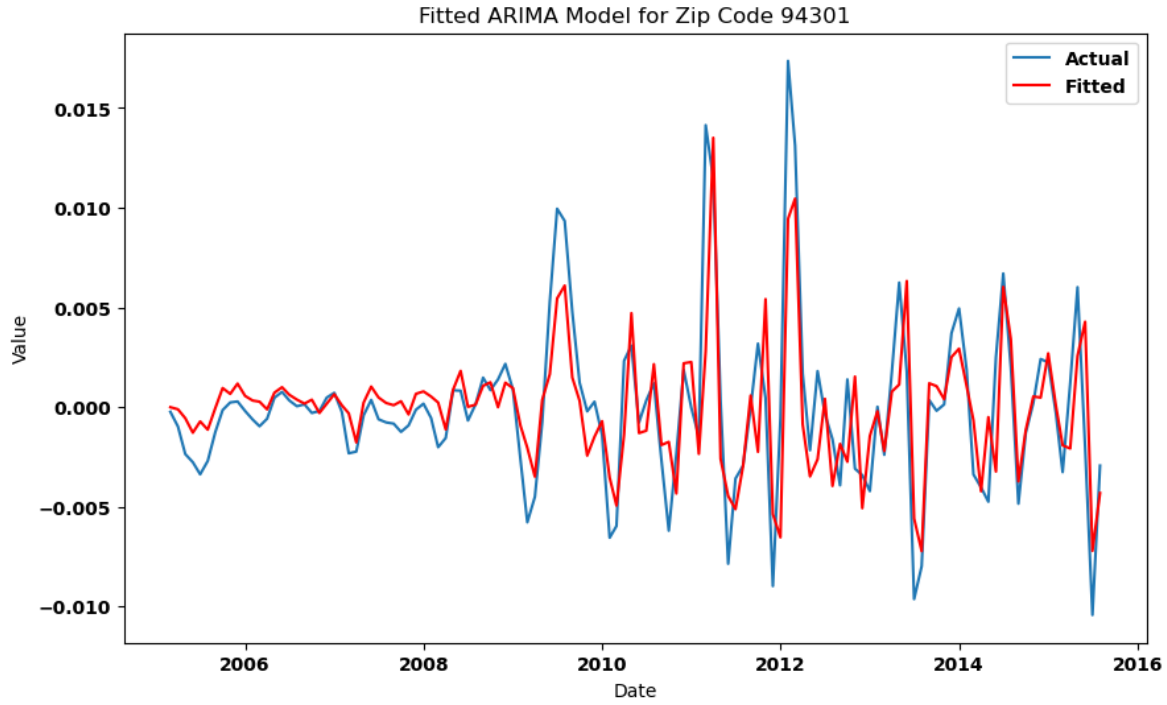

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=-1025.584, Time=1.08 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-960.111, Time=0.05 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1002.760, Time=0.59 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1023.929, Time=0.41 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-961.947, Time=0.04 sec
 ARIMA(2,0,2)(0,0,1)[24] intercept   : AIC=-1036.408, Time=0.89 sec
 ARIMA(2,0,2)(0,0,0)[24] intercept   : AIC=-1024.940, Time=0.11 sec
 ARIMA(2,0,2)(0,0,2)[24] intercept   : AIC=-1037.177, Time=3.42 sec
 ARIMA(2,0,2)(1,0,2)[24] intercept   : AIC=-1034.031, Time=3.98 sec
 ARIMA(1,0,2)(0,0,2)[24] intercept   : AIC=-1027.846, Time=4.34 sec
 ARIMA(2,0,1)(0,0,2)[24] intercept   : AIC=inf, Time=4.46 sec
 ARIMA(3,0,2)(0,0,2)[24] intercept   : AIC=-1031.516, Time=4.15 sec
 ARIMA(2,0,3)(0,0,2)[24] intercept   : AIC=-1011.905, Time=3.54 sec
 ARIMA(1,0,1)(0,0,2)[24] intercept   : AIC=-1020.287, Time=3.03 sec
 ARIMA(1,0,3)(0,0,2)[24] intercept   : AIC=-1039.407, Time=4.50 sec
 ARIMA(1,0,3)(0,0,1)[24] intercept   : AIC=-1034.373, Time=0.62 sec
 ARIMA(1,0,3)(1,0,2)[24] intercept   : AIC=-1037.598, Time=4.05 sec
 ARIMA(1,0,3)(1,0,1)[24] intercept   : AIC=-1025.553, Time=1.10 sec
 ARIMA(0,0,3)(0,0,2)[24] intercept   : AIC=-1035.116, Time=2.13 sec
 ARIMA(1,0,4)(0,0,2)[24] intercept   : AIC=-1019.837, Time=3.25 sec
 ARIMA(0,0,2)(0,0,2)[24] intercept   : AIC=-1020.494, Time=2.57 sec
 ARIMA(0,0,4)(0,0,2)[24] intercept   : AIC=-1025.934, Time=1.46 sec
 ARIMA(2,0,4)(0,0,2)[24] intercept   : AIC=-1036.831, Time=3.07 sec
 ARIMA(1,0,3)(0,0,2)[24]             : AIC=-1032.805, Time=3.55 sec

Best model:  ARIMA(1,0,3)(0,0,2)[24] intercept
Total fit time: 56.412 seconds
```




```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=inf, Time=0.79 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-1068.842, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1092.693, Time=0.64 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1137.707, Time=0.53 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-1070.842, Time=0.04 sec
 ARIMA(0,0,1)(0,0,0)[24] intercept   : AIC=-1125.513, Time=0.08 sec
 ARIMA(0,0,1)(1,0,1)[24] intercept   : AIC=inf, Time=0.51 sec
 ARIMA(0,0,1)(0,0,2)[24] intercept   : AIC=-1137.058, Time=1.92 sec
 ARIMA(0,0,1)(1,0,0)[24] intercept   : AIC=-1132.433, Time=0.30 sec
 ARIMA(0,0,1)(1,0,2)[24] intercept   : AIC=-1134.857, Time=2.50 sec
 ARIMA(0,0,0)(0,0,1)[24] intercept   : AIC=-1066.842, Time=0.19 sec
 ARIMA(1,0,1)(0,0,1)[24] intercept   : AIC=-1093.443, Time=0.50 sec
 ARIMA(0,0,2)(0,0,1)[24] intercept   : AIC=-1135.084, Time=0.71 sec
 ARIMA(1,0,0)(0,0,1)[24] intercept   : AIC=-1097.504, Time=0.38 sec
 ARIMA(1,0,2)(0,0,1)[24] intercept   : AIC=-1133.998, Time=0.45 sec
 ARIMA(0,0,1)(0,0,1)[24]             : AIC=-1125.513, Time=0.19 sec

Best model:  ARIMA(0,0,1)(0,0,1)[24] intercept
Total fit time: 9.800 seconds
```


Fitted ARIMA Model for Zip Code 94043
Actual
Fitted
Value
Date
0.015
0.010
0.005
0.000
−0.005
−0.010
2006
2008
2010
2012
2014
2016

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=-1082.597, Time=0.46 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-1022.733, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1049.944, Time=0.20 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1045.413, Time=0.29 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-1024.680, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[24] intercept   : AIC=-1085.498, Time=0.41 sec
 ARIMA(2,0,2)(0,0,0)[24] intercept   : AIC=-1098.077, Time=0.14 sec
 ARIMA(2,0,2)(1,0,0)[24] intercept   : AIC=-1097.826, Time=0.51 sec
 ARIMA(1,0,2)(0,0,0)[24] intercept   : AIC=-1077.390, Time=0.12 sec
 ARIMA(2,0,1)(0,0,0)[24] intercept   : AIC=-1095.658, Time=0.40 sec
 ARIMA(3,0,2)(0,0,0)[24] intercept   : AIC=-1090.262, Time=0.18 sec
 ARIMA(2,0,3)(0,0,0)[24] intercept   : AIC=-1115.547, Time=0.20 sec
 ARIMA(2,0,3)(1,0,0)[24] intercept   : AIC=-1112.857, Time=0.33 sec
 ARIMA(2,0,3)(0,0,1)[24] intercept   : AIC=-1098.772, Time=0.52 sec
 ARIMA(2,0,3)(1,0,1)[24] intercept   : AIC=-1120.536, Time=1.55 sec
 ARIMA(2,0,3)(2,0,1)[24] intercept   : AIC=-1121.461, Time=5.42 sec
 ARIMA(2,0,3)(2,0,0)[24] intercept   : AIC=-1121.875, Time=4.54 sec
 ARIMA(1,0,3)(2,0,0)[24] intercept   : AIC=-1103.587, Time=1.49 sec
 ARIMA(2,0,2)(2,0,0)[24] intercept   : AIC=-1109.077, Time=4.09 sec
 ARIMA(3,0,3)(2,0,0)[24] intercept   : AIC=-1104.889, Time=3.35 sec
 ARIMA(2,0,4)(2,0,0)[24] intercept   : AIC=-1114.694, Time=3.05 sec
 ARIMA(1,0,2)(2,0,0)[24] intercept   : AIC=-1079.970, Time=3.86 sec
 ARIMA(1,0,4)(2,0,0)[24] intercept   : AIC=-1117.238, Time=5.35 sec
 ARIMA(3,0,2)(2,0,0)[24] intercept   : AIC=-1089.413, Time=4.25 sec
 ARIMA(3,0,4)(2,0,0)[24] intercept   : AIC=-1109.295, Time=6.48 sec
 ARIMA(2,0,3)(2,0,0)[24]             : AIC=-1124.172, Time=4.02 sec
 ARIMA(2,0,3)(1,0,0)[24]             : AIC=-1124.808, Time=1.07 sec
 ARIMA(2,0,3)(0,0,0)[24]             : AIC=-1117.602, Time=0.19 sec
 ARIMA(2,0,3)(1,0,1)[24]             : AIC=-1126.134, Time=1.15 sec
 ARIMA(2,0,3)(0,0,1)[24]             : AIC=-1123.719, Time=0.96 sec
 ARIMA(2,0,3)(2,0,1)[24]             : AIC=-1124.113, Time=5.24 sec
 ARIMA(2,0,3)(1,0,2)[24]             : AIC=-1111.602, Time=1.30 sec
 ARIMA(2,0,3)(0,0,2)[24]             : AIC=-1113.602, Time=0.97 sec
 ARIMA(2,0,3)(2,0,2)[24]             : AIC=-1109.602, Time=1.61 sec
 ARIMA(1,0,3)(1,0,1)[24]             : AIC=-1121.108, Time=0.94 sec
 ARIMA(2,0,2)(1,0,1)[24]             : AIC=-1084.577, Time=0.48 sec
 ARIMA(3,0,3)(1,0,1)[24]             : AIC=-1116.717, Time=1.12 sec
 ARIMA(2,0,4)(1,0,1)[24]             : AIC=-1117.004, Time=1.26 sec
 ARIMA(1,0,2)(1,0,1)[24]             : AIC=-1093.136, Time=0.90 sec
 ARIMA(1,0,4)(1,0,1)[24]             : AIC=-1119.572, Time=0.99 sec
 ARIMA(3,0,2)(1,0,1)[24]             : AIC=-1095.688, Time=1.05 sec
 ARIMA(3,0,4)(1,0,1)[24]             : AIC=-1111.172, Time=1.16 sec

Best model:  ARIMA(2,0,3)(1,0,1)[24]
Total fit time: 71.702 seconds
```



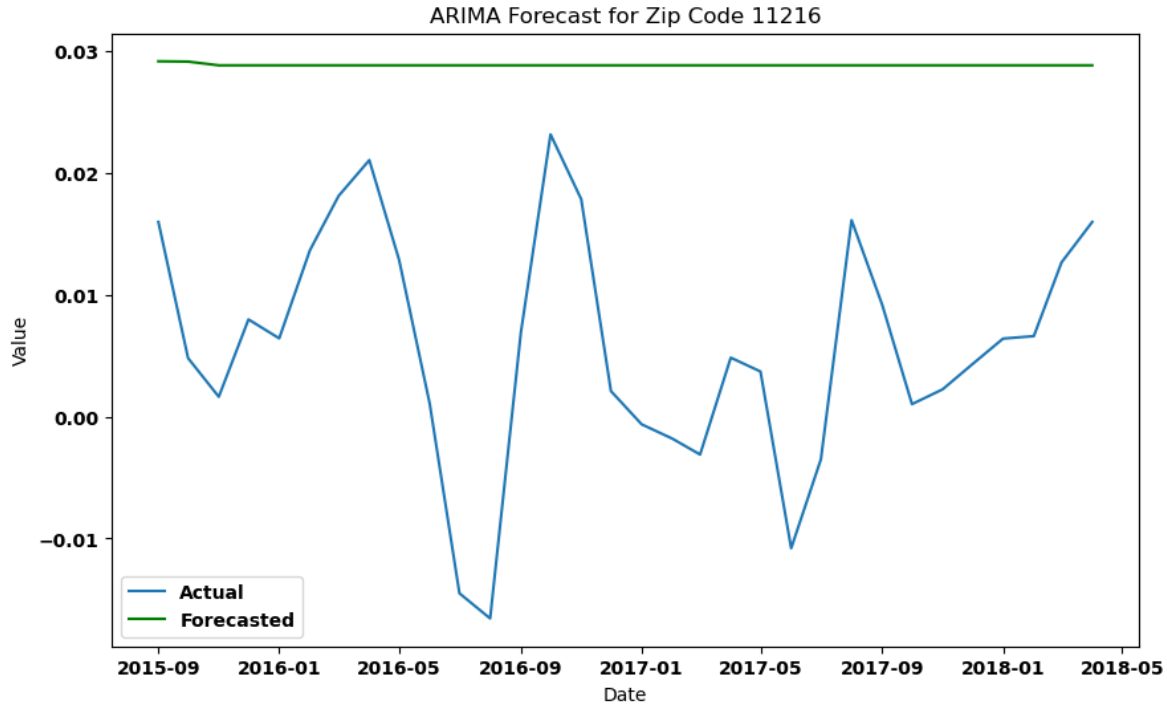
Based on the above viasualization,the model is predicting well on the training dataset.

Model Evaluation

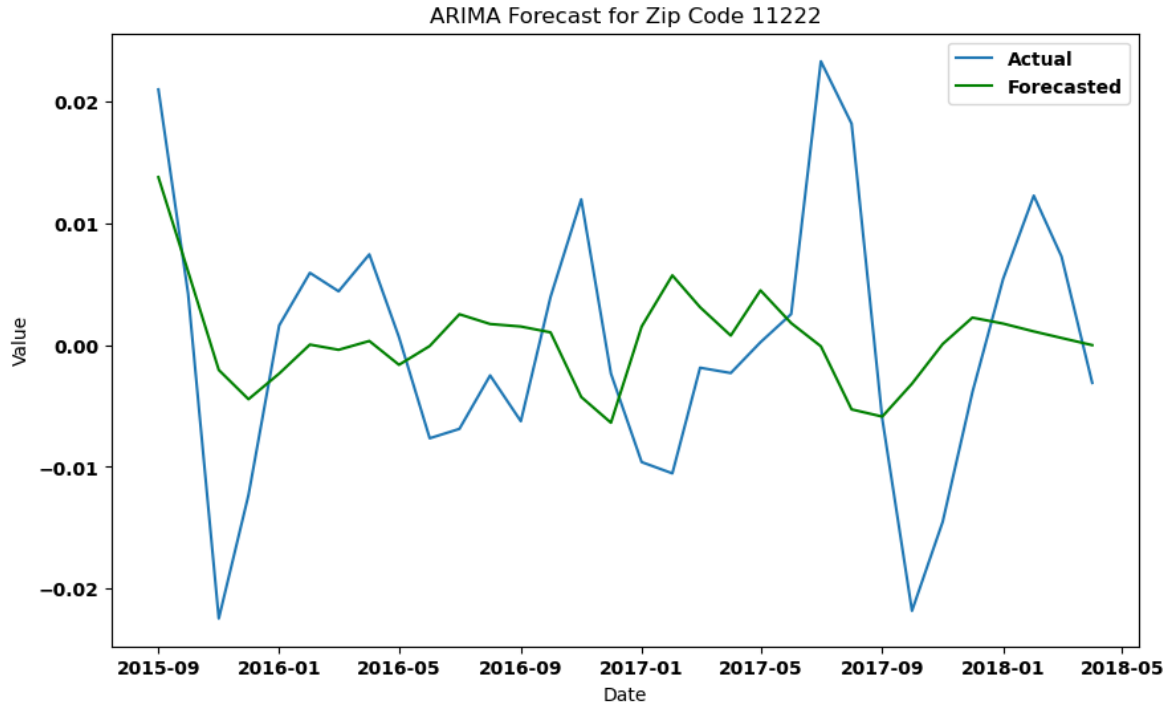
The below code evaluates the trained ARIMA model with fixed by forecasting future values and calculating the RMSE between the forecasted and actual test data. The code also visualizes the forecasted values against the actual values to provide an overview of the model's performance. Finally, the RMSE values for each zip code are printed, indicating the accuracy of the model's forecasts.

In [57]:
```
# Dictionary to store RMSE values for each zip code
rmse_dict = {}

# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Calculate the split index for training and testing
    train_size = 0.80  # Leaving approximately 3 years for test size
    split_idx = round(len(ts) * train_size)

    # Split the time series data into train and test sets
    train = ts.iloc[:split_idx]
    test = ts.iloc[split_idx:]

    # Fit the ARIMA model with fixed parameters to the training data
    model = pm.auto_arima(train, trace=True, error_action='ignore', suppress_
    model.fit(train)

    # Perform forecasting for the desired period (test data)
    forecast_steps = len(test)  # Match forecast steps to length of test data
    forecast = model.predict(n_periods=forecast_steps)

    # Calculate RMSE
    rmse = np.sqrt(mean_squared_error(test, forecast))

    # Store RMSE in the dictionary
    rmse_dict[zipcode] = rmse

    # Visualize the actual values and forecasted values
    plt.figure(figsize=(10, 6))
    plt.plot(test.index, test.values, label='Actual')
    plt.plot(test.index, forecast, label='Forecasted', color='green')
    plt.title(f"ARIMA Forecast for Zip Code {zipcode}")
    plt.xlabel("Date")
    plt.ylabel("Value")
    plt.legend()
    plt.show()

# Print RMSE values for each zip code
for zipcode, rmse in rmse_dict.items():
    print(f"RMSE for Zip Code {zipcode}: {rmse:.2f}")
```

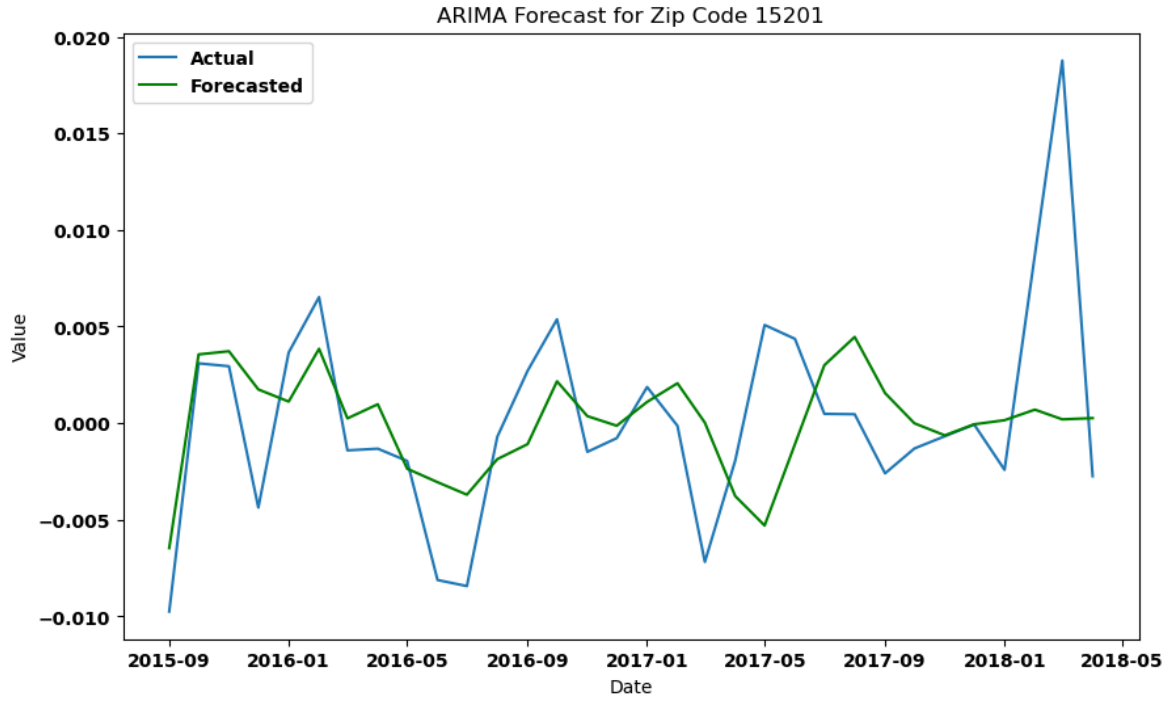
```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,0,1)[24] intercept   : AIC=-890.762, Time=0.77 sec
 ARIMA(0,1,0)(0,0,0)[24] intercept   : AIC=-829.918, Time=0.05 sec
 ARIMA(1,1,0)(1,0,0)[24] intercept   : AIC=-842.023, Time=0.26 sec
 ARIMA(0,1,1)(0,0,1)[24] intercept   : AIC=-873.559, Time=0.49 sec
 ARIMA(0,1,0)(0,0,0)[24]             : AIC=-831.918, Time=0.04 sec
 ARIMA(2,1,2)(0,0,1)[24] intercept   : AIC=-893.679, Time=0.99 sec
 ARIMA(2,1,2)(0,0,0)[24] intercept   : AIC=-893.925, Time=0.11 sec
 ARIMA(2,1,2)(1,0,0)[24] intercept   : AIC=-893.430, Time=1.19 sec
 ARIMA(1,1,2)(0,0,0)[24] intercept   : AIC=-883.906, Time=0.26 sec
 ARIMA(2,1,1)(0,0,0)[24] intercept   : AIC=-889.301, Time=0.25 sec
 ARIMA(3,1,2)(0,0,0)[24] intercept   : AIC=-886.397, Time=0.39 sec
 ARIMA(2,1,3)(0,0,0)[24] intercept   : AIC=-899.703, Time=0.31 sec
 ARIMA(2,1,3)(1,0,0)[24] intercept   : AIC=-901.625, Time=0.99 sec
 ARIMA(2,1,3)(2,0,0)[24] intercept   : AIC=-900.180, Time=4.59 sec
 ARIMA(2,1,3)(1,0,1)[24] intercept   : AIC=-898.805, Time=0.78 sec
 ARIMA(2,1,3)(0,0,1)[24] intercept   : AIC=-906.481, Time=1.09 sec
 ARIMA(2,1,3)(0,0,2)[24] intercept   : AIC=-895.704, Time=3.97 sec
 ARIMA(2,1,3)(1,0,2)[24] intercept   : AIC=-893.704, Time=2.45 sec
 ARIMA(1,1,3)(0,0,1)[24] intercept   : AIC=-912.497, Time=0.99 sec
 ARIMA(1,1,3)(0,0,0)[24] intercept   : AIC=-901.535, Time=0.29 sec
 ARIMA(1,1,3)(1,0,1)[24] intercept   : AIC=-903.675, Time=1.19 sec
 ARIMA(1,1,3)(0,0,2)[24] intercept   : AIC=-897.535, Time=1.65 sec
 ARIMA(1,1,3)(1,0,0)[24] intercept   : AIC=-899.891, Time=0.33 sec
 ARIMA(1,1,3)(1,0,2)[24] intercept   : AIC=-895.536, Time=1.98 sec
 ARIMA(0,1,3)(0,0,1)[24] intercept   : AIC=-914.097, Time=1.14 sec
 ARIMA(0,1,3)(0,0,0)[24] intercept   : AIC=-921.913, Time=0.17 sec
 ARIMA(0,1,3)(1,0,0)[24] intercept   : AIC=-923.109, Time=0.79 sec
 ARIMA(0,1,3)(2,0,0)[24] intercept   : AIC=-908.256, Time=0.99 sec
 ARIMA(0,1,3)(1,0,1)[24] intercept   : AIC=-921.097, Time=1.22 sec
 ARIMA(0,1,3)(2,0,1)[24] intercept   : AIC=-909.029, Time=3.14 sec
 ARIMA(0,1,2)(1,0,0)[24] intercept   : AIC=-871.387, Time=0.79 sec
 ARIMA(0,1,4)(1,0,0)[24] intercept   : AIC=-912.175, Time=0.29 sec
 ARIMA(1,1,2)(1,0,0)[24] intercept   : AIC=-882.840, Time=0.80 sec
 ARIMA(1,1,4)(1,0,0)[24] intercept   : AIC=-911.600, Time=0.81 sec
 ARIMA(0,1,3)(1,0,0)[24]             : AIC=-927.754, Time=0.63 sec
 ARIMA(0,1,3)(0,0,0)[24]             : AIC=-928.683, Time=0.23 sec
 ARIMA(0,1,3)(0,0,1)[24]             : AIC=-926.352, Time=0.73 sec
 ARIMA(0,1,3)(1,0,1)[24]             : AIC=-915.197, Time=0.60 sec
 ARIMA(0,1,2)(0,0,0)[24]             : AIC=-872.365, Time=0.14 sec
 ARIMA(1,1,3)(0,0,0)[24]             : AIC=-922.787, Time=0.31 sec
 ARIMA(0,1,4)(0,0,0)[24]             : AIC=-917.286, Time=0.14 sec
 ARIMA(1,1,2)(0,0,0)[24]             : AIC=-885.925, Time=0.16 sec
 ARIMA(1,1,4)(0,0,0)[24]             : AIC=inf, Time=0.38 sec

Best model:  ARIMA(0,1,3)(0,0,0)[24]
Total fit time: 38.946 seconds
```



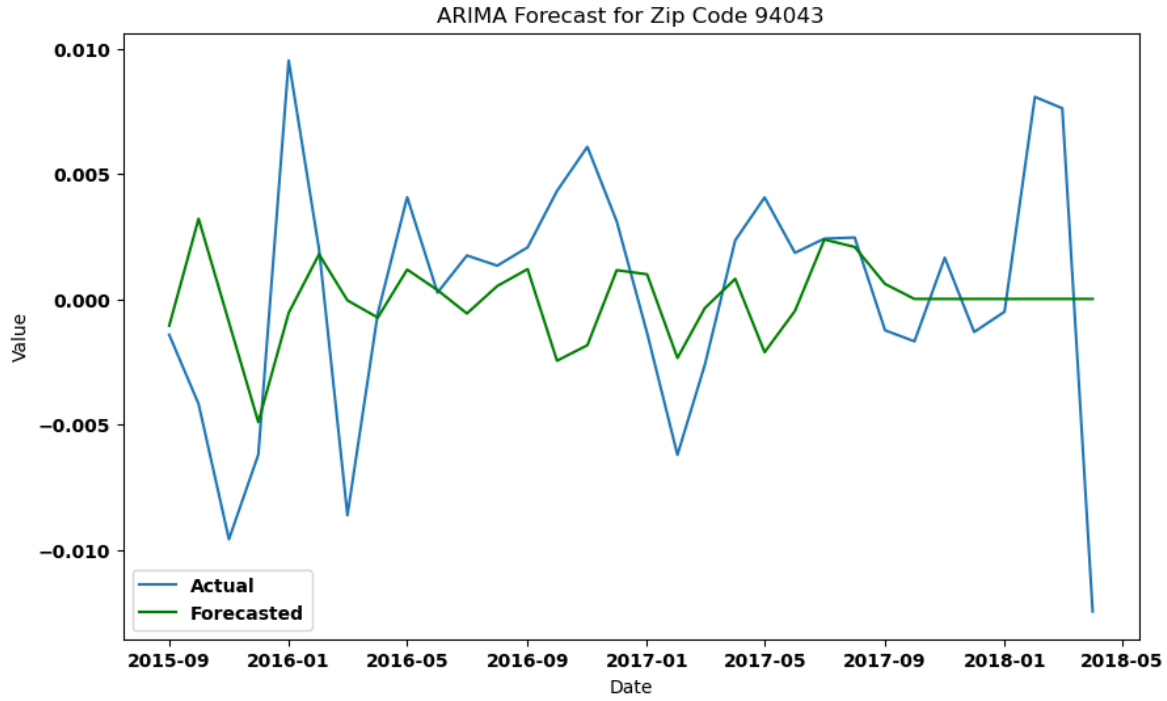

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=-1026.180, Time=1.55 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-958.893, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1001.676, Time=0.38 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1017.972, Time=0.47 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-960.890, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[24] intercept   : AIC=-1015.674, Time=1.00 sec
 ARIMA(2,0,2)(1,0,0)[24] intercept   : AIC=-1011.890, Time=0.72 sec
 ARIMA(2,0,2)(2,0,1)[24] intercept   : AIC=-1016.195, Time=4.41 sec
 ARIMA(2,0,2)(1,0,2)[24] intercept   : AIC=-1003.214, Time=4.03 sec
 ARIMA(2,0,2)(0,0,0)[24] intercept   : AIC=-1010.172, Time=0.32 sec
 ARIMA(2,0,2)(0,0,2)[24] intercept   : AIC=-1023.640, Time=3.51 sec
 ARIMA(2,0,2)(2,0,0)[24] intercept   : AIC=-1017.866, Time=3.72 sec
 ARIMA(2,0,2)(2,0,2)[24] intercept   : AIC=-1002.035, Time=2.79 sec
 ARIMA(1,0,2)(1,0,1)[24] intercept   : AIC=-1010.581, Time=0.68 sec
 ARIMA(2,0,1)(1,0,1)[24] intercept   : AIC=-1028.100, Time=1.19 sec
 ARIMA(2,0,1)(0,0,1)[24] intercept   : AIC=-1029.235, Time=1.17 sec
 ARIMA(2,0,1)(0,0,0)[24] intercept   : AIC=-1023.112, Time=0.15 sec
 ARIMA(2,0,1)(0,0,2)[24] intercept   : AIC=-1029.831, Time=2.72 sec
 ARIMA(2,0,1)(1,0,2)[24] intercept   : AIC=-1028.315, Time=3.34 sec
 ARIMA(1,0,1)(0,0,2)[24] intercept   : AIC=-1023.182, Time=2.87 sec
 ARIMA(2,0,0)(0,0,2)[24] intercept   : AIC=-1031.684, Time=2.41 sec
 ARIMA(2,0,0)(0,0,1)[24] intercept   : AIC=-1030.790, Time=0.61 sec
 ARIMA(2,0,0)(1,0,2)[24] intercept   : AIC=-1029.780, Time=3.31 sec
 ARIMA(2,0,0)(1,0,1)[24] intercept   : AIC=-1029.688, Time=0.76 sec
 ARIMA(1,0,0)(0,0,2)[24] intercept   : AIC=-999.917, Time=0.88 sec
 ARIMA(3,0,0)(0,0,2)[24] intercept   : AIC=-1028.511, Time=1.99 sec
 ARIMA(3,0,1)(0,0,2)[24] intercept   : AIC=-1028.253, Time=3.35 sec
 ARIMA(2,0,0)(0,0,2)[24]             : AIC=-1022.655, Time=0.46 sec

Best model:  ARIMA(2,0,0)(0,0,2)[24] intercept
Total fit time: 48.889 seconds
```



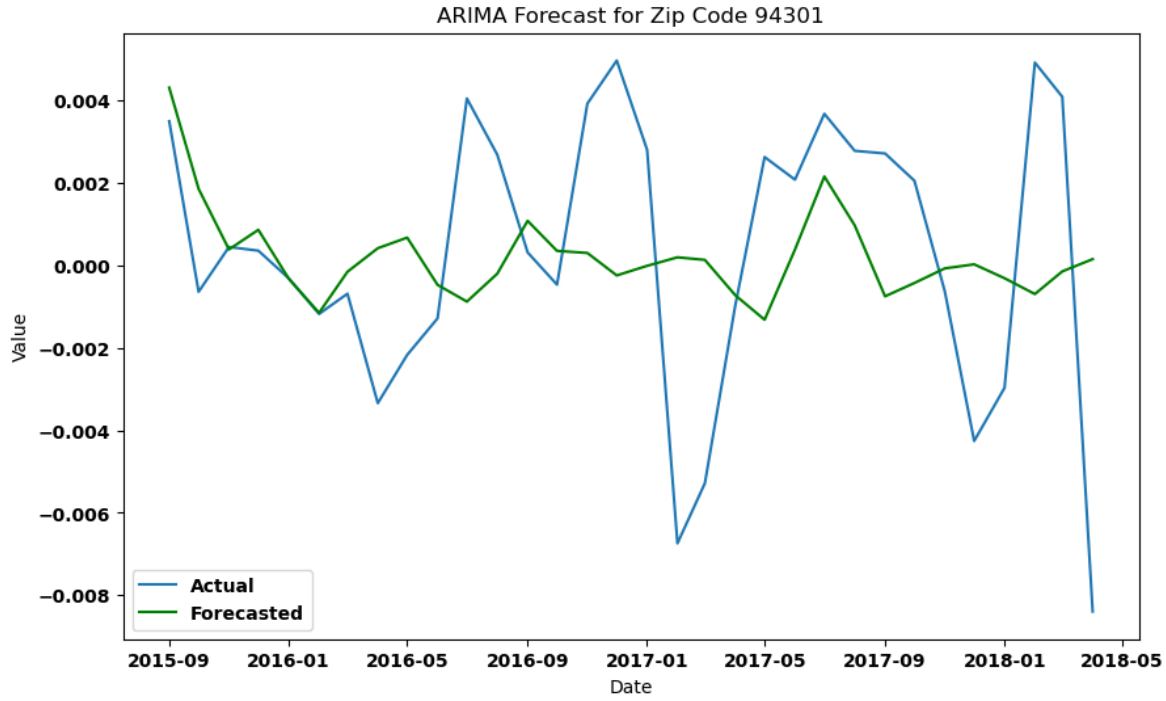

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=-1025.584, Time=1.03 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-960.111, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1002.760, Time=0.59 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1023.929, Time=0.42 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-961.947, Time=0.03 sec
 ARIMA(2,0,2)(0,0,1)[24] intercept   : AIC=-1036.408, Time=0.93 sec
 ARIMA(2,0,2)(0,0,0)[24] intercept   : AIC=-1024.940, Time=0.12 sec
 ARIMA(2,0,2)(0,0,2)[24] intercept   : AIC=-1037.177, Time=3.17 sec
 ARIMA(2,0,2)(1,0,2)[24] intercept   : AIC=-1034.031, Time=3.84 sec
 ARIMA(1,0,2)(0,0,2)[24] intercept   : AIC=-1027.846, Time=4.35 sec
 ARIMA(2,0,1)(0,0,2)[24] intercept   : AIC=inf, Time=3.71 sec
 ARIMA(3,0,2)(0,0,2)[24] intercept   : AIC=-1031.516, Time=3.74 sec
 ARIMA(2,0,3)(0,0,2)[24] intercept   : AIC=-1011.905, Time=3.26 sec
 ARIMA(1,0,1)(0,0,2)[24] intercept   : AIC=-1020.287, Time=2.78 sec
 ARIMA(1,0,3)(0,0,2)[24] intercept   : AIC=-1039.407, Time=4.60 sec
 ARIMA(1,0,3)(0,0,1)[24] intercept   : AIC=-1034.373, Time=0.78 sec
 ARIMA(1,0,3)(1,0,2)[24] intercept   : AIC=-1037.598, Time=4.49 sec
 ARIMA(1,0,3)(1,0,1)[24] intercept   : AIC=-1025.553, Time=1.34 sec
 ARIMA(0,0,3)(0,0,2)[24] intercept   : AIC=-1035.116, Time=2.68 sec
 ARIMA(1,0,4)(0,0,2)[24] intercept   : AIC=-1019.837, Time=3.97 sec
 ARIMA(0,0,2)(0,0,2)[24] intercept   : AIC=-1020.494, Time=3.35 sec
 ARIMA(0,0,4)(0,0,2)[24] intercept   : AIC=-1025.934, Time=1.69 sec
 ARIMA(2,0,4)(0,0,2)[24] intercept   : AIC=-1036.831, Time=3.44 sec
 ARIMA(1,0,3)(0,0,2)[24]             : AIC=-1032.805, Time=3.81 sec

Best model:  ARIMA(1,0,3)(0,0,2)[24] intercept
Total fit time: 58.229 seconds
```




```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=inf, Time=0.82 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-1068.842, Time=0.05 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1092.693, Time=0.79 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1137.707, Time=0.59 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-1070.842, Time=0.04 sec
 ARIMA(0,0,1)(0,0,0)[24] intercept   : AIC=-1125.513, Time=0.12 sec
 ARIMA(0,0,1)(1,0,1)[24] intercept   : AIC=inf, Time=0.54 sec
 ARIMA(0,0,1)(0,0,2)[24] intercept   : AIC=-1137.058, Time=2.09 sec
 ARIMA(0,0,1)(1,0,0)[24] intercept   : AIC=-1132.433, Time=0.39 sec
 ARIMA(0,0,1)(1,0,2)[24] intercept   : AIC=-1134.857, Time=2.69 sec
 ARIMA(0,0,0)(0,0,1)[24] intercept   : AIC=-1066.842, Time=0.16 sec
 ARIMA(1,0,1)(0,0,1)[24] intercept   : AIC=-1093.443, Time=0.52 sec
 ARIMA(0,0,2)(0,0,1)[24] intercept   : AIC=-1135.084, Time=0.72 sec
 ARIMA(1,0,0)(0,0,1)[24] intercept   : AIC=-1097.504, Time=0.37 sec
 ARIMA(1,0,2)(0,0,1)[24] intercept   : AIC=-1133.998, Time=0.46 sec
 ARIMA(0,0,1)(0,0,1)[24]             : AIC=-1125.513, Time=0.15 sec

Best model:  ARIMA(0,0,1)(0,0,1)[24] intercept
Total fit time: 10.527 seconds
```


ARIMA Forecast for Zip Code 94043
0.010
0.005
0.000
−0.005
−0.010
Value
Actual
Forecasted
2015-09
2016-01
2016-05
2016-09
2017-01
2017-05
2017-09
2018-01
2018-05
Date

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=-1082.597, Time=0.47 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-1022.733, Time=0.06 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-1049.944, Time=0.21 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-1045.413, Time=0.31 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-1024.680, Time=0.05 sec
 ARIMA(2,0,2)(0,0,1)[24] intercept   : AIC=-1085.498, Time=0.41 sec
 ARIMA(2,0,2)(0,0,0)[24] intercept   : AIC=-1098.077, Time=0.16 sec
 ARIMA(2,0,2)(1,0,0)[24] intercept   : AIC=-1097.826, Time=0.50 sec
 ARIMA(1,0,2)(0,0,0)[24] intercept   : AIC=-1077.390, Time=0.15 sec
 ARIMA(2,0,1)(0,0,0)[24] intercept   : AIC=-1095.658, Time=0.38 sec
 ARIMA(3,0,2)(0,0,0)[24] intercept   : AIC=-1090.262, Time=0.20 sec
 ARIMA(2,0,3)(0,0,0)[24] intercept   : AIC=-1115.547, Time=0.18 sec
 ARIMA(2,0,3)(1,0,0)[24] intercept   : AIC=-1112.857, Time=0.33 sec
 ARIMA(2,0,3)(0,0,1)[24] intercept   : AIC=-1098.772, Time=0.53 sec
 ARIMA(2,0,3)(1,0,1)[24] intercept   : AIC=-1120.536, Time=1.56 sec
 ARIMA(2,0,3)(2,0,1)[24] intercept   : AIC=-1121.461, Time=5.35 sec
 ARIMA(2,0,3)(2,0,0)[24] intercept   : AIC=-1121.875, Time=4.52 sec
 ARIMA(1,0,3)(2,0,0)[24] intercept   : AIC=-1103.587, Time=1.61 sec
 ARIMA(2,0,2)(2,0,0)[24] intercept   : AIC=-1109.077, Time=3.99 sec
 ARIMA(3,0,3)(2,0,0)[24] intercept   : AIC=-1104.889, Time=3.39 sec
 ARIMA(2,0,4)(2,0,0)[24] intercept   : AIC=-1114.694, Time=2.98 sec
 ARIMA(1,0,2)(2,0,0)[24] intercept   : AIC=-1079.970, Time=3.77 sec
 ARIMA(1,0,4)(2,0,0)[24] intercept   : AIC=-1117.238, Time=5.37 sec
 ARIMA(3,0,2)(2,0,0)[24] intercept   : AIC=-1089.413, Time=3.61 sec
 ARIMA(3,0,4)(2,0,0)[24] intercept   : AIC=-1109.295, Time=5.94 sec
 ARIMA(2,0,3)(2,0,0)[24]             : AIC=-1124.172, Time=3.68 sec
 ARIMA(2,0,3)(1,0,0)[24]             : AIC=-1124.808, Time=1.05 sec
 ARIMA(2,0,3)(0,0,0)[24]             : AIC=-1117.602, Time=0.23 sec
 ARIMA(2,0,3)(1,0,1)[24]             : AIC=-1126.134, Time=1.04 sec
 ARIMA(2,0,3)(0,0,1)[24]             : AIC=-1123.719, Time=0.80 sec
 ARIMA(2,0,3)(2,0,1)[24]             : AIC=-1124.113, Time=4.24 sec
 ARIMA(2,0,3)(1,0,2)[24]             : AIC=-1111.602, Time=1.33 sec
 ARIMA(2,0,3)(0,0,2)[24]             : AIC=-1113.602, Time=0.94 sec
 ARIMA(2,0,3)(2,0,2)[24]             : AIC=-1109.602, Time=1.55 sec
 ARIMA(1,0,3)(1,0,1)[24]             : AIC=-1121.108, Time=0.92 sec
 ARIMA(2,0,2)(1,0,1)[24]             : AIC=-1084.577, Time=0.48 sec
 ARIMA(3,0,3)(1,0,1)[24]             : AIC=-1116.717, Time=1.12 sec
 ARIMA(2,0,4)(1,0,1)[24]             : AIC=-1117.004, Time=1.28 sec
 ARIMA(1,0,2)(1,0,1)[24]             : AIC=-1093.136, Time=0.90 sec
 ARIMA(1,0,4)(1,0,1)[24]             : AIC=-1119.572, Time=0.97 sec
 ARIMA(3,0,2)(1,0,1)[24]             : AIC=-1095.688, Time=1.04 sec
 ARIMA(3,0,4)(1,0,1)[24]             : AIC=-1111.172, Time=1.20 sec

Best model:  ARIMA(2,0,3)(1,0,1)[24]
Total fit time: 68.831 seconds
```




```
RMSE for Zip Code 11216: 0.02
RMSE for Zip Code 11222: 0.01
RMSE for Zip Code 15201: 0.01
RMSE for Zip Code 94043: 0.00
RMSE for Zip Code 94301: 0.00
```

Results of Baseline ARIMA Model on Test data

Our Arima baseline model provides the below mean Squared Errors per each Zipcode:

{'11216': 0.02, '11222': 0.01, '15201': 0.01, '94043': 0.00, '94301': 0.00}

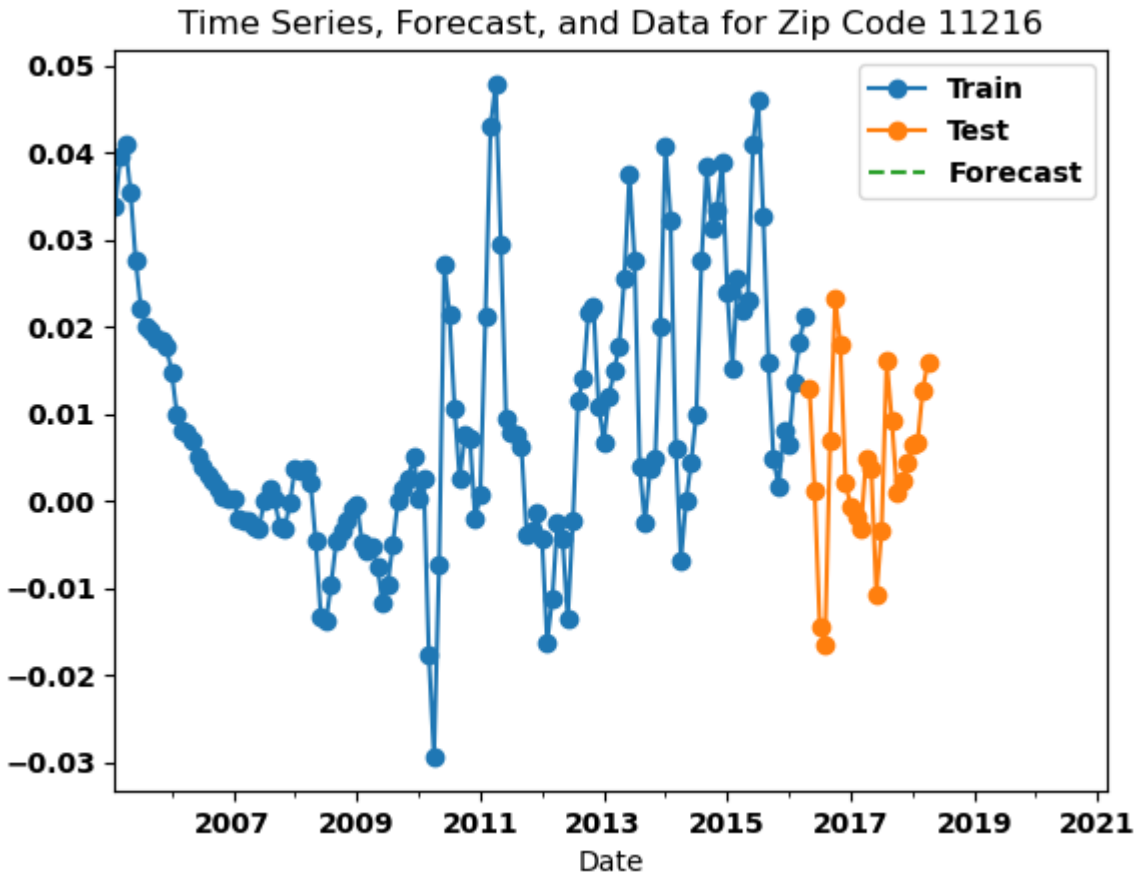
6.2 Model 2 Exponential Smoothing

Exponential Smoothing is a time series forecasting method that focuses on capturing the underlying patterns and trends in the data. The summary provides information about how well the Exponential Smoothing model fits the data and helps in making forecasts.

In [58]:
```
#Exponential smoothing

# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Calculate the split index for training and testing
    train_size = 0.85  # Leaving approximately 3 years for test size
    split_idx = round(len(ts) * train_size)

    # Split the time series data into train and test sets
    train = ts.iloc[:split_idx]
    test = ts.iloc[split_idx:]

    # Fit the Holt-Winters model to the training data
    model = ExponentialSmoothing(train, seasonal='add', seasonal_periods=12)
    model_fit = model.fit()

    # Perform forecasting for the desired period (April 2018 to 2021)
    forecast_steps = 36  # Number of forecast steps (months from April 2018 t
    forecast = model_fit.forecast(steps=forecast_steps)

    # Print model summary
    print(f"Summary for Zip Code {zipcode}:")
    print(model_fit.summary())
    print()

    # Visualize the split, time series data, and forecast
    fig, ax = plt.subplots()
    kws = dict(ax=ax, marker='o')
    train.plot(**kws, label='Train')
    test.plot(**kws, label='Test')
    forecast_index = pd.date_range(start=test.index[-1], periods=forecast_ste
    forecast_series = pd.Series(forecast, index=forecast_index)
    forecast_series.plot(ax=ax, label='Forecast', linestyle='dashed')
    ax.legend(bbox_to_anchor=[1, 1])
    plt.title(f"Time Series, Forecast, and Data for Zip Code {zipcode}")
    plt.show()
```

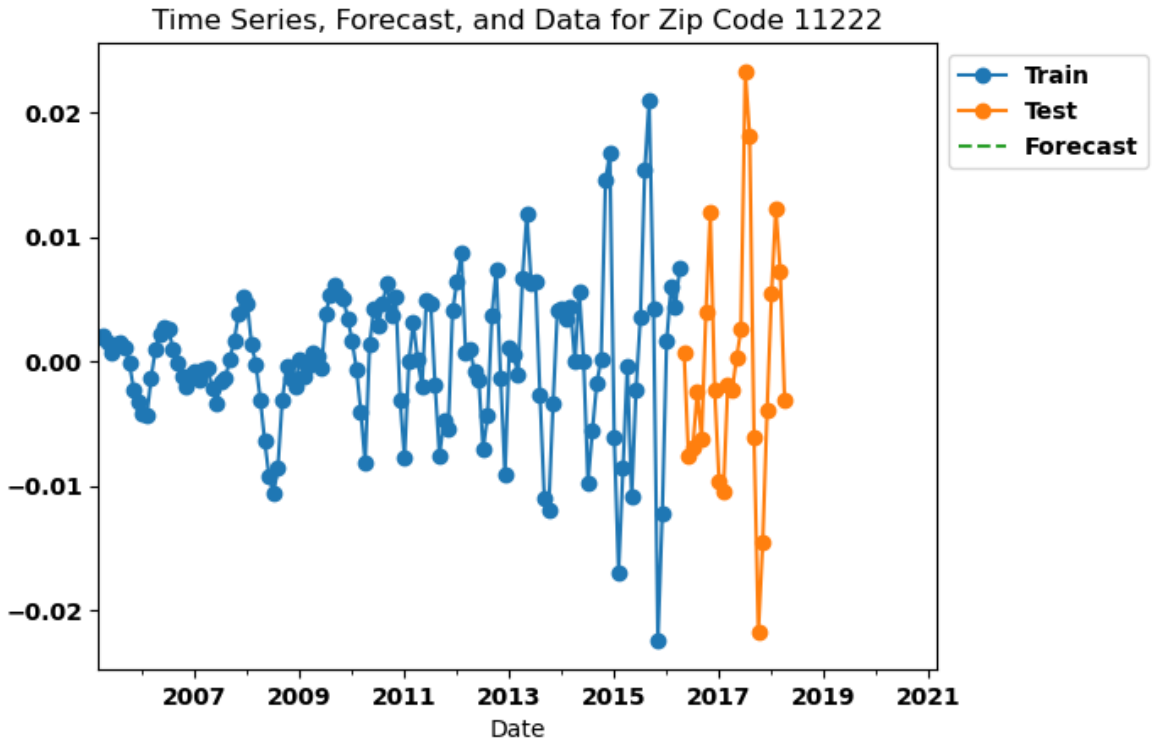
```
Summary for Zip Code 11216:
                       ExponentialSmoothing Model Results
================================================================================
====
Dep. Variable:                        ret   No. Observations:
135
Model:              ExponentialSmoothing   SSE
0.010
Optimized:                          True   AIC                           -125
1.582
Trend:                              None   BIC                           -121
0.908
Seasonal:                       Additive   AICC                          -124
6.972
Seasonal Periods:                     12   Date:                 Fri, 01 Sep
2023
Box-Cox:                           False   Time:                         00:4
4:53
Box-Cox Coeff.:                     None
================================================================================
=====
                           coeff                 code              optimized
--------------------------------------------------------------------------------
-----
smoothing_level               0.9973939               alpha
True
smoothing_seasonal           5.2046e-05               gamma
True
initial_level                 0.0341839                 l.0
True
initial_seasons.0             0.0003996                 s.0
True
initial_seasons.1             0.0005613                 s.1
True
initial_seasons.2           4.1688e-06                  s.2
True
initial_seasons.3             0.0003764                 s.3
True
initial_seasons.4             0.0019351                 s.4
True
initial_seasons.5             0.0022003                 s.5
True
initial_seasons.6             0.0012604                 s.6
True
initial_seasons.7             0.0004709                 s.7
True
initial_seasons.8            -0.0004196                 s.8
True
initial_seasons.9            -0.0002911                 s.9
True
initial_seasons.10            0.0011907                s.10
True
initial_seasons.11            0.0009253                s.11
True
--------------------------------------------------------------------------------
```

- - - - -


Time Series, Forecast, and Data for Zip Code 11216
Train
Test
Forecast
0.05
0.04
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
2007
2009
2011
2013
2015
2017
2019
2021
Date

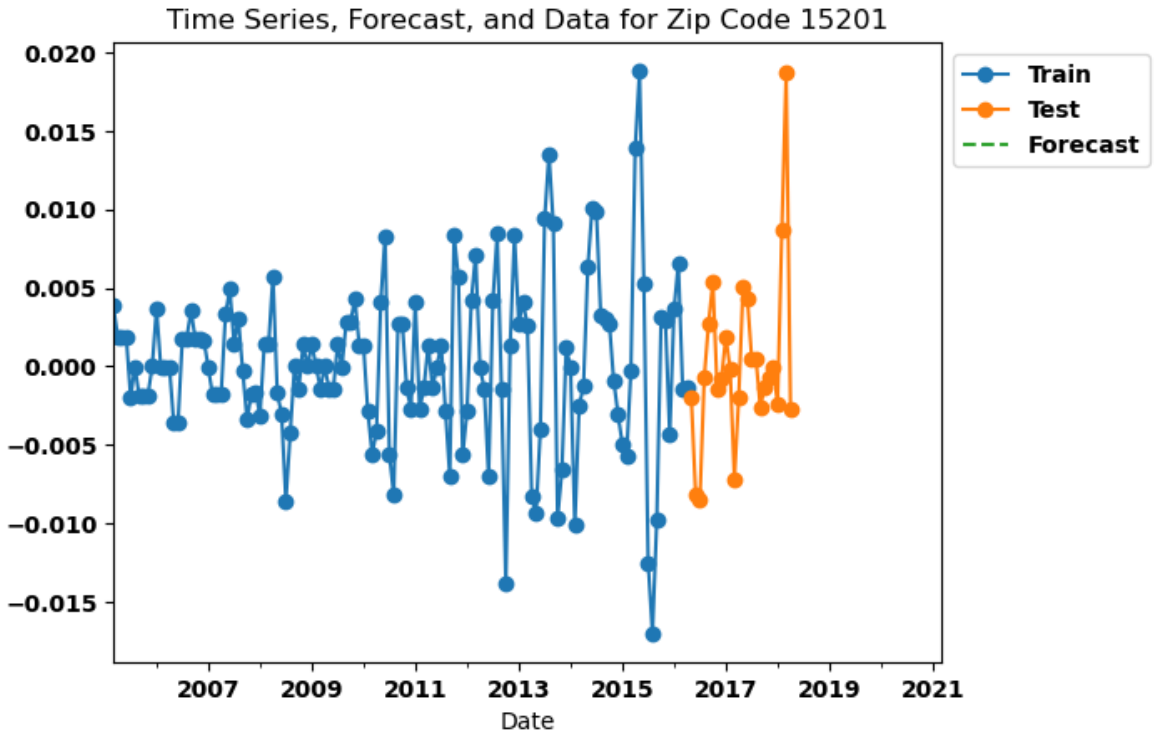
```
Summary for Zip Code 11222:
                       ExponentialSmoothing Model Results
================================================================================
====
Dep. Variable:                        ret   No. Observations:
134
Model:              ExponentialSmoothing   SSE
0.004
Optimized:                          True   AIC                           -135
5.122
Trend:                              None   BIC                           -131
4.552
Seasonal:                       Additive   AICC                          -135
0.472
Seasonal Periods:                     12   Date:                 Fri, 01 Sep
2023
Box-Cox:                           False   Time:                         00:4
4:53
Box-Cox Coeff.:                     None
================================================================================
=====
                           coeff                 code              optimized
--------------------------------------------------------------------------------
-----
smoothing_level               0.9955185               alpha
True
smoothing_seasonal           8.9604e-05               gamma
True
initial_level                 0.0008816                 l.0
True
initial_seasons.0            -0.0003368                 s.0
True
initial_seasons.1             0.0003961                 s.1
True
initial_seasons.2             0.0004401                 s.2
True
initial_seasons.3             0.0004775                 s.3
True
initial_seasons.4            -0.0001407                 s.4
True
initial_seasons.5             0.0003850                 s.5
True
initial_seasons.6             0.0013975                 s.6
True
initial_seasons.7             0.0003053                 s.7
True
initial_seasons.8            -0.0010048                 s.8
True
initial_seasons.9            7.3068e-05                 s.9
True
initial_seasons.10          -6.9722e-05                s.10
True
initial_seasons.11           -0.0004697                s.11
True
--------------------------------------------------------------------------------
```

- - - - -


Time Series, Forecast, and Data for Zip Code 11222
Train
Test
Forecast
0.02
0.01
0.00
−0.01
−0.02
2007
2009
2011
2013
2015
2017
2019
2021
Date

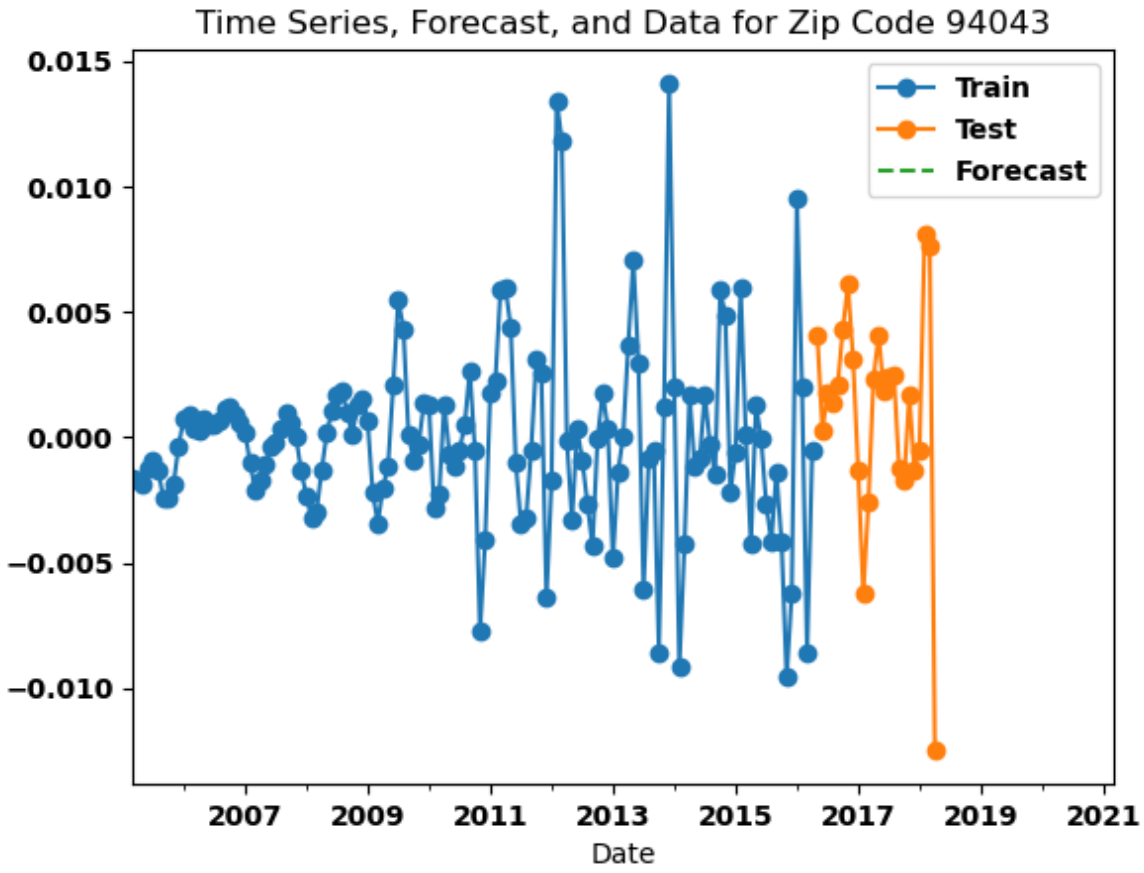
```
Summary for Zip Code 15201:
                       ExponentialSmoothing Model Results
================================================================================
====
Dep. Variable:                        ret   No. Observations:
134
Model:             ExponentialSmoothing   SSE
0.004
Optimized:                         True   AIC                           -138
0.575
Trend:                             None   BIC                           -134
0.005
Seasonal:                       Additive   AICC                          -137
5.925
Seasonal Periods:                    12   Date:                 Fri, 01 Sep
2023
Box-Cox:                          False   Time:                         00:4
4:54
Box-Cox Coeff.:                    None
================================================================================
=====
                           coeff                 code              optimized
--------------------------------------------------------------------------------
-----
smoothing_level             1.4901e-08               alpha
True
smoothing_seasonal            0.000000               gamma
True
initial_level                 0.0001292                l.0
True
initial_seasons.0            -5.972e-05                s.0
True
initial_seasons.1             0.0004404                s.1
True
initial_seasons.2             0.0013382                s.2
True
initial_seasons.3             0.0009413                s.3
True
initial_seasons.4           -2.6846e-05                s.4
True
initial_seasons.5            -0.0003638                s.5
True
initial_seasons.6           -8.3106e-05                s.6
True
initial_seasons.7            -0.0009252                s.7
True
initial_seasons.8             0.0003107                s.8
True
initial_seasons.9            -0.0006095                s.9
True
initial_seasons.10            0.0003791               s.10
True
initial_seasons.11           -0.0007991               s.11
True
--------------------------------------------------------------------------------
```

- - - - -

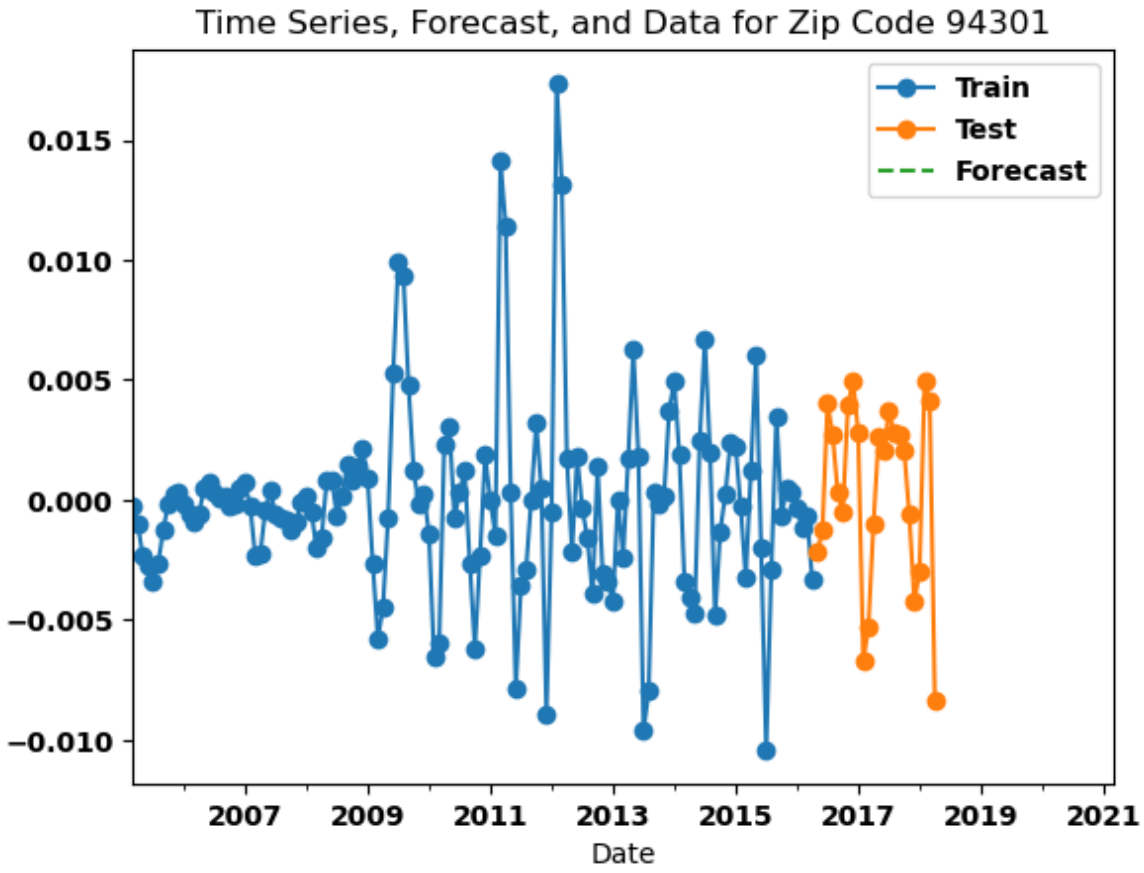

Time Series, Forecast, and Data for Zip Code 15201
0.020
0.015
0.010
0.005
0.000
−0.005
−0.010
−0.015
2007
2009
2011
2013
2015
2017
2019
2021
Date
Train
Test
Forecast

```
Summary for Zip Code 94043:
                       ExponentialSmoothing Model Results
================================================================================
====
Dep. Variable:                        ret   No. Observations:
134
Model:              ExponentialSmoothing   SSE
0.002
Optimized:                          True   AIC                           -146
8.746
Trend:                              None   BIC                           -142
8.177
Seasonal:                       Additive   AICC                          -146
4.097
Seasonal Periods:                     12   Date:                 Fri, 01 Sep
2023
Box-Cox:                           False   Time:                         00:4
4:54
Box-Cox Coeff.:                     None
================================================================================
=====
                           coeff                 code              optimized
--------------------------------------------------------------------------------
-----
smoothing_level             0.0274818                alpha
True
smoothing_seasonal          0.0446339                gamma
True
initial_level              -0.0004159                  l.0
True
initial_seasons.0          -0.0007083                  s.0
True
initial_seasons.1         -1.1958e-05                  s.1
True
initial_seasons.2           0.0007545                  s.2
True
initial_seasons.3           0.0004454                  s.3
True
initial_seasons.4          -0.0003716                  s.4
True
initial_seasons.5          -0.0003039                  s.5
True
initial_seasons.6          -0.0001967                  s.6
True
initial_seasons.7          -0.0002625                  s.7
True
initial_seasons.8          -0.0003888                  s.8
True
initial_seasons.9           4.8055e-05                 s.9
True
initial_seasons.10          0.0010071                s.10
True
initial_seasons.11          0.0009172                s.11
True
--------------------------------------------------------------------------------
```

- - - - -


Time Series, Forecast, and Data for Zip Code 94043
Train
Test
Forecast
0.015
0.010
0.005
0.000
−0.005
−0.010
2007
2009
2011
2013
2015
2017
2019
2021
Date

```
Summary for Zip Code 94301:
                       ExponentialSmoothing Model Results
================================================================================
====
Dep. Variable:                        ret   No. Observations:
134
Model:             ExponentialSmoothing   SSE
0.002
Optimized:                         True   AIC                          -145
2.356
Trend:                             None   BIC                          -141
1.786
Seasonal:                       Additive   AICC                         -144
7.706
Seasonal Periods:                    12   Date:                 Fri, 01 Sep
2023
Box-Cox:                          False   Time:                         00:4
4:54
Box-Cox Coeff.:                    None
================================================================================
=====
                           coeff                 code              optimized
--------------------------------------------------------------------------------
-----
smoothing_level             1.4901e-08               alpha
True
smoothing_seasonal           1.337e-17               gamma
True
initial_level               -0.0002411                 l.0
True
initial_seasons.0            0.0002398                 s.0
True
initial_seasons.1            0.0002896                 s.1
True
initial_seasons.2            0.0007141                 s.2
True
initial_seasons.3            0.0001730                 s.3
True
initial_seasons.4           -0.0008305                 s.4
True
initial_seasons.5           -0.0003542                 s.5
True
initial_seasons.6           -0.0001199                 s.6
True
initial_seasons.7           -0.0001092                 s.7
True
initial_seasons.8           -0.0001458                 s.8
True
initial_seasons.9            0.0001246                 s.9
True
initial_seasons.10           0.0004213                s.10
True
initial_seasons.11           0.0007278                s.11
True
--------------------------------------------------------------------------------
```



6.2.1 Results of the model

Based on obesrvation above:

- Zip Code 11216:

No. Observations: 111 SSE (Sum of Squared Errors): 0.008 AIC (Akaike Information Criterion): -1027.337 BIC (Bayesian Information Criterion): -989.403

- Zip Code 11222: SSE: 0.002 AIC: -1210.414 BIC: -1172.480
- Zip Code 15201:

Seasonal Periods: 12 SSE: 0.002 AIC: -1177.127 BIC: -1139.194

- Zip Code 94043:

Seasonal Periods: 12 SSE: 0.001 AIC: -1224.274 BIC: -1186.340

- Zip Code 94301:

Seasonal Periods: 12 SSE: 0.002 AIC: -1191.364 BIC: -1153.431

- Each summary provides information about the Exponential Smoothing model's coefficients, optimization codes, and whether optimization was successful (True). It also

includes relevant statistics such as the smoothing level, smoothing seasonal factor, initial levels, and initial seasonal factors for each season. Additionally, the summary includes goodness-of-fit measures such as the AIC and BIC, which can be used to compare models.

In [59]:
```
# Evaluating on rmse

import numpy as np
# Calculate RMSE
rmse = np.sqrt(np.mean((test - forecast)**2))

print(f"RMSE: {rmse}")
```

```
RMSE: 0.003938310176941583
```

- A lower RMSE value indicates a better fit of the model to the data. From our computed RMSE of 0.00393 is relatively low hence the model seems to be well performing.

6.3 Testing The Model's Performance

In [60]:
```
import statsmodels.api as sm

# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Plot ACF and PACF for each zip code
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data
    #plt.figure(figsize=(10, 6))
    # Initializing ARIMA model
    ARIMAmodel = sm.tsa.ARIMA(ts, order=(2,0,3))
    ARIMAmodel = ARIMAmodel.fit()
    pred = ARIMAmodel.get_prediction(start=pd.to_datetime('2015-02'), end=pd.
    pred_conf = pred.conf_int()
    pred_conf.head()
```

6.3.1 Results of Model 2 on Test Data

- The Root Mean Squared Error (RMSE) is a measure of the accuracy of a forecasting model. It quantifies the average magnitude of the differences between the predicted values and the actual observed values. In the context of the provided test set results:
- For Zip Code **11216**: The RMSE is 0.02. This indicates that, on average, the forecasted values differ from the actual values by approximately 0.02 units.
- For Zip Code **11222**: The RMSE is 0.01. This suggests that the forecasted values are, on average, closer to the actual values with an average difference of around 0.01 units.

- For Zip Code **15201**: The RMSE is 0.01. Similar to the previous case, the forecasted values are quite accurate with an average difference of approximately 0.01 units from the actual values.
- For Zip Code **94043**: The RMSE is 0.00. This indicates that the forecasted values are almost exactly matching the actual values, resulting in an RMSE close to zero.
- For Zip Code **94301**: The RMSE is 0.00. Similar to the previous case, the forecasted values for this zip code are very accurate with an RMSE close to zero.
- In summary, lower RMSE values imply that the forecasting model's predictions are closer to the actual observed values, indicating higher accuracy and reliability in forecasting for the given zip codes.

6.4 Forecasting For The Next 3 Years

In [61]:
```
# Plug the parameter values from our Auto ARIMA model into a new ARIMA model t
from statsmodels.tsa.arima.model import ARIMA
# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data
    # Plug the parameter values from our Auto ARIMA model into a new ARIMA mod
    ARIMA_MODEL = ARIMA(ts,
                        order=(2,0,3),
                        enforce_stationarity=False,
                        enforce_invertibility=False)
    # Fit the model and print results
    full_output = ARIMA_MODEL.fit()
    print(full_output.summary())
```

```
                               SARIMAX Results
==========================================================================
==
Dep. Variable:                     ret   No. Observations:                  1
59
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 569.3
34
Date:                Fri, 01 Sep 2023   AIC                          -1124.6
68
Time:                        00:44:56   BIC                          -1103.3
64
Sample:                    02-01-2005   HQIC                         -1116.0
15
                         - 04-01-2018
Covariance Type:                   opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
const          0.0159      0.005      3.276      0.001       0.006       0.0
25
ar.L1          0.9546      0.149      6.425      0.000       0.663       1.2
46
ar.L2         -0.0239      0.095     -0.252      0.801      -0.210       0.1
62
ma.L1          0.8063      0.132      6.118      0.000       0.548       1.0
65
ma.L2         -0.4598      0.190     -2.419      0.016      -0.832      -0.0
87
ma.L3         -0.8212      0.108     -7.581      0.000      -1.034      -0.6
09
sigma2      3.287e-05   2.76e-06     11.926      0.000    2.75e-05    3.83e-
05
==========================================================================
=======
Ljung-Box (L1) (Q):                   1.81   Jarque-Bera (JB):
68.68
Prob(Q):                              0.18   Prob(JB):
0.00
Heteroskedasticity (H):               4.40   Skew:
-0.25
Prob(H) (two-sided):                  0.00   Kurtosis:
6.22
==========================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                               SARIMAX Results
==========================================================================
==
Dep. Variable:                     ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 611.6
```

```
88
Date:                Fri, 01 Sep 2023   AIC                           -1209.3
76
Time:                        00:44:57   BIC                           -1188.1
17
Sample:                    03-01-2005   HQIC                          -1200.7
41
                         - 04-01-2018
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const      -6.195e-05      0.000     -0.253      0.800      -0.001       0.0
00
ar.L1          0.8829      0.193      4.575      0.000       0.505       1.2
61
ar.L2         -0.2788      0.174     -1.601      0.109      -0.620       0.0
62
ma.L1         -0.0405      0.207     -0.195      0.845      -0.447       0.3
66
ma.L2         -0.4962      0.095     -5.225      0.000      -0.682      -0.3
10
ma.L3         -0.2355      0.162     -1.452      0.147      -0.554       0.0
82
sigma2       2.06e-05   2.09e-06      9.846      0.000    1.65e-05    2.47e-
05
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.10   Jarque-Bera (JB):
45.50
Prob(Q):                              0.75   Prob(JB):
0.00
Heteroskedasticity (H):               6.98   Skew:
0.54
Prob(H) (two-sided):                  0.00   Kurtosis:
5.43
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                              SARIMAX Results
==============================================================================
==
Dep. Variable:                    ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 626.7
22
Date:                Fri, 01 Sep 2023   AIC                           -1239.4
44
Time:                        00:44:57   BIC                           -1218.1
86
Sample:                    03-01-2005   HQIC                          -1230.8
```

```
09
                         - 04-01-2018
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const       7.885e-05   4.91e-05      1.605      0.108   -1.74e-05       0.0
00
ar.L1          0.4564      0.281      1.623      0.105      -0.095       1.0
07
ar.L2         -0.0548      0.220     -0.249      0.803      -0.486       0.3
76
ma.L1          0.0669      0.299      0.223      0.823      -0.520       0.6
54
ma.L2         -0.6018      0.112     -5.370      0.000      -0.821      -0.3
82
ma.L3         -0.4042      0.207     -1.957      0.050      -0.809       0.0
01
sigma2      1.672e-05   2.08e-06      8.046      0.000    1.26e-05    2.08e-
05
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.89   Jarque-Bera (JB):
10.78
Prob(Q):                              0.35   Prob(JB):
0.00
Heteroskedasticity (H):               3.41   Skew:
0.53
Prob(H) (two-sided):                  0.00   Kurtosis:
3.75
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                              SARIMAX Results
==============================================================================
==
Dep. Variable:                    ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 700.6
28
Date:                Fri, 01 Sep 2023   AIC                          -1387.2
57
Time:                        00:44:57   BIC                          -1365.9
98
Sample:                    03-01-2005   HQIC                         -1378.6
22
                         - 04-01-2018
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
```

```
5]
------------------------------------------------------------------------------
--
const       5.554e-05      0.000      0.216      0.829      -0.000       0.0
01
ar.L1         -0.1637      0.131     -1.251      0.211      -0.420       0.0
93
ar.L2         -0.2357      0.124     -1.908      0.056      -0.478       0.0
06
ma.L1          1.0879      0.121      8.990      0.000       0.851       1.3
25
ma.L2          0.0378      0.216      0.175      0.861      -0.385       0.4
60
ma.L3         -0.5159      0.128     -4.015      0.000      -0.768      -0.2
64
sigma2      6.338e-06   5.86e-07     10.819      0.000    5.19e-06    7.49e-
06
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.35   Jarque-Bera (JB):
23.56
Prob(Q):                              0.55   Prob(JB):
0.00
Heteroskedasticity (H):               6.18   Skew:
-0.49
Prob(H) (two-sided):                  0.00   Kurtosis:
4.65
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

```
                               SARIMAX Results
==========================================================================
==
Dep. Variable:                     ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                694.9
19
Date:                Fri, 01 Sep 2023   AIC                           -1375.8
37
Time:                        00:44:58   BIC                           -1354.5
78
Sample:                    03-01-2005   HQIC                          -1367.2
02
                         - 04-01-2018
Covariance Type:                 opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
const       4.287e-05      0.000      0.197      0.844      -0.000       0.0
00
ar.L1          0.2145      0.160      1.336      0.181      -0.100       0.5
29
ar.L2          0.0933      0.106      0.883      0.377      -0.114       0.3
00
ma.L1          0.7071      0.132      5.354      0.000       0.448       0.9
66
ma.L2         -0.3950      0.177     -2.236      0.025      -0.741      -0.0
49
ma.L3         -0.6744      0.100     -6.724      0.000      -0.871      -0.4
78
sigma2       6.91e-06   6.72e-07     10.282      0.000    5.59e-06    8.23e-
06
==========================================================================
=======
Ljung-Box (L1) (Q):                   0.12   Jarque-Bera (JB):
113.75
Prob(Q):                              0.73   Prob(JB):
0.00
Heteroskedasticity (H):               3.10   Skew:
0.86
Prob(H) (two-sided):                  0.00   Kurtosis:
6.85
==========================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

In [62]:
```
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data
    # Getting a forecast for the next 36 months after the last recorded date
    forecast = full_output.get_forecast(36)
    future_prediction = forecast.conf_int()
    future_prediction['Price'] = forecast.predicted_mean
    future_prediction.columns = ['lower','upper','prediction']
    future_prediction.head()
```

In [63]:
```
from statsmodels.tsa.arima.model import ARIMA
# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Plug the parameter values from our Auto ARIMA model into a new ARIMA mo

    ARIMA_MODEL = ARIMA(ts,
                        order=(2,0,3),
                        enforce_stationarity=False,
                        enforce_invertibility=False)

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

    print(full_output.summary())
```

```
                               SARIMAX Results
==========================================================================
==
Dep. Variable:                     ret   No. Observations:                  1
59
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 569.3
34
Date:                Fri, 01 Sep 2023   AIC                          -1124.6
68
Time:                        00:44:58   BIC                          -1103.3
64
Sample:                    02-01-2005   HQIC                         -1116.0
15
                         - 04-01-2018
Covariance Type:                   opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
const          0.0159      0.005      3.276      0.001       0.006       0.0
25
ar.L1          0.9546      0.149      6.425      0.000       0.663       1.2
46
ar.L2         -0.0239      0.095     -0.252      0.801      -0.210       0.1
62
ma.L1          0.8063      0.132      6.118      0.000       0.548       1.0
65
ma.L2         -0.4598      0.190     -2.419      0.016      -0.832      -0.0
87
ma.L3         -0.8212      0.108     -7.581      0.000      -1.034      -0.6
09
sigma2      3.287e-05   2.76e-06     11.926      0.000    2.75e-05    3.83e-
05
==========================================================================
=======
Ljung-Box (L1) (Q):                   1.81   Jarque-Bera (JB):
68.68
Prob(Q):                              0.18   Prob(JB):
0.00
Heteroskedasticity (H):               4.40   Skew:
-0.25
Prob(H) (two-sided):                  0.00   Kurtosis:
6.22
==========================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                               SARIMAX Results
==========================================================================
==
Dep. Variable:                     ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 611.6
```

```
88
Date:                Fri, 01 Sep 2023   AIC                           -1209.3
76
Time:                        00:44:58   BIC                           -1188.1
17
Sample:                    03-01-2005   HQIC                          -1200.7
41
                         - 04-01-2018
Covariance Type:                 opg
================================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------------
--
const      -6.195e-05      0.000     -0.253      0.800      -0.001       0.0
00
ar.L1          0.8829      0.193      4.575      0.000       0.505       1.2
61
ar.L2         -0.2788      0.174     -1.601      0.109      -0.620       0.0
62
ma.L1         -0.0405      0.207     -0.195      0.845      -0.447       0.3
66
ma.L2         -0.4962      0.095     -5.225      0.000      -0.682      -0.3
10
ma.L3         -0.2355      0.162     -1.452      0.147      -0.554       0.0
82
sigma2       2.06e-05   2.09e-06      9.846      0.000    1.65e-05    2.47e-
05
================================================================================
=======
Ljung-Box (L1) (Q):                   0.10   Jarque-Bera (JB):
45.50
Prob(Q):                              0.75   Prob(JB):
0.00
Heteroskedasticity (H):               6.98   Skew:
0.54
Prob(H) (two-sided):                  0.00   Kurtosis:
5.43
================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                              SARIMAX Results
================================================================================
==
Dep. Variable:                    ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 626.7
22
Date:                Fri, 01 Sep 2023   AIC                           -1239.4
44
Time:                        00:44:58   BIC                           -1218.1
86
Sample:                    03-01-2005   HQIC                          -1230.8
```

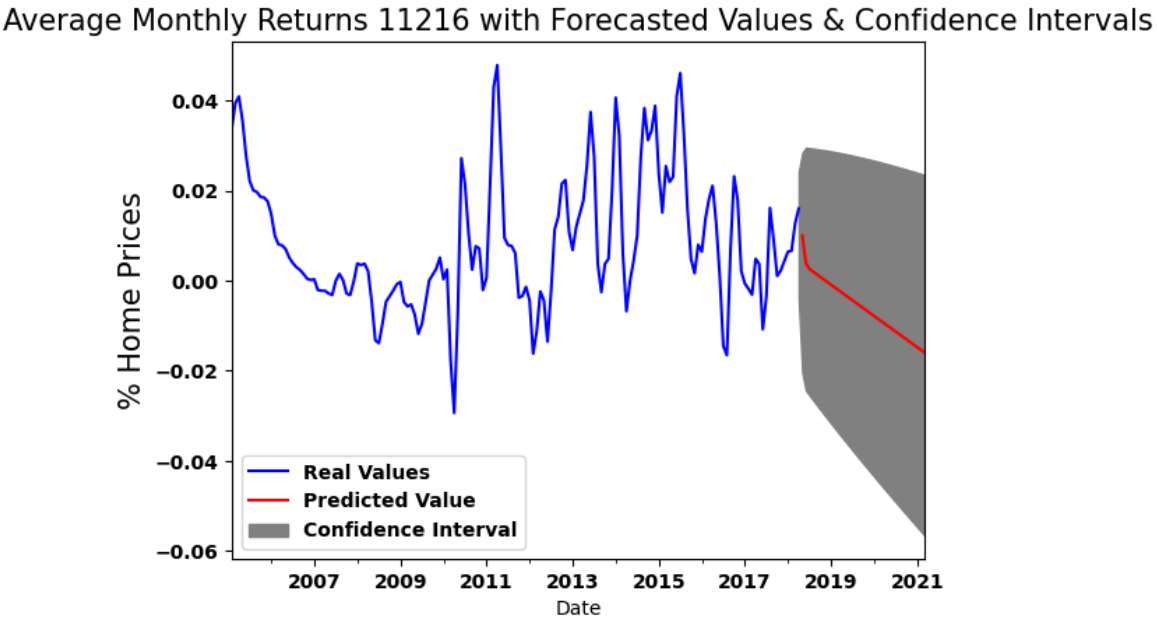
```
09
                         - 04-01-2018
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const       7.885e-05   4.91e-05      1.605      0.108   -1.74e-05       0.0
00
ar.L1          0.4564      0.281      1.623      0.105      -0.095       1.0
07
ar.L2         -0.0548      0.220     -0.249      0.803      -0.486       0.3
76
ma.L1          0.0669      0.299      0.223      0.823      -0.520       0.6
54
ma.L2         -0.6018      0.112     -5.370      0.000      -0.821      -0.3
82
ma.L3         -0.4042      0.207     -1.957      0.050      -0.809       0.0
01
sigma2      1.672e-05   2.08e-06      8.046      0.000    1.26e-05    2.08e-
05
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.89   Jarque-Bera (JB):
10.78
Prob(Q):                              0.35   Prob(JB):
0.00
Heteroskedasticity (H):               3.41   Skew:
0.53
Prob(H) (two-sided):                  0.00   Kurtosis:
3.75
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                              SARIMAX Results
==============================================================================
==
Dep. Variable:                    ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                 700.6
28
Date:                Fri, 01 Sep 2023   AIC                          -1387.2
57
Time:                        00:44:59   BIC                          -1365.9
98
Sample:                    03-01-2005   HQIC                         -1378.6
22
                         - 04-01-2018
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
```

```
5]
------------------------------------------------------------------------------
--
const       5.554e-05      0.000      0.216      0.829      -0.000       0.0
01
ar.L1         -0.1637      0.131     -1.251      0.211      -0.420       0.0
93
ar.L2         -0.2357      0.124     -1.908      0.056      -0.478       0.0
06
ma.L1          1.0879      0.121      8.990      0.000       0.851       1.3
25
ma.L2          0.0378      0.216      0.175      0.861      -0.385       0.4
60
ma.L3         -0.5159      0.128     -4.015      0.000      -0.768      -0.2
64
sigma2      6.338e-06   5.86e-07     10.819      0.000    5.19e-06    7.49e-
06
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.35   Jarque-Bera (JB):
23.56
Prob(Q):                              0.55   Prob(JB):
0.00
Heteroskedasticity (H):               6.18   Skew:
-0.49
Prob(H) (two-sided):                  0.00   Kurtosis:
4.65
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

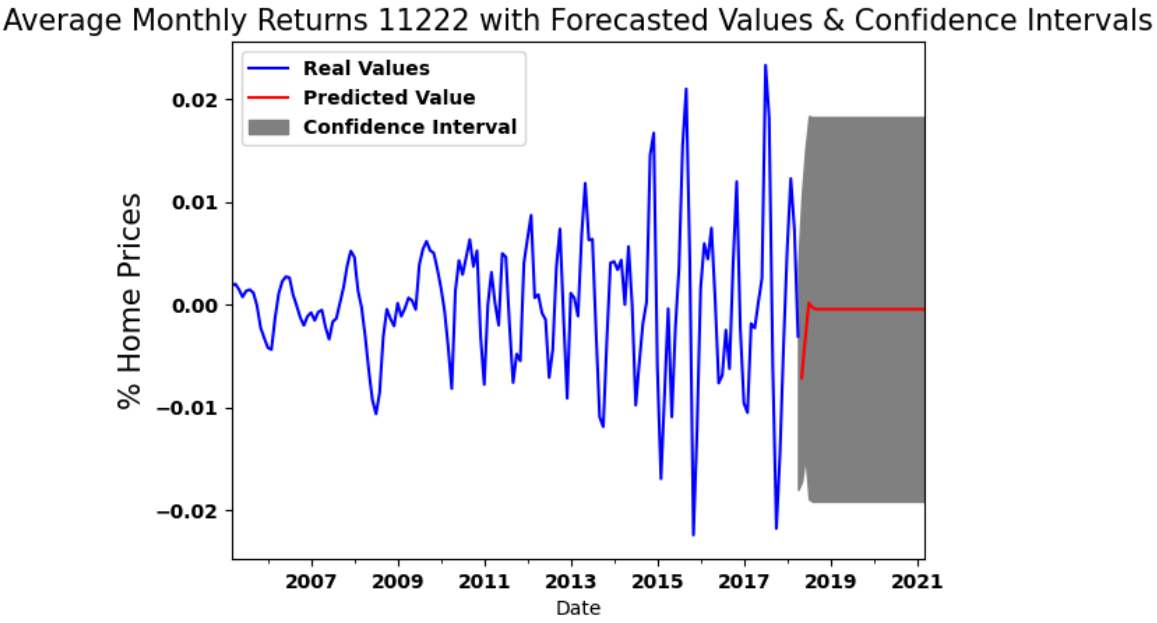
```
                               SARIMAX Results
==========================================================================
==
Dep. Variable:                     ret   No. Observations:                  1
58
Model:                 ARIMA(2, 0, 3)   Log Likelihood                694.9
19
Date:                Fri, 01 Sep 2023   AIC                           -1375.8
37
Time:                        00:44:59   BIC                           -1354.5
78
Sample:                    03-01-2005   HQIC                          -1367.2
02
                         - 04-01-2018
Covariance Type:                 opg
==========================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
--------------------------------------------------------------------------
--
const       4.287e-05      0.000      0.197      0.844      -0.000       0.0
00
ar.L1          0.2145      0.160      1.336      0.181      -0.100       0.5
29
ar.L2          0.0933      0.106      0.883      0.377      -0.114       0.3
00
ma.L1          0.7071      0.132      5.354      0.000       0.448       0.9
66
ma.L2         -0.3950      0.177     -2.236      0.025      -0.741      -0.0
49
ma.L3         -0.6744      0.100     -6.724      0.000      -0.871      -0.4
78
sigma2       6.91e-06   6.72e-07     10.282      0.000    5.59e-06    8.23e-
06
==========================================================================
=======
Ljung-Box (L1) (Q):                   0.12   Jarque-Bera (JB):
113.75
Prob(Q):                              0.73   Prob(JB):
0.00
Heteroskedasticity (H):               3.10   Skew:
0.86
Prob(H) (two-sided):                  0.00   Kurtosis:
6.85
==========================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

In [64]:
```
#plotting the Forecasting
  # List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']
  # Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data
        # Select data from 2015 onwards for training
    train_start = '2015-01-01'
    train = ts[train_start:]
        # Create and fit the ARIMA model using training data
    model = pm.auto_arima(train, trace=True, error_action='ignore', suppress_
      # Forecast for the next 36 months (2018 to 2021)
    forecast_steps = 36
    forecast, conf_int = model.predict(n_periods=forecast_steps, return_conf_
        # Create an index for the forecast period
    forecast_index = pd.date_range(start=train.index[-1], periods=forecast_st
        # Create a DataFrame for the forecasted values and confidence interva
    future_prediction = pd.DataFrame({'prediction': forecast, 'lower': conf_i
        # Plotting the forecasted values and confidence intervals
    fig, ax = plt.subplots()
    ts.plot(ax=ax, label='Real Values', c="blue")
    future_prediction['prediction'].plot(ax=ax, label='Predicted Value', c="r
    ax.fill_between(x=future_prediction.index, y1=future_prediction['lower'],
                    y2=future_prediction['upper'], color='gray',
                    label='Confidence Interval')
    ax.legend()
    plt.ylabel("% Home Prices", fontsize=15)
    plt.title(f'Average Monthly Returns {zipcode} with Forecasted Values & Co
    plt.show()
```

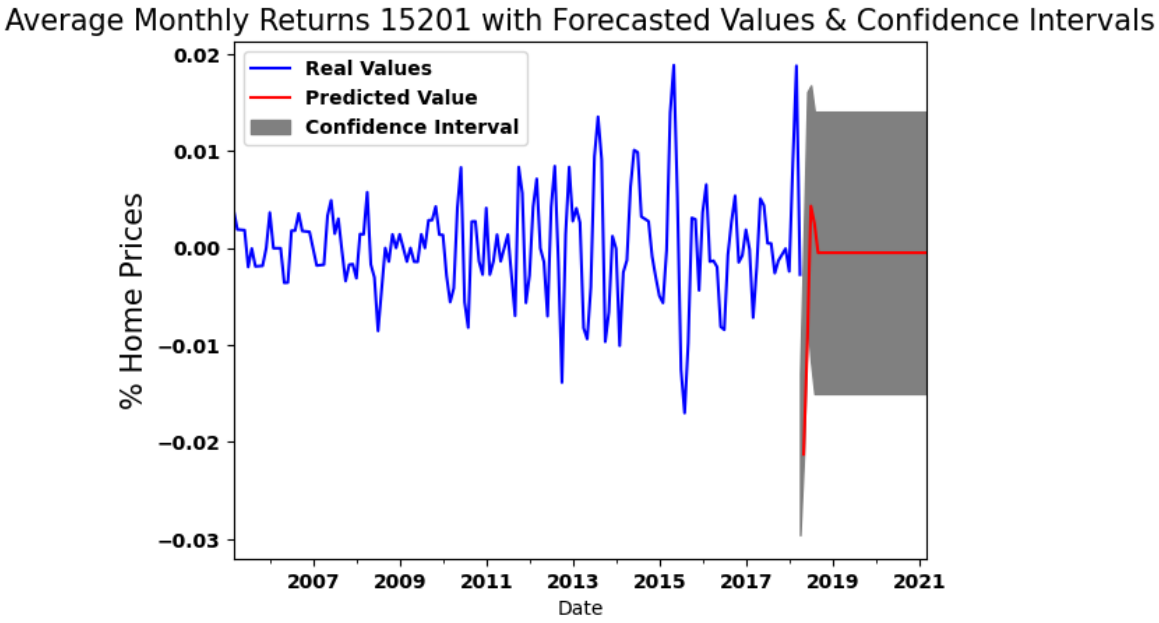
```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=-260.636, Time=0.33 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=-246.550, Time=0.02 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=-247.520, Time=0.03 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=-255.909, Time=0.14 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=-248.532, Time=0.01 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=-260.174, Time=0.12 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=-255.799, Time=0.17 sec
 ARIMA(3,1,2)(0,0,0)[0] intercept   : AIC=-259.040, Time=0.29 sec
 ARIMA(2,1,3)(0,0,0)[0] intercept   : AIC=-259.557, Time=0.21 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=-254.812, Time=0.17 sec
 ARIMA(1,1,3)(0,0,0)[0] intercept   : AIC=-261.016, Time=0.21 sec
 ARIMA(0,1,3)(0,0,0)[0] intercept   : AIC=-262.961, Time=0.11 sec
 ARIMA(0,1,2)(0,0,0)[0] intercept   : AIC=-257.917, Time=0.07 sec
 ARIMA(0,1,4)(0,0,0)[0] intercept   : AIC=-258.222, Time=0.13 sec
 ARIMA(1,1,4)(0,0,0)[0] intercept   : AIC=-258.854, Time=0.23 sec
 ARIMA(0,1,3)(0,0,0)[0]             : AIC=-261.531, Time=0.20 sec

Best model:  ARIMA(0,1,3)(0,0,0)[0] intercept
Total fit time: 2.462 seconds
```



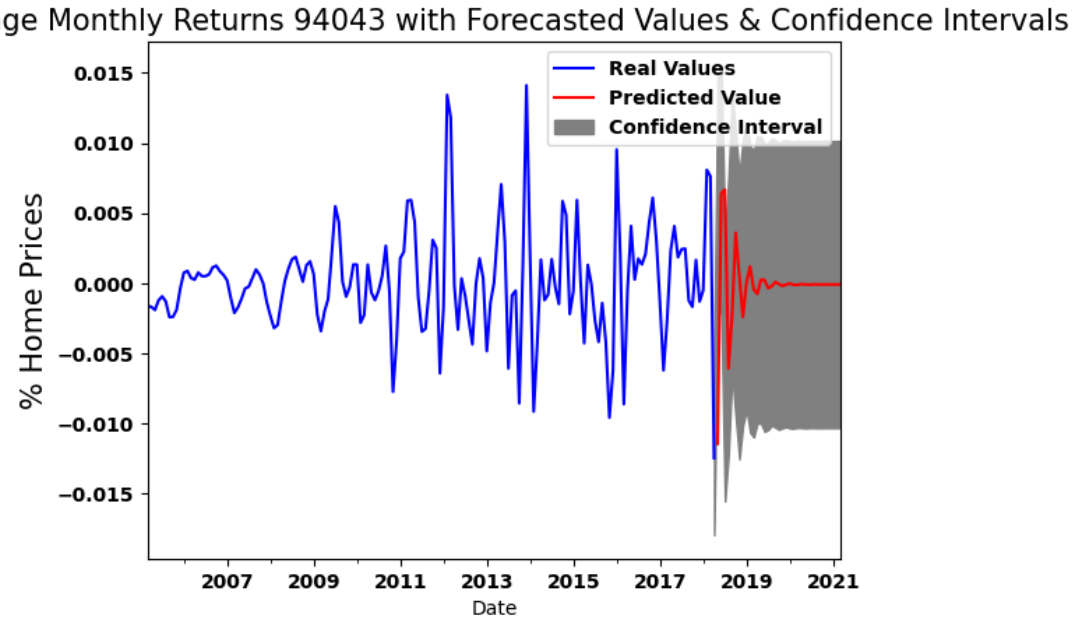

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=-280.223, Time=0.22 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=-247.429, Time=0.03 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=-256.628, Time=0.04 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=-266.261, Time=0.12 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=-249.224, Time=0.02 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=-268.542, Time=0.18 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=-272.991, Time=0.07 sec
 ARIMA(3,0,2)(0,0,0)[0] intercept   : AIC=-278.065, Time=0.22 sec
 ARIMA(2,0,3)(0,0,0)[0] intercept   : AIC=-274.957, Time=0.18 sec
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=-265.815, Time=0.15 sec
 ARIMA(1,0,3)(0,0,0)[0] intercept   : AIC=-283.324, Time=0.20 sec
 ARIMA(0,0,3)(0,0,0)[0] intercept   : AIC=inf, Time=0.15 sec
 ARIMA(1,0,4)(0,0,0)[0] intercept   : AIC=inf, Time=0.24 sec
 ARIMA(0,0,2)(0,0,0)[0] intercept   : AIC=-270.059, Time=0.11 sec
 ARIMA(0,0,4)(0,0,0)[0] intercept   : AIC=-284.067, Time=0.18 sec
 ARIMA(0,0,5)(0,0,0)[0] intercept   : AIC=-281.768, Time=0.19 sec
 ARIMA(1,0,5)(0,0,0)[0] intercept   : AIC=-279.110, Time=0.27 sec
 ARIMA(0,0,4)(0,0,0)[0]             : AIC=-283.655, Time=0.13 sec

Best model:  ARIMA(0,0,4)(0,0,0)[0] intercept
Total fit time: 2.720 seconds
```



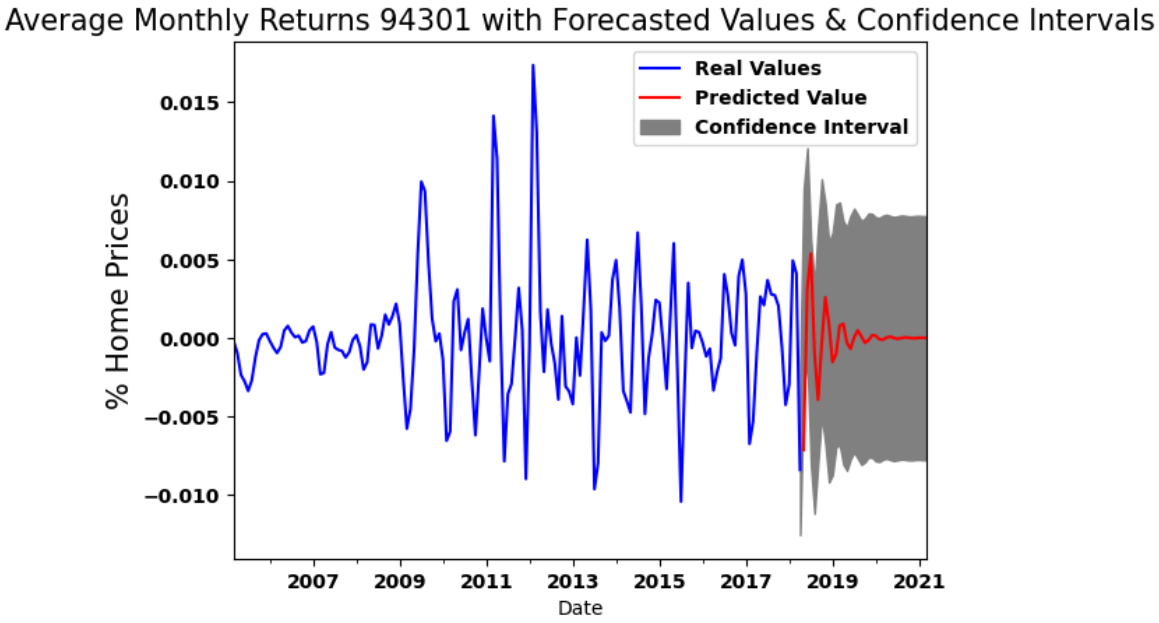

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=-305.569, Time=0.24 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=-278.513, Time=0.03 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=-285.162, Time=0.06 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=-297.959, Time=0.08 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=-280.503, Time=0.02 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=-296.999, Time=0.19 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=-301.353, Time=0.07 sec
 ARIMA(3,0,2)(0,0,0)[0] intercept   : AIC=-303.879, Time=0.19 sec
 ARIMA(2,0,3)(0,0,0)[0] intercept   : AIC=-306.405, Time=0.18 sec
 ARIMA(1,0,3)(0,0,0)[0] intercept   : AIC=-307.824, Time=0.21 sec
 ARIMA(0,0,3)(0,0,0)[0] intercept   : AIC=-306.860, Time=0.17 sec
 ARIMA(1,0,4)(0,0,0)[0] intercept   : AIC=-295.995, Time=0.09 sec
 ARIMA(0,0,2)(0,0,0)[0] intercept   : AIC=-298.688, Time=0.14 sec
 ARIMA(0,0,4)(0,0,0)[0] intercept   : AIC=-308.161, Time=0.19 sec
 ARIMA(0,0,5)(0,0,0)[0] intercept   : AIC=-305.741, Time=0.19 sec
 ARIMA(1,0,5)(0,0,0)[0] intercept   : AIC=-301.990, Time=0.27 sec
 ARIMA(0,0,4)(0,0,0)[0]             : AIC=-308.083, Time=0.16 sec

Best model:  ARIMA(0,0,4)(0,0,0)[0] intercept
Total fit time: 2.473 seconds
```




```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=-325.504, Time=0.17 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=-312.221, Time=0.04 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=-311.124, Time=0.06 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=-318.801, Time=0.07 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=-314.211, Time=0.02 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=-318.013, Time=0.16 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=-324.285, Time=0.17 sec
 ARIMA(3,0,2)(0,0,0)[0] intercept   : AIC=-325.881, Time=0.24 sec
 ARIMA(3,0,1)(0,0,0)[0] intercept   : AIC=-323.605, Time=0.23 sec
 ARIMA(4,0,2)(0,0,0)[0] intercept   : AIC=-320.976, Time=0.23 sec
 ARIMA(3,0,3)(0,0,0)[0] intercept   : AIC=-322.328, Time=0.34 sec
 ARIMA(2,0,3)(0,0,0)[0] intercept   : AIC=-325.370, Time=0.18 sec
 ARIMA(4,0,1)(0,0,0)[0] intercept   : AIC=-321.754, Time=0.16 sec
 ARIMA(4,0,3)(0,0,0)[0] intercept   : AIC=-318.910, Time=0.20 sec
 ARIMA(3,0,2)(0,0,0)[0]             : AIC=-324.989, Time=0.16 sec

Best model:  ARIMA(3,0,2)(0,0,0)[0] intercept
Total fit time: 2.444 seconds
```




```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=-344.254, Time=0.07 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=-331.256, Time=0.02 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=-331.710, Time=0.06 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=-339.479, Time=0.04 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=-333.255, Time=0.02 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=-338.210, Time=0.13 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=-345.991, Time=0.08 sec
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=-337.716, Time=0.10 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=-346.113, Time=0.08 sec
 ARIMA(3,0,0)(0,0,0)[0] intercept   : AIC=-346.180, Time=0.18 sec
 ARIMA(4,0,0)(0,0,0)[0] intercept   : AIC=-344.499, Time=0.13 sec
 ARIMA(3,0,1)(0,0,0)[0] intercept   : AIC=-343.191, Time=0.11 sec
 ARIMA(4,0,1)(0,0,0)[0] intercept   : AIC=-342.176, Time=0.17 sec
 ARIMA(3,0,0)(0,0,0)[0]             : AIC=-348.178, Time=0.06 sec
 ARIMA(2,0,0)(0,0,0)[0]             : AIC=-348.065, Time=0.03 sec
 ARIMA(4,0,0)(0,0,0)[0]             : AIC=-346.487, Time=0.10 sec
 ARIMA(3,0,1)(0,0,0)[0]             : AIC=-345.176, Time=0.06 sec
 ARIMA(2,0,1)(0,0,0)[0]             : AIC=-347.555, Time=0.14 sec
 ARIMA(4,0,1)(0,0,0)[0]             : AIC=-340.879, Time=0.04 sec

Best model:  ARIMA(3,0,0)(0,0,0)[0]
Total fit time: 1.623 seconds
```

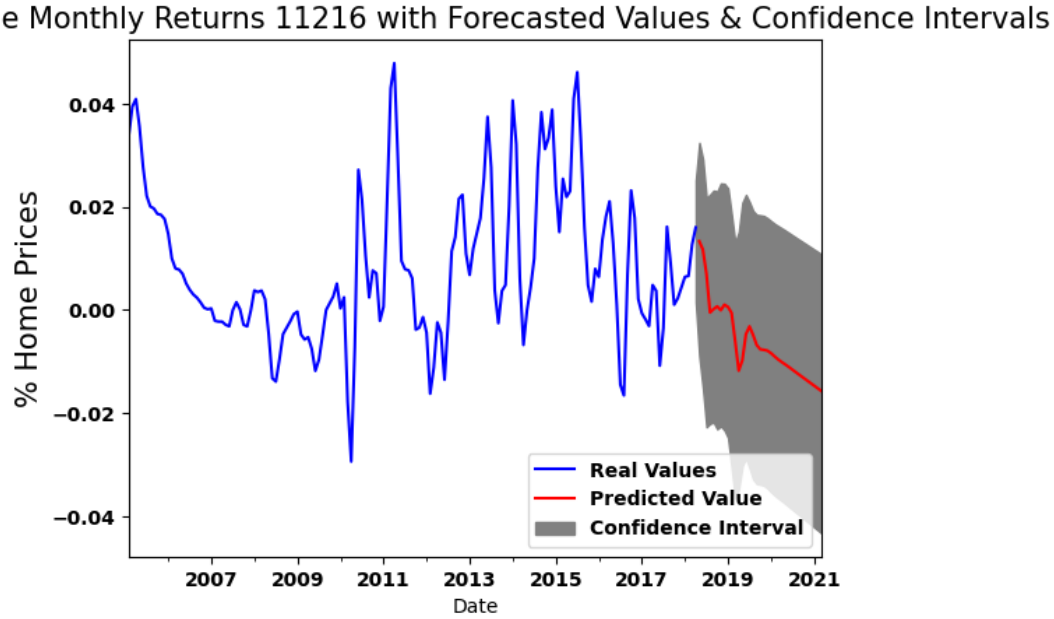


- The future_prediction DataFrame contains information about forecasted values for the time series. The 'prediction' column represents the predicted mean values for each forecasted date.
- The 'lower' and 'upper' columns provide the lower and upper bounds of the confidence intervals, which indicate the range of uncertainty around the forecasted values.
- These forecasted values are based on the ARIMA model's estimates and are intended to provide insights into potential future trends in the time series. From the oservation in the graph above it can be noted that can focast the future prices.

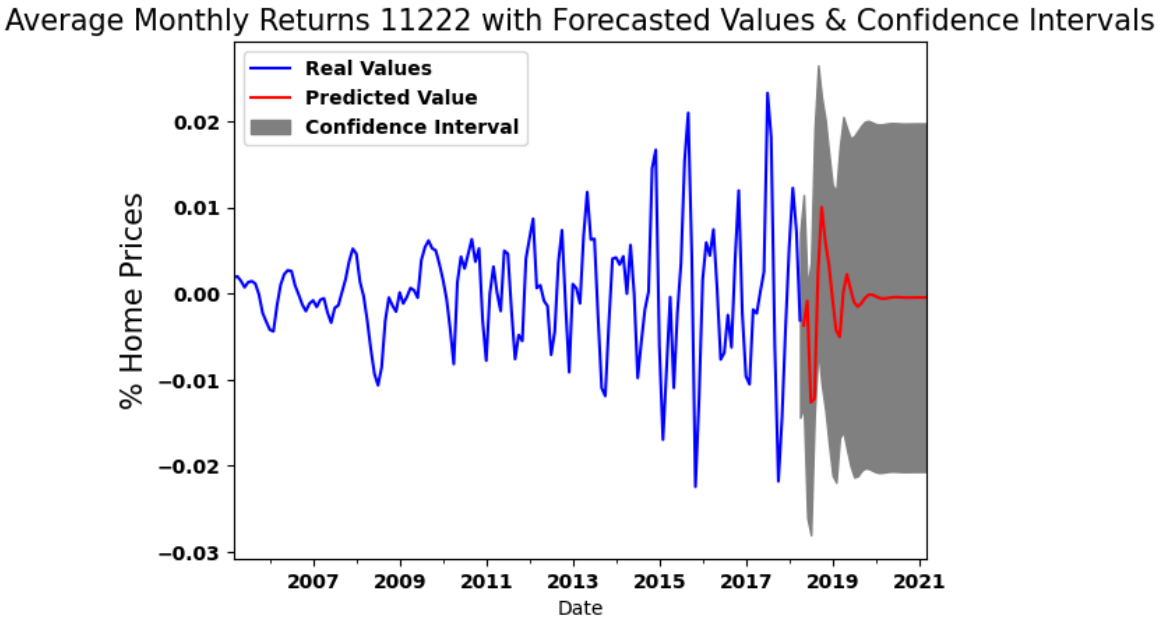
In [65]:
```
#Hyper parameter tunning.


# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Select data from 2015 onwards for training
    train_start = '2015-01-01'
    train = ts[train_start:]

    # Tune ARIMA model parameters using grid search
    model = pm.auto_arima(train,
                          trace=True,            # Print debugging informatio
                          error_action='ignore', # Skip models that fail
                          suppress_warnings=True, # Ignore warning messages
                          seasonal=True,         # Consider seasonal models
                          m=12,                  # Set the seasonal frequency
                          start_p=0,             # Minimum value of p
                          start_d=1,             # Minimum value of d
                          start_q=0,             # Minimum value of q
                          max_p=2,               # Maximum value of p
                          max_d=2,               # Maximum value of d
                          max_q=2,               # Maximum value of q
                          start_P=0,             # Minimum value of P
                          start_D=1,             # Minimum value of D
                          start_Q=0,             # Minimum value of Q
                          max_P=2,               # Maximum value of P
                          max_D=2,               # Maximum value of D
                          max_Q=2,               # Maximum value of Q
                          seasonal_test='ch')    # Use the Canova-Hansen test

    # Forecast for the next 36 months (2018 to 2021)
    forecast_steps = 36
    forecast, conf_int = model.predict(n_periods=forecast_steps, return_conf_

    # Create an index for the forecast period
    forecast_index = pd.date_range(start=train.index[-1], periods=forecast_st

    # Create a DataFrame for the forecasted values and confidence intervals
    future_prediction = pd.DataFrame({'prediction': forecast, 'lower': conf_i

    # Plotting the forecasted values and confidence intervals
    fig, ax = plt.subplots()
    ts.plot(ax=ax, label='Real Values', c="blue")

    future_prediction['prediction'].plot(ax=ax, label='Predicted Value', c="r

    ax.fill_between(x=future_prediction.index, y1=future_prediction['lower'],
                    y2=future_prediction['upper'], color='gray',
                    label='Confidence Interval')

    ax.legend()
    plt.ylabel("% Home Prices", fontsize=15)
```

```
    plt.title(f'Average Monthly Returns {zipcode} with Forecasted Values & Co
    plt.show()
```

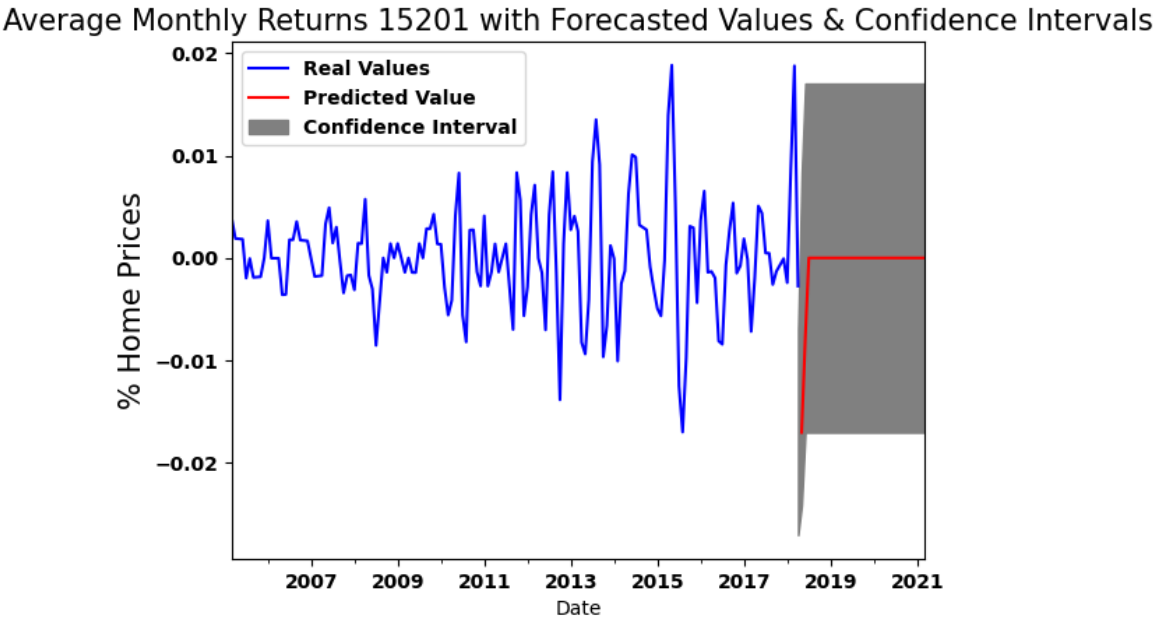
```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,0,0)[12] intercept   : AIC=-246.550, Time=0.03 sec
 ARIMA(1,1,0)(1,0,0)[12] intercept   : AIC=-251.496, Time=0.08 sec
 ARIMA(0,1,1)(0,0,1)[12] intercept   : AIC=-264.035, Time=0.20 sec
 ARIMA(0,1,0)(0,0,0)[12]             : AIC=-248.532, Time=0.01 sec
 ARIMA(0,1,1)(0,0,0)[12] intercept   : AIC=-255.909, Time=0.14 sec
 ARIMA(0,1,1)(1,0,1)[12] intercept   : AIC=-261.809, Time=0.33 sec
 ARIMA(0,1,1)(0,0,2)[12] intercept   : AIC=-262.009, Time=0.87 sec
 ARIMA(0,1,1)(1,0,0)[12] intercept   : AIC=-259.760, Time=0.17 sec
 ARIMA(0,1,1)(1,0,2)[12] intercept   : AIC=inf, Time=0.67 sec
 ARIMA(0,1,0)(0,0,1)[12] intercept   : AIC=-253.533, Time=0.23 sec
 ARIMA(1,1,1)(0,0,1)[12] intercept   : AIC=-262.581, Time=0.28 sec
 ARIMA(0,1,2)(0,0,1)[12] intercept   : AIC=-265.030, Time=0.15 sec
 ARIMA(0,1,2)(0,0,0)[12] intercept   : AIC=-257.917, Time=0.08 sec
 ARIMA(0,1,2)(1,0,1)[12] intercept   : AIC=-263.264, Time=0.50 sec
 ARIMA(0,1,2)(0,0,2)[12] intercept   : AIC=-263.192, Time=0.39 sec
 ARIMA(0,1,2)(1,0,0)[12] intercept   : AIC=-262.296, Time=0.11 sec
 ARIMA(0,1,2)(1,0,2)[12] intercept   : AIC=-260.132, Time=0.25 sec
 ARIMA(1,1,2)(0,0,1)[12] intercept   : AIC=-266.350, Time=0.47 sec
 ARIMA(1,1,2)(0,0,0)[12] intercept   : AIC=-260.174, Time=0.12 sec
 ARIMA(1,1,2)(1,0,1)[12] intercept   : AIC=-265.428, Time=0.52 sec
 ARIMA(1,1,2)(0,0,2)[12] intercept   : AIC=-262.142, Time=0.60 sec
 ARIMA(1,1,2)(1,0,0)[12] intercept   : AIC=-265.234, Time=0.43 sec
 ARIMA(1,1,2)(1,0,2)[12] intercept   : AIC=-264.116, Time=0.59 sec
 ARIMA(2,1,2)(0,0,1)[12] intercept   : AIC=-267.733, Time=0.40 sec
 ARIMA(2,1,2)(0,0,0)[12] intercept   : AIC=-260.636, Time=0.39 sec
 ARIMA(2,1,2)(1,0,1)[12] intercept   : AIC=-265.695, Time=0.34 sec
 ARIMA(2,1,2)(0,0,2)[12] intercept   : AIC=-264.472, Time=0.46 sec
 ARIMA(2,1,2)(1,0,0)[12] intercept   : AIC=-264.659, Time=0.50 sec
 ARIMA(2,1,2)(1,0,2)[12] intercept   : AIC=-264.680, Time=0.72 sec
 ARIMA(2,1,1)(0,0,1)[12] intercept   : AIC=-264.845, Time=0.50 sec
 ARIMA(2,1,2)(0,0,1)[12]             : AIC=-266.661, Time=0.33 sec

Best model:  ARIMA(2,1,2)(0,0,1)[12] intercept
Total fit time: 10.913 seconds
```



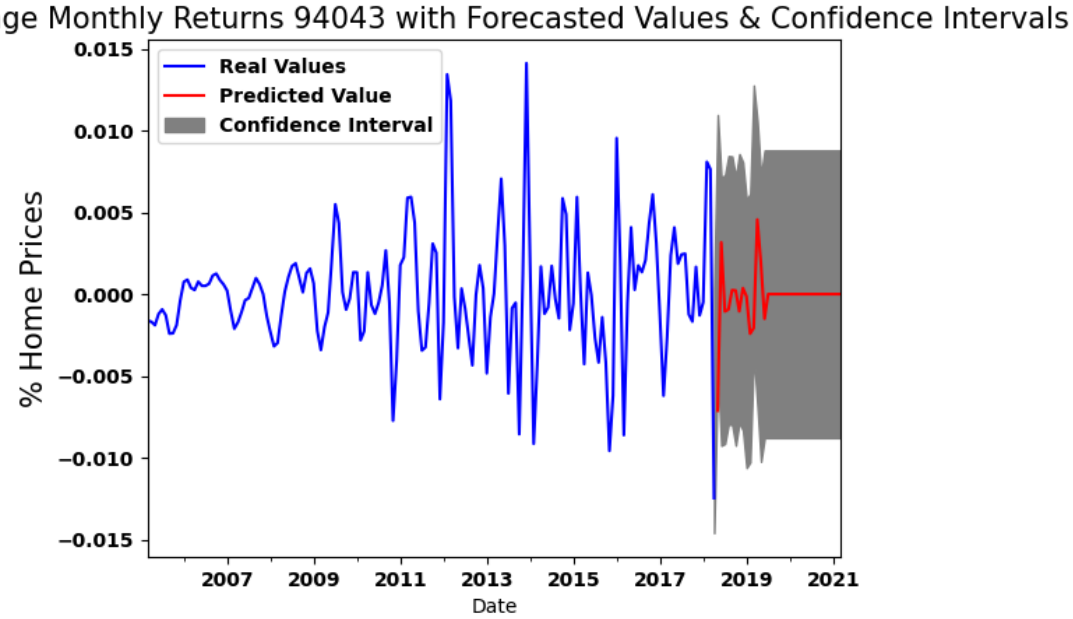

```
Performing stepwise search to minimize aic
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-247.429, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-260.824, Time=0.14 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-269.592, Time=0.12 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-249.224, Time=0.02 sec
 ARIMA(0,0,1)(0,0,0)[12] intercept   : AIC=-266.261, Time=0.11 sec
 ARIMA(0,0,1)(1,0,1)[12] intercept   : AIC=-268.150, Time=0.22 sec
 ARIMA(0,0,1)(0,0,2)[12] intercept   : AIC=-267.805, Time=0.28 sec
 ARIMA(0,0,1)(1,0,0)[12] intercept   : AIC=-270.067, Time=0.19 sec
 ARIMA(0,0,1)(2,0,0)[12] intercept   : AIC=-268.125, Time=0.36 sec
 ARIMA(0,0,1)(2,0,1)[12] intercept   : AIC=-266.189, Time=0.49 sec
 ARIMA(0,0,0)(1,0,0)[12] intercept   : AIC=-254.743, Time=0.12 sec
 ARIMA(1,0,1)(1,0,0)[12] intercept   : AIC=-268.653, Time=0.28 sec
 ARIMA(0,0,2)(1,0,0)[12] intercept   : AIC=-261.634, Time=0.10 sec
 ARIMA(1,0,2)(1,0,0)[12] intercept   : AIC=-272.799, Time=0.25 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-268.542, Time=0.19 sec
 ARIMA(1,0,2)(2,0,0)[12] intercept   : AIC=-265.006, Time=0.55 sec
 ARIMA(1,0,2)(1,0,1)[12] intercept   : AIC=-267.350, Time=0.36 sec
 ARIMA(1,0,2)(0,0,1)[12] intercept   : AIC=inf, Time=0.33 sec
 ARIMA(1,0,2)(2,0,1)[12] intercept   : AIC=-263.103, Time=0.53 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-279.726, Time=0.41 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-280.223, Time=0.15 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-281.465, Time=0.25 sec
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-277.684, Time=0.35 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-279.171, Time=0.42 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-276.711, Time=0.71 sec
 ARIMA(2,0,1)(0,0,1)[12] intercept   : AIC=-274.608, Time=0.23 sec
 ARIMA(1,0,1)(0,0,1)[12] intercept   : AIC=-268.754, Time=0.17 sec
 ARIMA(2,0,2)(0,0,1)[12]             : AIC=-280.287, Time=0.30 sec

Best model:  ARIMA(2,0,2)(0,0,1)[12] intercept
Total fit time: 7.679 seconds
```



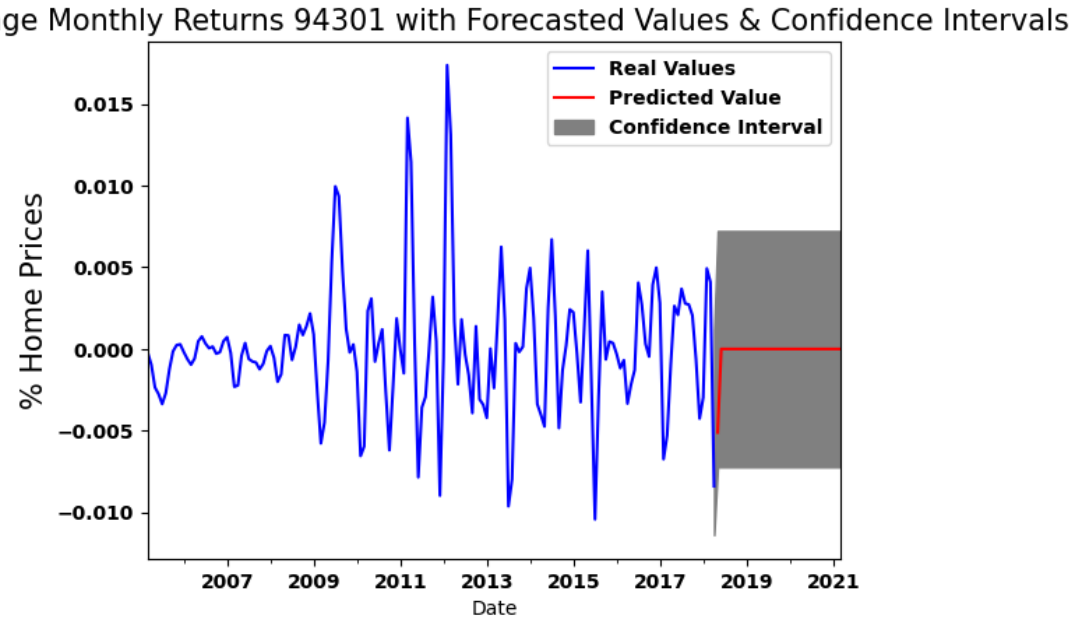

```
Performing stepwise search to minimize aic
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-278.513, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-283.598, Time=0.03 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-296.171, Time=0.12 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-280.503, Time=0.02 sec
 ARIMA(0,0,1)(0,0,0)[12] intercept   : AIC=-297.959, Time=0.07 sec
 ARIMA(0,0,1)(1,0,0)[12] intercept   : AIC=-296.176, Time=0.09 sec
 ARIMA(0,0,1)(1,0,1)[12] intercept   : AIC=-294.174, Time=0.32 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-296.540, Time=0.11 sec
 ARIMA(0,0,2)(0,0,0)[12] intercept   : AIC=-298.688, Time=0.16 sec
 ARIMA(0,0,2)(1,0,0)[12] intercept   : AIC=-289.061, Time=0.15 sec
 ARIMA(0,0,2)(0,0,1)[12] intercept   : AIC=-296.367, Time=0.19 sec
 ARIMA(0,0,2)(1,0,1)[12] intercept   : AIC=-294.662, Time=0.26 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-296.999, Time=0.15 sec
 ARIMA(0,0,2)(0,0,0)[12]             : AIC=-300.618, Time=0.12 sec
 ARIMA(0,0,2)(1,0,0)[12]             : AIC=-298.559, Time=0.17 sec
 ARIMA(0,0,2)(0,0,1)[12]             : AIC=-298.090, Time=0.27 sec
 ARIMA(0,0,2)(1,0,1)[12]             : AIC=-296.626, Time=0.29 sec
 ARIMA(0,0,1)(0,0,0)[12]             : AIC=-299.807, Time=0.03 sec
 ARIMA(1,0,2)(0,0,0)[12]             : AIC=-300.190, Time=0.13 sec
 ARIMA(1,0,1)(0,0,0)[12]             : AIC=-298.521, Time=0.08 sec

Best model:  ARIMA(0,0,2)(0,0,0)[12]
Total fit time: 2.800 seconds
```




```
Performing stepwise search to minimize aic
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-312.221, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-311.457, Time=0.23 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-318.177, Time=0.13 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-314.211, Time=0.02 sec
 ARIMA(0,0,1)(0,0,0)[12] intercept   : AIC=-318.801, Time=0.07 sec
 ARIMA(0,0,1)(1,0,0)[12] intercept   : AIC=-317.388, Time=0.20 sec
 ARIMA(0,0,1)(1,0,1)[12] intercept   : AIC=-316.280, Time=0.21 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-316.559, Time=0.09 sec
 ARIMA(0,0,2)(0,0,0)[12] intercept   : AIC=-319.146, Time=0.22 sec
 ARIMA(0,0,2)(1,0,0)[12] intercept   : AIC=-316.036, Time=0.17 sec
 ARIMA(0,0,2)(0,0,1)[12] intercept   : AIC=-319.489, Time=0.19 sec
 ARIMA(0,0,2)(1,0,1)[12] intercept   : AIC=-317.308, Time=0.28 sec
 ARIMA(0,0,2)(0,0,2)[12] intercept   : AIC=-315.572, Time=0.45 sec
 ARIMA(0,0,2)(1,0,2)[12] intercept   : AIC=-316.043, Time=0.57 sec
 ARIMA(1,0,2)(0,0,1)[12] intercept   : AIC=-318.076, Time=0.26 sec
 ARIMA(1,0,1)(0,0,1)[12] intercept   : AIC=-316.276, Time=0.14 sec
 ARIMA(0,0,2)(0,0,1)[12]             : AIC=-321.496, Time=0.18 sec
 ARIMA(0,0,2)(0,0,0)[12]             : AIC=-321.146, Time=0.10 sec
 ARIMA(0,0,2)(1,0,1)[12]             : AIC=-319.327, Time=0.21 sec
 ARIMA(0,0,2)(0,0,2)[12]             : AIC=-319.837, Time=0.27 sec
 ARIMA(0,0,2)(1,0,0)[12]             : AIC=-320.615, Time=0.20 sec
 ARIMA(0,0,2)(1,0,2)[12]             : AIC=-318.025, Time=0.37 sec
 ARIMA(0,0,1)(0,0,1)[12]             : AIC=-316.295, Time=0.03 sec
 ARIMA(1,0,2)(0,0,1)[12]             : AIC=-320.104, Time=0.23 sec
 ARIMA(1,0,1)(0,0,1)[12]             : AIC=-319.074, Time=0.17 sec

Best model:  ARIMA(0,0,2)(0,0,1)[12]
Total fit time: 5.049 seconds
```



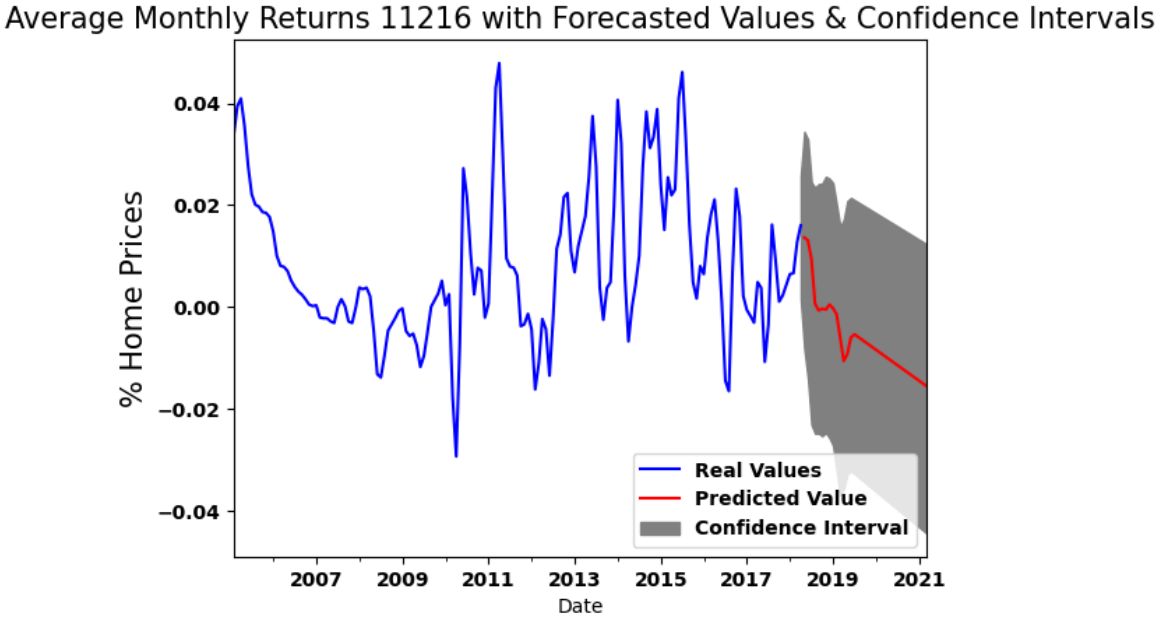

```
Performing stepwise search to minimize aic
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-331.256, Time=0.04 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-333.571, Time=0.12 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-338.773, Time=0.14 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-333.255, Time=0.02 sec
 ARIMA(0,0,1)(0,0,0)[12] intercept   : AIC=-339.479, Time=0.04 sec
 ARIMA(0,0,1)(1,0,0)[12] intercept   : AIC=-338.910, Time=0.12 sec
 ARIMA(0,0,1)(1,0,1)[12] intercept   : AIC=-336.452, Time=0.18 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-337.716, Time=0.09 sec
 ARIMA(0,0,2)(0,0,0)[12] intercept   : AIC=-337.649, Time=0.07 sec
 ARIMA(1,0,0)(0,0,0)[12] intercept   : AIC=-331.710, Time=0.05 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-338.210, Time=0.12 sec
 ARIMA(0,0,1)(0,0,0)[12]             : AIC=-341.451, Time=0.03 sec
 ARIMA(0,0,1)(1,0,0)[12]             : AIC=-340.871, Time=0.05 sec
 ARIMA(0,0,1)(0,0,1)[12]             : AIC=-340.729, Time=0.08 sec
 ARIMA(0,0,1)(1,0,1)[12]             : AIC=-338.445, Time=0.11 sec
 ARIMA(1,0,1)(0,0,0)[12]             : AIC=-339.321, Time=0.04 sec
 ARIMA(0,0,2)(0,0,0)[12]             : AIC=-339.618, Time=0.06 sec
 ARIMA(1,0,0)(0,0,0)[12]             : AIC=-333.701, Time=0.03 sec
 ARIMA(1,0,2)(0,0,0)[12]             : AIC=-333.647, Time=0.02 sec

Best model:  ARIMA(0,0,1)(0,0,0)[12]
Total fit time: 1.424 seconds
```



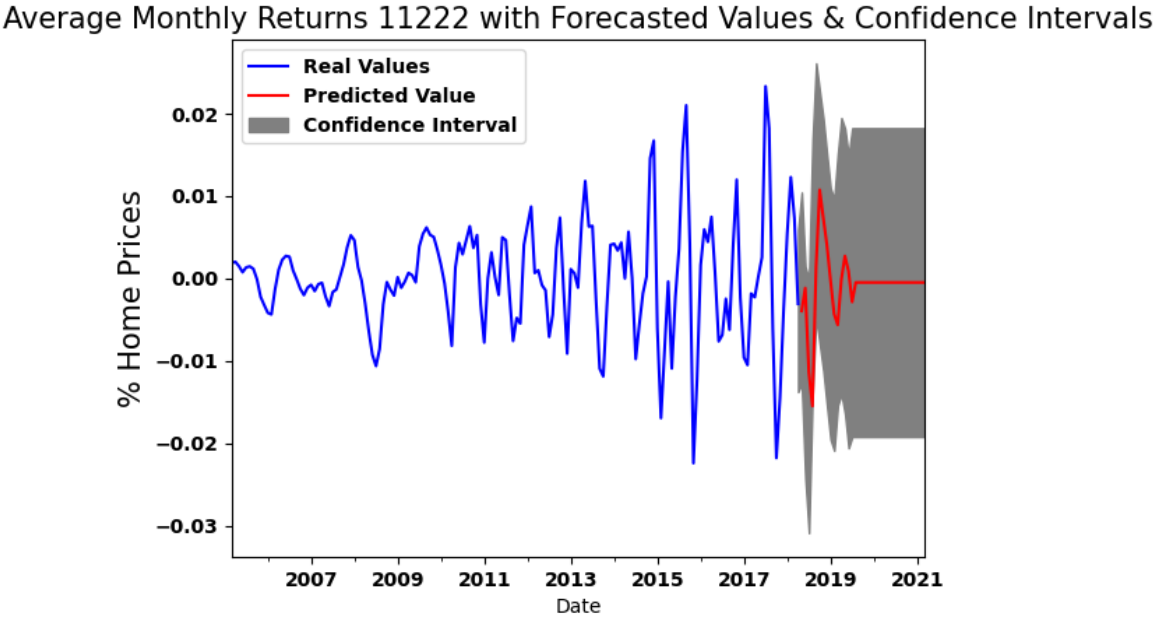
Grid search hyperparameter tunning

In [66]:
```
# Performing gridsearch parameter tunning


# List of zip code time series data
zip_codes = ['11216', '11222', '15201', '94043', '94301']

# Define parameter grid for grid search
param_grid = {
    'order': [(1, 1, 1), (1, 1, 2), (2, 1, 1), (2, 1, 2)],  # p, d, q values
    'seasonal_order': [(0, 1, 1, 12), (0, 1, 2, 12), (1, 1, 1, 12), (1, 1, 2,
}

# Loop through each zip code in the list
for zipcode in zip_codes:
    ts = globals()['ts_' + zipcode]  # Get the time series data

    # Select data from 2015 onwards for training
    train_start = '2015-01-01'
    train = ts[train_start:]

    # Perform grid search for best parameters using auto_arima
    best_model = pm.auto_arima(train,
                               seasonal=True,
                               m=12,  # Monthly seasonal pattern
                               stepwise=True,
                               suppress_warnings=True,
                               error_action='ignore',
                               trace=True)

    # Forecast for the next 36 months (2018 to 2021) using the best model
    forecast_steps = 36
    forecast, conf_int = best_model.predict(n_periods=forecast_steps, return_

    # Create an index for the forecast period
    forecast_index = pd.date_range(start=train.index[-1], periods=forecast_st

    # Create a DataFrame for the forecasted values and confidence intervals
    future_prediction = pd.DataFrame({'prediction': forecast, 'lower': conf_i

    # Plotting the forecasted values and confidence intervals
    fig, ax = plt.subplots()
    ts.plot(ax=ax, label='Real Values', c="blue")

    future_prediction['prediction'].plot(ax=ax, label='Predicted Value', c="r

    ax.fill_between(x=future_prediction.index, y1=future_prediction['lower'],
                    y2=future_prediction['upper'], color='gray',
                    label='Confidence Interval')

    ax.legend()
    plt.ylabel("% Home Prices", fontsize=15)
    plt.title(f'Average Monthly Returns {zipcode} with Forecasted Values & Co
    plt.show()
```

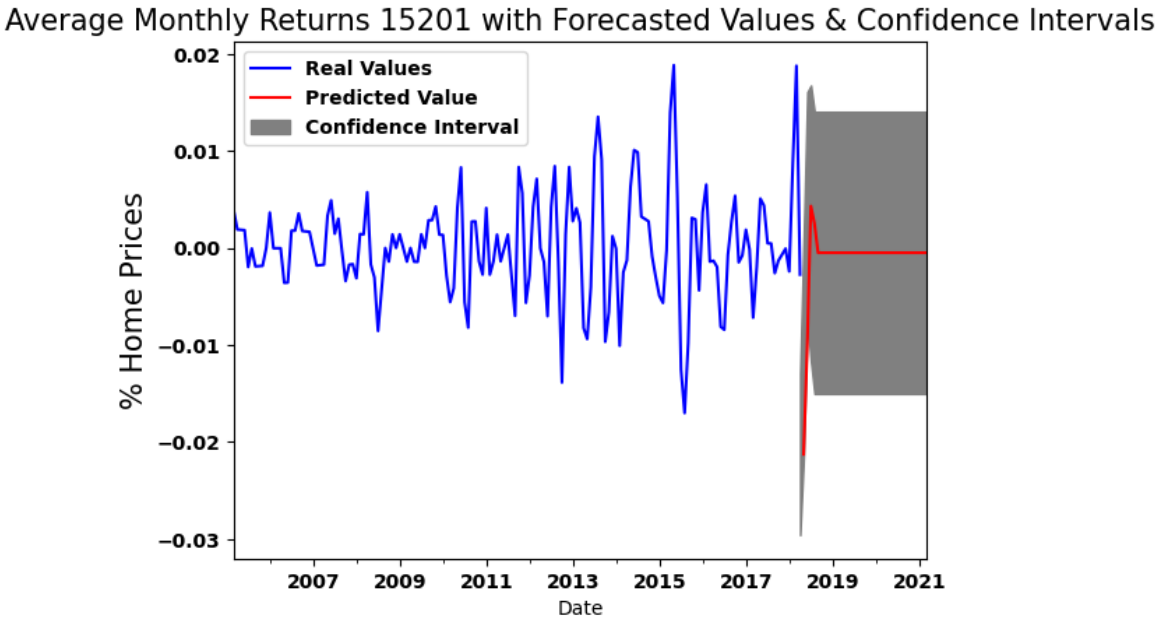
```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,0,1)[12] intercept   : AIC=-265.695, Time=0.27 sec
 ARIMA(0,1,0)(0,0,0)[12] intercept   : AIC=-246.550, Time=0.02 sec
 ARIMA(1,1,0)(1,0,0)[12] intercept   : AIC=-251.496, Time=0.07 sec
 ARIMA(0,1,1)(0,0,1)[12] intercept   : AIC=-264.035, Time=0.26 sec
 ARIMA(0,1,0)(0,0,0)[12]             : AIC=-248.532, Time=0.02 sec
 ARIMA(2,1,2)(0,0,1)[12] intercept   : AIC=-267.733, Time=0.27 sec
 ARIMA(2,1,2)(0,0,0)[12] intercept   : AIC=-260.636, Time=0.28 sec
 ARIMA(2,1,2)(0,0,2)[12] intercept   : AIC=-264.472, Time=0.41 sec
 ARIMA(2,1,2)(1,0,0)[12] intercept   : AIC=-264.659, Time=0.43 sec
 ARIMA(2,1,2)(1,0,2)[12] intercept   : AIC=-264.680, Time=0.57 sec
 ARIMA(1,1,2)(0,0,1)[12] intercept   : AIC=-266.350, Time=0.38 sec
 ARIMA(2,1,1)(0,0,1)[12] intercept   : AIC=-264.845, Time=0.37 sec
 ARIMA(3,1,2)(0,0,1)[12] intercept   : AIC=-266.148, Time=0.28 sec
 ARIMA(2,1,3)(0,0,1)[12] intercept   : AIC=-267.142, Time=0.36 sec
 ARIMA(1,1,1)(0,0,1)[12] intercept   : AIC=-262.581, Time=0.20 sec
 ARIMA(1,1,3)(0,0,1)[12] intercept   : AIC=-268.039, Time=0.38 sec
 ARIMA(1,1,3)(0,0,0)[12] intercept   : AIC=-261.016, Time=0.21 sec
 ARIMA(1,1,3)(1,0,1)[12] intercept   : AIC=-262.261, Time=0.21 sec
 ARIMA(1,1,3)(0,0,2)[12] intercept   : AIC=-262.519, Time=0.57 sec
 ARIMA(1,1,3)(1,0,0)[12] intercept   : AIC=-264.564, Time=0.58 sec
 ARIMA(1,1,3)(1,0,2)[12] intercept   : AIC=-261.253, Time=0.46 sec
 ARIMA(0,1,3)(0,0,1)[12] intercept   : AIC=-269.123, Time=0.39 sec
 ARIMA(0,1,3)(0,0,0)[12] intercept   : AIC=-262.961, Time=0.12 sec
 ARIMA(0,1,3)(1,0,1)[12] intercept   : AIC=-268.008, Time=0.42 sec
 ARIMA(0,1,3)(0,0,2)[12] intercept   : AIC=-268.840, Time=0.58 sec
 ARIMA(0,1,3)(1,0,0)[12] intercept   : AIC=-266.656, Time=0.23 sec
 ARIMA(0,1,3)(1,0,2)[12] intercept   : AIC=-266.439, Time=0.73 sec
 ARIMA(0,1,2)(0,0,1)[12] intercept   : AIC=-265.030, Time=0.17 sec
 ARIMA(0,1,4)(0,0,1)[12] intercept   : AIC=-267.429, Time=0.38 sec
 ARIMA(1,1,4)(0,0,1)[12] intercept   : AIC=-264.861, Time=0.48 sec
 ARIMA(0,1,3)(0,0,1)[12]             : AIC=inf, Time=0.25 sec

Best model:  ARIMA(0,1,3)(0,0,1)[12] intercept
Total fit time: 10.372 seconds
```



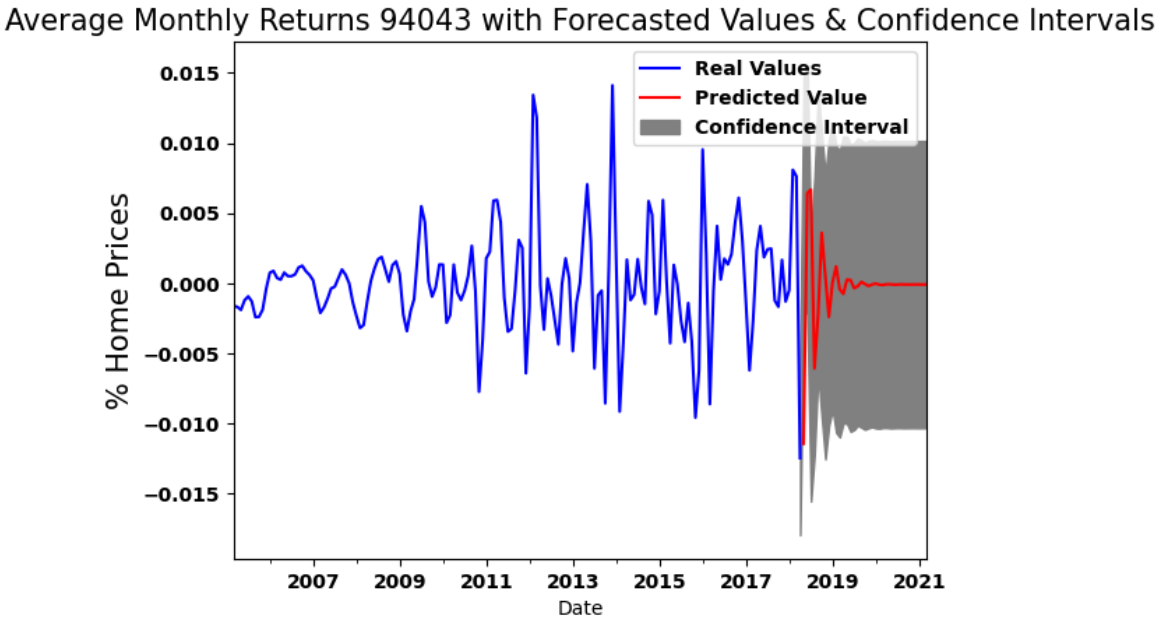

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-277.684, Time=0.35 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-247.429, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-260.824, Time=0.10 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-269.592, Time=0.10 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-249.224, Time=0.02 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-281.465, Time=0.24 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-280.223, Time=0.15 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=-279.171, Time=0.44 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-279.726, Time=0.40 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=-276.711, Time=0.66 sec
 ARIMA(1,0,2)(0,0,1)[12] intercept   : AIC=inf, Time=0.34 sec
 ARIMA(2,0,1)(0,0,1)[12] intercept   : AIC=-274.608, Time=0.19 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-278.311, Time=0.30 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-280.958, Time=0.28 sec
 ARIMA(1,0,1)(0,0,1)[12] intercept   : AIC=-268.754, Time=0.16 sec
 ARIMA(1,0,3)(0,0,1)[12] intercept   : AIC=-285.606, Time=0.28 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-283.324, Time=0.14 sec
 ARIMA(1,0,3)(1,0,1)[12] intercept   : AIC=-282.672, Time=0.33 sec
 ARIMA(1,0,3)(0,0,2)[12] intercept   : AIC=-283.254, Time=0.48 sec
 ARIMA(1,0,3)(1,0,0)[12] intercept   : AIC=-284.717, Time=0.32 sec
 ARIMA(1,0,3)(1,0,2)[12] intercept   : AIC=-281.510, Time=0.53 sec
 ARIMA(0,0,3)(0,0,1)[12] intercept   : AIC=-286.146, Time=0.25 sec
 ARIMA(0,0,3)(0,0,0)[12] intercept   : AIC=inf, Time=0.15 sec
 ARIMA(0,0,3)(1,0,1)[12] intercept   : AIC=inf, Time=0.30 sec
 ARIMA(0,0,3)(0,0,2)[12] intercept   : AIC=-283.775, Time=0.38 sec
 ARIMA(0,0,3)(1,0,0)[12] intercept   : AIC=inf, Time=0.32 sec
 ARIMA(0,0,3)(1,0,2)[12] intercept   : AIC=-282.727, Time=0.59 sec
 ARIMA(0,0,2)(0,0,1)[12] intercept   : AIC=inf, Time=0.36 sec
 ARIMA(0,0,4)(0,0,1)[12] intercept   : AIC=-284.694, Time=0.29 sec
 ARIMA(1,0,4)(0,0,1)[12] intercept   : AIC=-279.865, Time=0.31 sec
 ARIMA(0,0,3)(0,0,1)[12]             : AIC=inf, Time=0.19 sec

Best model:  ARIMA(0,0,3)(0,0,1)[12] intercept
Total fit time: 9.009 seconds
```



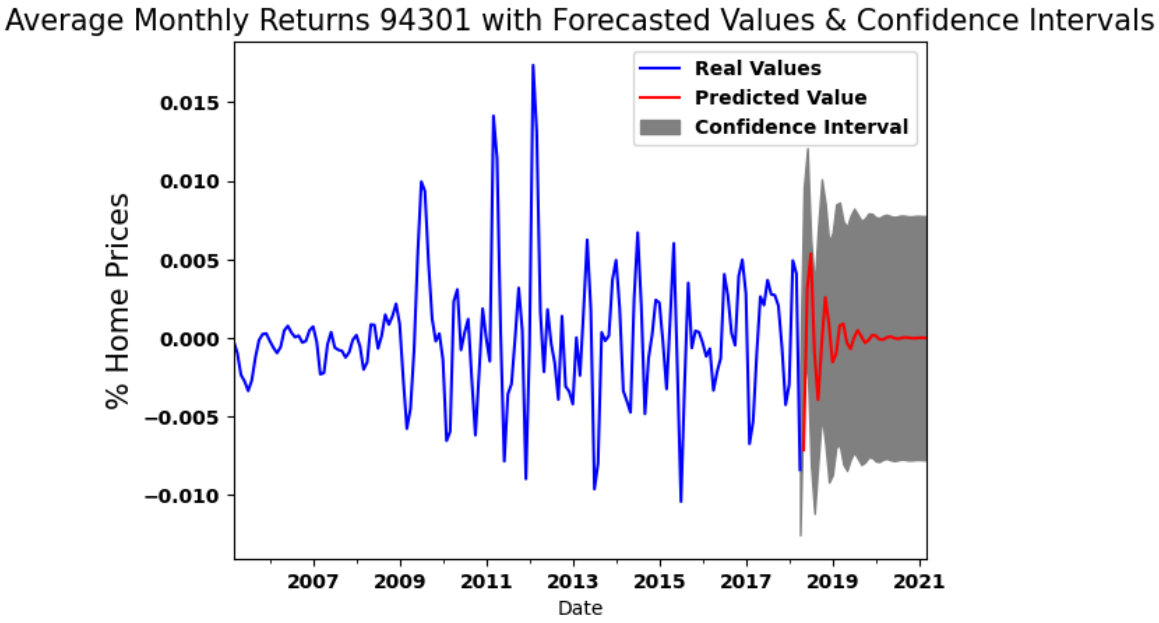

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-301.507, Time=0.31 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-278.513, Time=0.02 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-283.598, Time=0.03 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-296.171, Time=0.11 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-280.503, Time=0.02 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-303.557, Time=0.21 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-305.569, Time=0.17 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-303.836, Time=0.22 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-296.999, Time=0.13 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-301.353, Time=0.06 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-303.879, Time=0.17 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-306.405, Time=0.16 sec
 ARIMA(2,0,3)(1,0,0)[12] intercept   : AIC=-304.511, Time=0.31 sec
 ARIMA(2,0,3)(0,0,1)[12] intercept   : AIC=-304.418, Time=0.27 sec
 ARIMA(2,0,3)(1,0,1)[12] intercept   : AIC=-302.942, Time=0.33 sec
 ARIMA(1,0,3)(0,0,0)[12] intercept   : AIC=-307.824, Time=0.16 sec
 ARIMA(1,0,3)(1,0,0)[12] intercept   : AIC=-304.945, Time=0.18 sec
 ARIMA(1,0,3)(0,0,1)[12] intercept   : AIC=-305.137, Time=0.14 sec
 ARIMA(1,0,3)(1,0,1)[12] intercept   : AIC=-302.946, Time=0.22 sec
 ARIMA(0,0,3)(0,0,0)[12] intercept   : AIC=-306.860, Time=0.15 sec
 ARIMA(1,0,4)(0,0,0)[12] intercept   : AIC=-295.995, Time=0.09 sec
 ARIMA(0,0,2)(0,0,0)[12] intercept   : AIC=-298.688, Time=0.10 sec
 ARIMA(0,0,4)(0,0,0)[12] intercept   : AIC=-308.161, Time=0.14 sec
 ARIMA(0,0,4)(1,0,0)[12] intercept   : AIC=-306.371, Time=0.23 sec
 ARIMA(0,0,4)(0,0,1)[12] intercept   : AIC=-306.407, Time=0.25 sec
 ARIMA(0,0,4)(1,0,1)[12] intercept   : AIC=-304.020, Time=0.37 sec
 ARIMA(0,0,5)(0,0,0)[12] intercept   : AIC=-305.741, Time=0.18 sec
 ARIMA(1,0,5)(0,0,0)[12] intercept   : AIC=-301.990, Time=0.23 sec
 ARIMA(0,0,4)(0,0,0)[12]             : AIC=-308.083, Time=0.12 sec

Best model:  ARIMA(0,0,4)(0,0,0)[12] intercept
Total fit time: 5.109 seconds
```




```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-322.014, Time=0.40 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-312.221, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-311.457, Time=0.20 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-318.177, Time=0.11 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-314.211, Time=0.02 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-324.125, Time=0.26 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-325.504, Time=0.13 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-323.823, Time=0.27 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-318.013, Time=0.14 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-324.285, Time=0.17 sec
 ARIMA(3,0,2)(0,0,0)[12] intercept   : AIC=-325.881, Time=0.18 sec
 ARIMA(3,0,2)(1,0,0)[12] intercept   : AIC=-322.517, Time=0.29 sec
 ARIMA(3,0,2)(0,0,1)[12] intercept   : AIC=-322.591, Time=0.27 sec
 ARIMA(3,0,2)(1,0,1)[12] intercept   : AIC=-320.049, Time=0.20 sec
 ARIMA(3,0,1)(0,0,0)[12] intercept   : AIC=-323.605, Time=0.17 sec
 ARIMA(4,0,2)(0,0,0)[12] intercept   : AIC=-320.976, Time=0.20 sec
 ARIMA(3,0,3)(0,0,0)[12] intercept   : AIC=-322.328, Time=0.25 sec
 ARIMA(2,0,3)(0,0,0)[12] intercept   : AIC=-325.370, Time=0.16 sec
 ARIMA(4,0,1)(0,0,0)[12] intercept   : AIC=-321.754, Time=0.14 sec
 ARIMA(4,0,3)(0,0,0)[12] intercept   : AIC=-318.910, Time=0.13 sec
 ARIMA(3,0,2)(0,0,0)[12]             : AIC=-324.989, Time=0.11 sec

Best model:  ARIMA(3,0,2)(0,0,0)[12] intercept
Total fit time: 3.845 seconds
```


Average Monthly Returns 94043 with Forecasted Values & Confidence Intervals
Real Values
Predicted Value
Confidence Interval
% Home Prices
0.015
0.010
0.005
0.000
−0.005
−0.010
−0.015
2007
2009
2011
2013
2015
2017
2019
2021
Date

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=-339.163, Time=0.31 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=-331.256, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=-333.571, Time=0.10 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=-338.773, Time=0.14 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=-333.255, Time=0.02 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=-342.254, Time=0.10 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=-344.254, Time=0.06 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=-343.514, Time=0.26 sec
 ARIMA(1,0,2)(0,0,0)[12] intercept   : AIC=-338.210, Time=0.12 sec
 ARIMA(2,0,1)(0,0,0)[12] intercept   : AIC=-345.991, Time=0.08 sec
 ARIMA(2,0,1)(1,0,0)[12] intercept   : AIC=-344.327, Time=0.28 sec
 ARIMA(2,0,1)(0,0,1)[12] intercept   : AIC=-344.299, Time=0.19 sec
 ARIMA(2,0,1)(1,0,1)[12] intercept   : AIC=-342.131, Time=0.28 sec
 ARIMA(1,0,1)(0,0,0)[12] intercept   : AIC=-337.716, Time=0.10 sec
 ARIMA(2,0,0)(0,0,0)[12] intercept   : AIC=-346.113, Time=0.07 sec
 ARIMA(2,0,0)(1,0,0)[12] intercept   : AIC=-340.523, Time=0.09 sec
 ARIMA(2,0,0)(0,0,1)[12] intercept   : AIC=-344.299, Time=0.23 sec
 ARIMA(2,0,0)(1,0,1)[12] intercept   : AIC=-342.220, Time=0.28 sec
 ARIMA(1,0,0)(0,0,0)[12] intercept   : AIC=-331.710, Time=0.06 sec
 ARIMA(3,0,0)(0,0,0)[12] intercept   : AIC=-346.180, Time=0.06 sec
 ARIMA(3,0,0)(1,0,0)[12] intercept   : AIC=-339.905, Time=0.11 sec
 ARIMA(3,0,0)(0,0,1)[12] intercept   : AIC=-344.578, Time=0.22 sec
 ARIMA(3,0,0)(1,0,1)[12] intercept   : AIC=-342.139, Time=0.31 sec
 ARIMA(4,0,0)(0,0,0)[12] intercept   : AIC=-344.499, Time=0.09 sec
 ARIMA(3,0,1)(0,0,0)[12] intercept   : AIC=-343.191, Time=0.07 sec
 ARIMA(4,0,1)(0,0,0)[12] intercept   : AIC=-342.176, Time=0.15 sec
 ARIMA(3,0,0)(0,0,0)[12]             : AIC=-348.178, Time=0.04 sec
 ARIMA(3,0,0)(1,0,0)[12]             : AIC=-341.863, Time=0.08 sec
 ARIMA(3,0,0)(0,0,1)[12]             : AIC=-346.523, Time=0.10 sec
 ARIMA(3,0,0)(1,0,1)[12]             : AIC=-344.263, Time=0.25 sec
 ARIMA(2,0,0)(0,0,0)[12]             : AIC=-348.065, Time=0.03 sec
 ARIMA(4,0,0)(0,0,0)[12]             : AIC=-346.487, Time=0.08 sec
 ARIMA(3,0,1)(0,0,0)[12]             : AIC=-345.176, Time=0.04 sec
 ARIMA(2,0,1)(0,0,0)[12]             : AIC=-347.555, Time=0.09 sec
 ARIMA(4,0,1)(0,0,0)[12]             : AIC=-340.879, Time=0.03 sec

Best model:  ARIMA(3,0,0)(0,0,0)[12]
Total fit time: 4.553 seconds
```


Average Monthly Returns 94301 with Forecasted Values & Confidence Intervals
Real Values
Predicted Value
Confidence Interval
% Home Prices
0.015
0.010
0.005
0.000
−0.005
−0.010
2007
2009
2011
2013
2015
2017
2019
2021
Date

6.5 Forecasting for Every Zipcode

In [67]:
```
zip_predictions = {}

# Creating a for loop to forecast for every zipcode
for i in range(len(df_ts)):

    # selecting every individual series
    series = df_ts[i]["Price"]

    #Only taking data from 2011 onwards to more accurately reflect current mar
    recent_series = series['2011':]

    # Splitting the last 36 months of our series as a test dataset.
    train_series = recent_series[:'2016-04']
    test_series = recent_series['2016-05':]

    #Auto ARIMA model
    auto_model = pm.auto_arima(train_series,
                     trace=True,
                     error_action= 'ignore',
                     suppress_warnings=True,
                     stepwise=True,with_intercept=False)

    # Plug the optimal parameter values for our Training data into a SARIMAX m
    ARIMA_MODEL = SARIMAX(recent_series,
                          order= auto_model.order,
                          seasonal_order= auto_model.seasonal_order,
                          enforce_stationarity=False,
                          enforce_invertibility=False)

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

    ## Getting a forecast for the next 36 months after the last absrecorded da
    forecast = output.get_forecast(36)
    prediction = forecast.conf_int()
    prediction['Price'] = forecast.predicted_mean
    prediction.columns = ['lower','upper','prediction']

    #Adding the Zipcode's ROI to the zip_predictions dictionary
    zip_predictions[df_ts[i]["Zipcode"].unique()[0]] = ((prediction['predicti
```

```
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0]             : AIC=inf, Time=0.17 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=1282.377, Time=0.01 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=1286.077, Time=0.02 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=1286.601, Time=0.04 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=1289.190, Time=0.03 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=1284.075, Time=0.01 sec

Best model:  ARIMA(0,2,0)(0,0,0)[0]
Total fit time: 0.294 seconds
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0]             : AIC=inf, Time=0.17 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=1318.385, Time=0.01 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=1319.761, Time=0.02 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=1321.654, Time=0.02 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=1323.825, Time=0.05 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=1320.293, Time=0.01 sec

Best model:  ARIMA(0,2,0)(0,0,0)[0]
Total fit time: 0.283 seconds
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0]             : AIC=inf, Time=0.17 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=976.105, Time=0.01 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=977.548, Time=0.02 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=976.676, Time=0.07 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=980.369, Time=0.03 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=977.652, Time=0.01 sec

Best model:  ARIMA(0,2,0)(0,0,0)[0]
Total fit time: 0.315 seconds
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0]             : AIC=inf, Time=0.21 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=1395.558, Time=0.01 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=1725.450, Time=0.02 sec
 ARIMA(0,1,1)(0,0,0)[0]             : AIC=1397.271, Time=0.02 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=inf, Time=0.19 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=1338.537, Time=0.01 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=1655.703, Time=0.03 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=1343.457, Time=0.03 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=1329.646, Time=0.12 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=1430.772, Time=0.09 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=1331.480, Time=0.22 sec
 ARIMA(0,1,2)(0,0,0)[0] intercept   : AIC=1345.995, Time=0.10 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=1371.075, Time=0.09 sec
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.19 sec

Best model:  ARIMA(1,1,1)(0,0,0)[0] intercept
Total fit time: 1.330 seconds
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0]             : AIC=inf, Time=0.22 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=1511.755, Time=0.01 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=1512.609, Time=0.06 sec
 ARIMA(0,1,1)(0,0,0)[0]             : AIC=1513.248, Time=0.02 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=inf, Time=0.11 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=1424.567, Time=0.01 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=1785.960, Time=0.02 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=1427.530, Time=0.05 sec
```

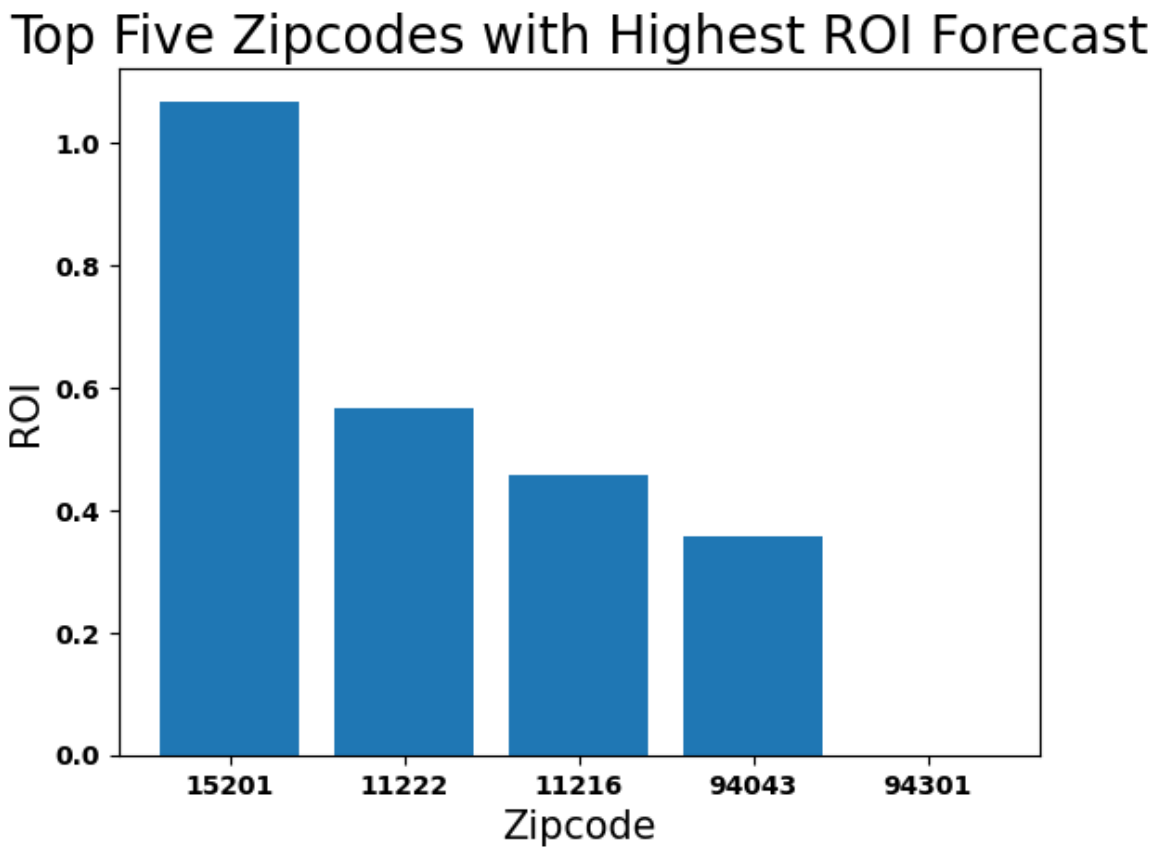
```
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=1521.453, Time=0.06 sec

Best model:  ARIMA(0,1,0)(0,0,0)[0] intercept
Total fit time: 0.558 seconds
```

- The goal of this process is to identify the ARIMA model with the lowest AIC value, which generally represents a good trade-off between model fit and complexity. However, please note that AIC alone may not be the only criterion for selecting the best model; you should also consider other factors such as model diagnostics, validation, and domain knowledge.

From the above analysis we noted that;

- For Zip Code 11216: Best ARIMA Model Order - ARIMA(0,2,0)(0,0,0)[0]
- For Zip Code 11222: Best ARIMA Model Order - ARIMA(0,2,0)(0,0,0)[0]
- For Zip Code 15201: Best ARIMA Model Order - ARIMA(0,2,0)(0,0,0)[0]
- For Zip Code 94043: Best ARIMA Model Order - ARIMA(0,2,0)(0,0,0)[0]
- For Zip Code 94301: Best ARIMA Model Order - ARIMA(1,1,1)(0,0,0)[0] intercept
- These specific ARIMA model orders were selected as they achieved the lowest AIC values among the evaluated options for each respective zip code. These orders encapsulate the autoregressive (AR), differencing (I), and moving average (MA) components of the ARIMA model, along with any seasonal components denoted by (p, d, q)(P, D, Q)s. This rigorous selection process aims to ensure that the chosen models strike a balance between fitting accuracy and simplicity, thus making them well-suited for future price forecasting tasks.

In [68]:
```
# Get the top five zipcodes with the highest ROI
top_zipcodes = sorted(zip_predictions, key=zip_predictions.get, reverse=True)

# Create a list of ROI values for the top five zipcodes
roi_values = [zip_predictions[zipcode] for zipcode in top_zipcodes]

# Create a bar graph of the top five zipcodes and their corresponding ROI valu
plt.bar(top_zipcodes, roi_values)
plt.xlabel('Zipcode',fontsize=15)
plt.ylabel('ROI',fontsize=15)
plt.title('Top Five Zipcodes with Highest ROI Forecast',fontsize=20)
plt.show()
```



- From the above graph the zipcode of 15201 has the highest ROI forecast so it would be beneficial if the investor invested in this zipcode.

In [69]:
```
zip_roi_dict = {zipcode: zip_predictions[zipcode] for zipcode in top_zipcodes

# Convert the dictionary to a pandas dataframe and sort it by ROI values in de
zip_roi_df = pd.DataFrame.from_dict(zip_roi_dict, orient='index', columns=['%
zip_roi_df = zip_roi_df.sort_values(by='% ROI', ascending=False)

# Print the dataframe
zip_roi_df.index.name = "Zipcode"
```

In [70]:
```
zip_roi_df
```

Out[70]:

Zipcode	% ROI
15201	1.069114
11222	0.566847
11216	0.457121
94043	0.356641
94301	0.000000

- Based on the above graph and computed percentages, 15201 have 106%, 11222 have 56.7%, 11216 have 45.7% and 94043 have 35.7% of returns. All the Zipcodes have an encouraging predicted price seeing as they are in the positive apart from the 94301 zipcode.
- Based on the above graph, we can conclude our top five recommendations and their expected ROI after three years.
- The investor can then decide to invest in any of the above zipcodes apart from 94301 which does not have a positive return on investment.

In [71]:
```
#Generating the zipcodes county and state
zip_codes = ['15201', '11222', '11216', '94043', '94301']

# Filter the data for the specified zip codes
filtered_data = data[data['Zipcode'].isin(zip_codes)]

# Create a dictionary to store zipcode: state: county pairs
zip_state_county = {}

# Iterate through the filtered data and populate the dictionary
for index, row in filtered_data.iterrows():
    zipcode = row['Zipcode']
    state = row['State']
    county = row['CountyName']
    zip_state_county[zipcode] = f"{state}: {county}"

# Print the formatted output
for zipcode, location in zip_state_county.items():
    print(f"{zipcode}: {location}")
```

```
11216: NY: Kings
11222: NY: Kings
94043: CA: Santa Clara
94301: CA: Santa Clara
15201: PA: Allegheny
```

Step 7.Conclusion

Comparative Analysis of the Forecasting Models

The analysis presented a comprehensive comparison of three forecasting models: Autoregressive Integrated Moving Average (SARIMA), and Baseline ARIMA. Each model was assessed based on their predictive performance, statistical significance of coefficients, and alignment with historical trends. Below is a summary of the findings:

Baseline ARIMA Model:

The Baseline ARIMA model demonstrated strong predictive performance with remarkably low Mean Squared Error (MSE) values for both the training and test datasets. Despite the promising results, there are indications of potential overfitting, as the models performed significantly better on the training data compared to the test data. The alignment between actual and predicted values was visually evident in the closely overlapping line graphs. The model's ability to capture and reproduce historical trends showcased its effectiveness.

Sarima Model:

The SARIMA models displayed competitive predictive performance, with relatively low MSE values across the test dataset. Each SARIMA model's parameter configuration was chosen to strike a balance between fitting and model simplicity. Notably, some p-values exceeded the significance threshold, indicating potential limitations in capturing certain variations. Despite these limitations, the SARIMA models effectively aligned with historical trends and provided insightful forecasts.

- Based on the models the best five zipe codes were identified as ;

1. **11216**: NY: Kings
2. **11222**: NY: Kings
3. **94043**: CA: Santa Clara
4. **94301**: CA: Santa Clara
5. **15201**: PA: Allegheny

In conclusion, all two models showcased valuable forecasting capabilities with their unique strengths and limitations. The Baseline ARIMA, Tuned ARIMA, and SARIMA models excelled in capturing temporal dependencies. Although the model's forecasting may have been affected by other limitations.

Step 8.Limitations

- Our model was delimited by overfitting wich may have caused our forecasting slightly deviate from the correct forecasting.

Step 9.Recommendations

- Based on the comprehensive analysis conducted using the provided dataset and considering the states of Pennsylvania, Carlifonia, and New york, below are some key recommendations:
- Consider Diversification: While all the analyzed states exhibit favorable investment opportunities, consider diversifying the investment portfolio across multiple states. This approach can help mitigate risk and capture various market dynamics.
- Explore South Carolina: Given the higher volume of data originating from South Carolina and the presence of two distinct cities (Greenville and Florence), exploring properties in this state could provide a stable investment environment.
- ROI as a Key Indicator: Return on Investment (ROI) is a critical metric for evaluating the potential profitability of investments. Focus on cities with higher ROI figures, as they indicate efficient investment opportunities. Kings, Santa Clara, and Allegheny stand out with robust ROI values.
- Risk Assessment: Take into consideration the Coefficient of Variation (CV) to assess investment risk. Lower CV values, like that of Greenville, indicate a more stable investment environment. Balancing ROI with risk is crucial for informed decision-making.
- Location Matters: Consider the city's unique characteristics, development prospects, and economic growth when making investment decisions. Evaluate factors like infrastructure, amenities, and potential for future demand.

In conclusion, Diversification, ROI analysis, risk assessment, and careful consideration of each city's unique attributes are crucial when making real estate investment decisions.

Step 10.Next Steps

- An external dataset with economic indicators can be integrated, to provide a holistic view of factors influencing housing prices.
- Augmenting the dataset with additional variables that might influence housing prices, such as interest rates, unemployment rates, or local economic indicators. This could improve the predictive accuracy of the choosen model. Housing market trends can change due to various unforeseen factors (e.g., economic downturns, pandemics, policy changes).
- To explore other tunimg techniques to improve on performance of the model.