

```
In [35]: import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import matplotlib.dates as mdates
import statsmodels.api as sm
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.statespace.sarimax import SARIMAX
from sklearn.metrics import mean_absolute_error, mean_squared_error
from tqdm import tqdm
import itertools
```

```
In [10]: # Load the data and parse dates
non_farm_data = pd.read_csv('PAYNSA.csv', parse_dates=['DATE'])
non_farm_data.set_index('DATE', inplace=True)
non_farm_data.index = non_farm_data.index.strftime('%Y-%m')
non_farm_data
```

Out[10]:

| PAYNSA  |        |
|---------|--------|
| DATE    |        |
| 1939-01 | 29296  |
| 1939-02 | 29394  |
| 1939-03 | 29804  |
| 1939-04 | 29786  |
| 1939-05 | 30145  |
| ...     | ...    |
| 2024-05 | 158842 |
| 2024-06 | 159341 |
| 2024-07 | 158399 |
| 2024-08 | 158717 |
| 2024-09 | 159177 |

1029 rows × 1 columns

```
In [68]: total_population = pd.read_csv('POPTOTUSA647NWDB.csv', parse_dates=['DATE'])
total_population.set_index('DATE', inplace=True)
total_population.index = total_population.index.strftime('%Y-%m')
total_population
```

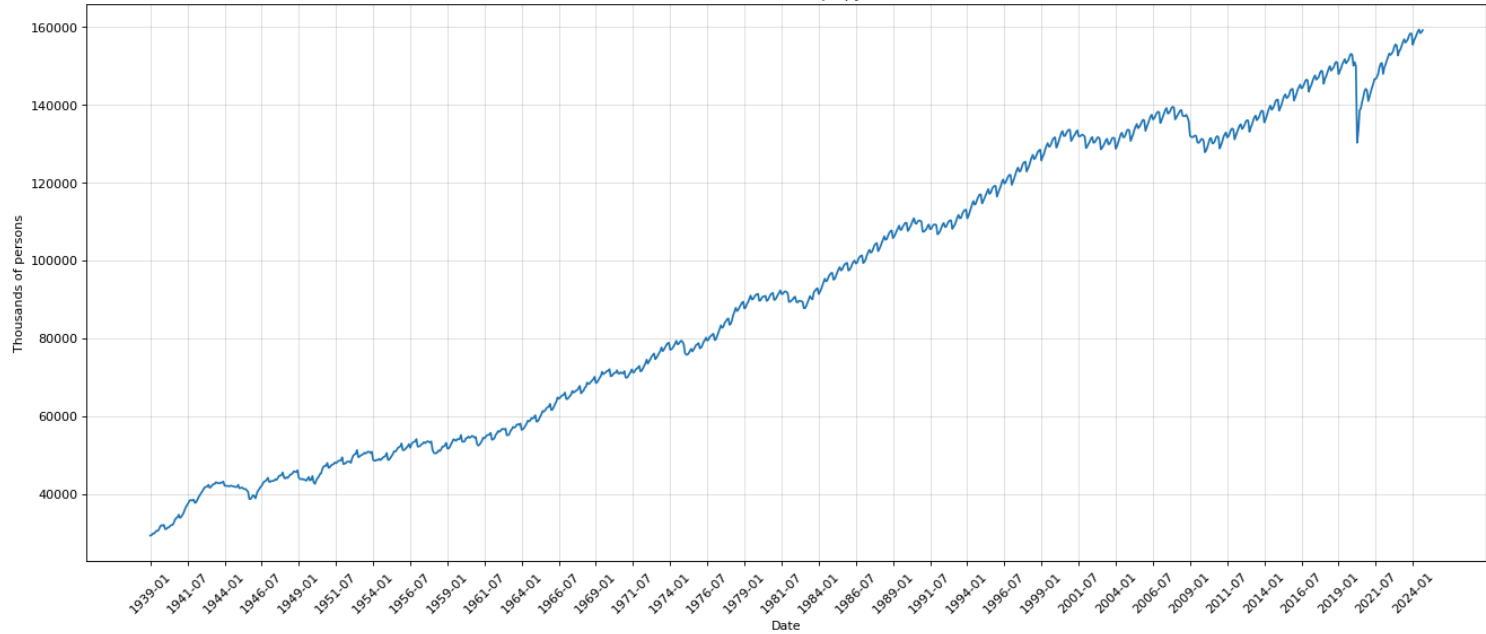
Out[68]:

| POPTOTUSA647NWDB |           |
|------------------|-----------|
| DATE             |           |
| 1960-01          | 180671000 |
| 1961-01          | 183691000 |
| 1962-01          | 186538000 |
| 1963-01          | 189242000 |
| 1964-01          | 191889000 |
| ...              | ...       |
| 2019-01          | 328329953 |
| 2020-01          | 331526933 |
| 2021-01          | 332048977 |
| 2022-01          | 333271411 |
| 2023-01          | 334914895 |

64 rows × 1 columns

```
In [32]: # Plot the figure
plt.figure(figsize=(20,8), dpi=80)
plt.plot(non_farm_data.index, non_farm_data['PAYNSA'])
plt.title('Historical Non-farm Employees data')
plt.yticks()
plt.xticks(non_farm_data.index[::30], rotation = 45)
plt.ylabel('Thousands of persons')
plt.xlabel('Date')
plt.grid(alpha=0.4)
plt.show()
```

Historical Non-farm Employees data



In [71]: # Assuming `non\_farm\_data` and `population\_data` have different date ranges

```
fig, ax1 = plt.subplots(figsize=(20, 8), dpi=80)

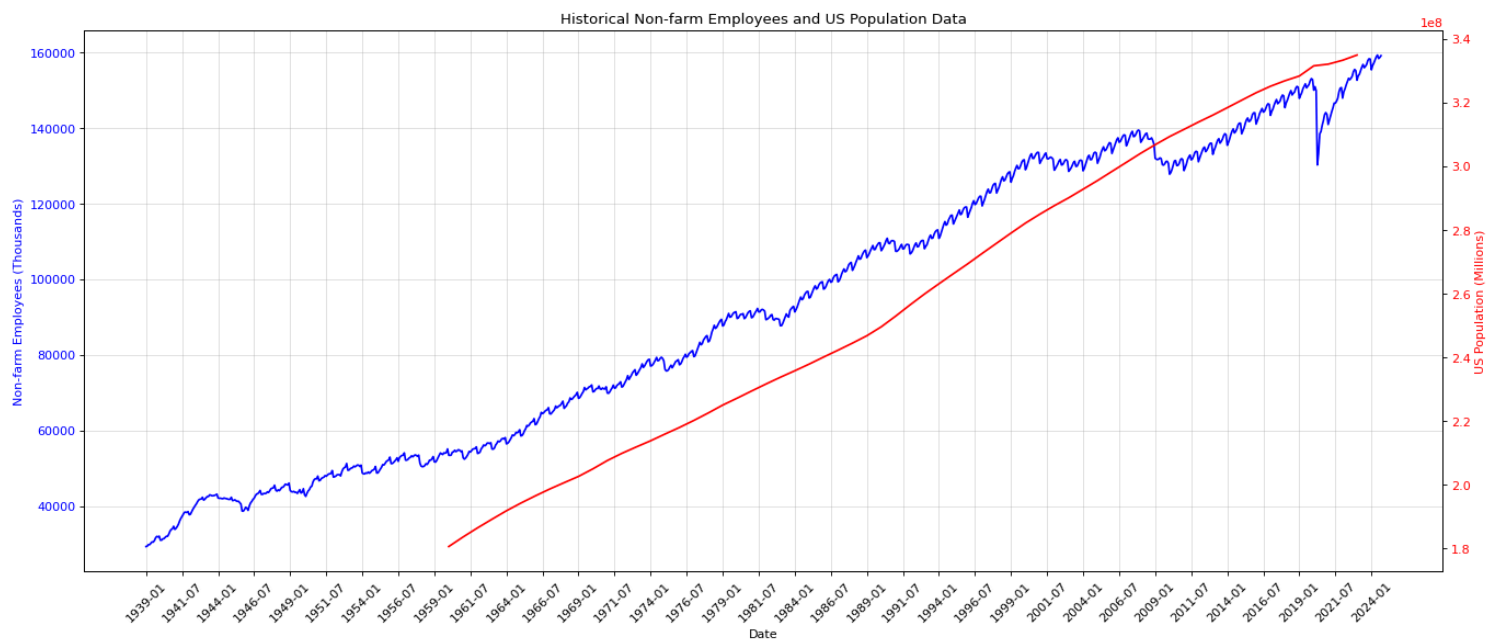
# Plot the non-farm employees data on the Left y-axis (ax1)
ax1.plot(non_farm_data.index, non_farm_data['PAYNSA'], color='b', label='Non-farm Employees (Thousands)')
ax1.set_xlabel('Date')
ax1.set_ylabel('Non-farm Employees (Thousands)', color='b')
ax1.tick_params(axis='y', labelcolor='b')
ax1.set_xticks(non_farm_data.index[::30])
ax1.set_xticklabels(non_farm_data.index[::30], rotation=45)

# Create a secondary y-axis for the population data
ax2 = ax1.twinx()
ax2.plot(total_population.index, total_population['POPTOTUSA647NWDB'], color='r', label='US Population (Millions)')
ax2.set_ylabel('US Population (Millions)', color='r')
ax2.tick_params(axis='y', labelcolor='r')

# Set the title
plt.title('Historical Non-farm Employees and US Population Data')

# Add grid for readability
ax1.grid(alpha=0.4)

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



#### AD-Fuller Test

```
In [73]: # Run the adfuller test
time_series = non_farm_data['PAYNSA'].dropna()

result = adfuller(time_series)
print(f'ADF Statistic: {result[0]}')
```

```
print(f'p-value: {result[1]}')
print(result[1]<0.05)
```

ADF Statistic: 0.13897515326502352  
p-value: 0.9686235485908465  
False

```
In [13]: # Apply first-order differencing
df_diff1 = non_farm_data['PAYNSA'].diff().dropna()

# Perform ADF test on differenced data
result_diff1 = adfuller(df_diff1)
print(f'ADF Statistic (1st Difference): {result_diff1[0]}')
print(f'p-value (1st Difference): {result_diff1[1]}')
print(result_diff1[1]<0.05)
```

ADF Statistic (1st Difference): -8.183795431744441  
p-value (1st Difference): 7.999713484582715e-13  
True

```
In [14]: time_series_diff = time_series.diff().dropna()

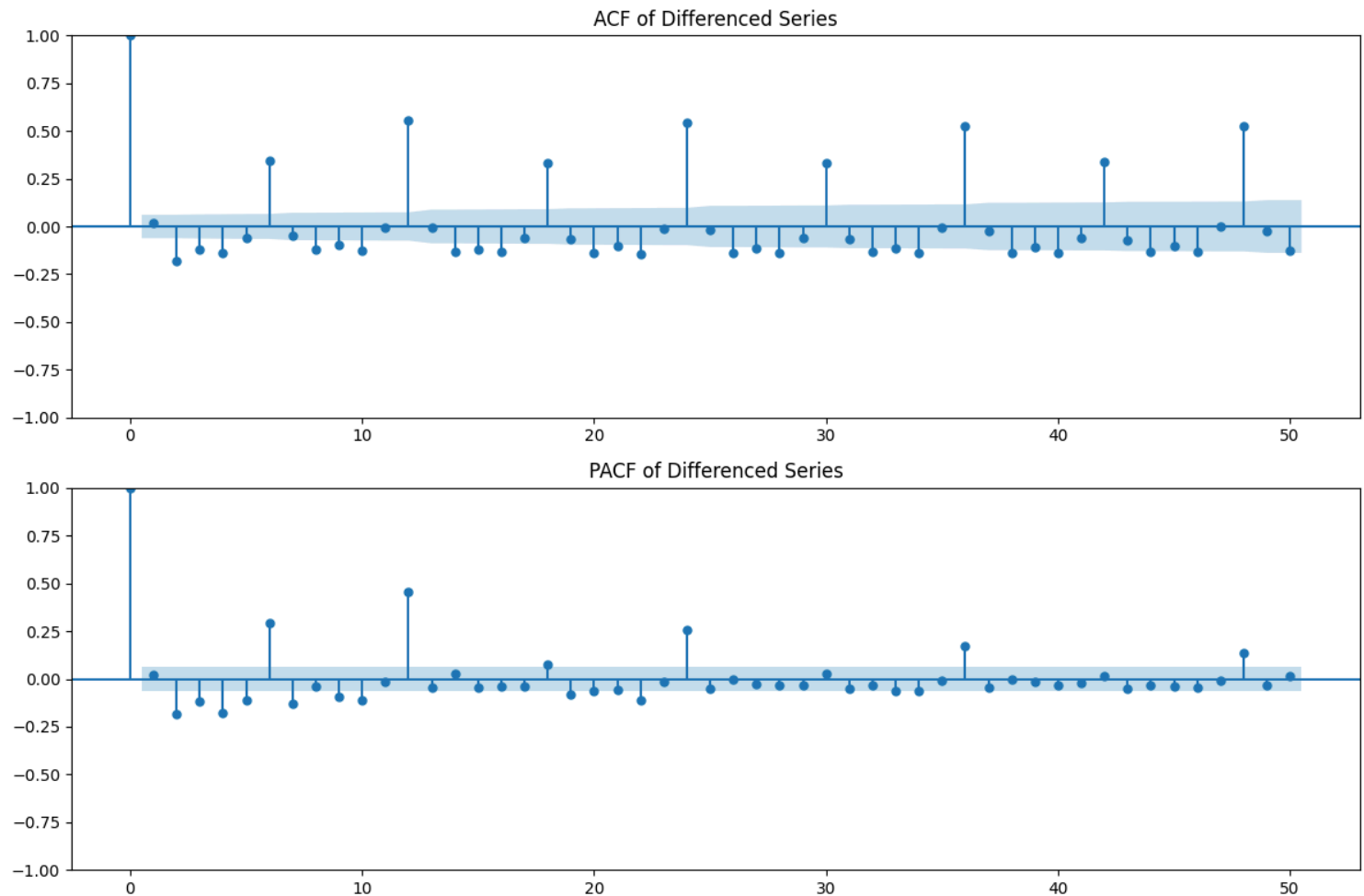
# Re-run the ADF test on the differenced data
result = adfuller(time_series_diff)
print(f'ADF Statistic: {result[0]}')
print(f'p-value: {result[1]}')
```

ADF Statistic: -8.183795431744441  
p-value: 7.999713484582715e-13

```
In [78]: # Plot ACF
fig, ax = plt.subplots(2,1, figsize=(12,8))
plot_acf(time_series_diff.dropna(), lags=50, ax=ax[0])
ax[0].set_title('ACF of Differenced Series')

# Plot PACF
plot_pacf(time_series_diff.dropna(), lags=50, ax=ax[1])
ax[1].set_title('PACF of Differenced Series')

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



#### SARIMAX Model

```
In [21]: model = sm.tsa.statespace.SARIMAX(
non_farm_data['PAYNSA'],
order=(1, 1, 1), # Replace with your determined (p, d, q)
seasonal_order=(1, 1, 1, 12), # Replace with your determined (P, D, Q, s)
enforce_stationarity=False,
enforce_invertibility=False)
```

```
)  
# Fit the model  
results = model.fit()  
print(results.summary())
```

C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11\_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa\_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.  
self.\_init\_dates(dates, freq)

C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11\_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa\_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.  
self.\_init\_dates(dates, freq)

```
SARIMAX Results  
=====
```

|                |                                |                   |           |
|----------------|--------------------------------|-------------------|-----------|
| Dep. Variable: | PAYNSA                         | No. Observations: | 1029      |
| Model:         | SARIMAX(1, 1, 1)x(1, 1, 1, 12) | Log Likelihood    | -8061.480 |
| Date:          | Mon, 14 Oct 2024               | AIC               | 16132.961 |
| Time:          | 23:18:45                       | BIC               | 16157.509 |
| Sample:        | 01-01-1939                     | HQIC              | 16142.290 |
|                | - 09-01-2024                   |                   |           |

Covariance Type: opg

```
=====
```

|          | coef      | std err  | z       | P> z  | [0.025   | 0.975]   |
|----------|-----------|----------|---------|-------|----------|----------|
| -----    | -----     | -----    | -----   | ----- | -----    | -----    |
| ar.L1    | -0.3343   | 0.070    | -4.747  | 0.000 | -0.472   | -0.196   |
| ma.L1    | 0.4413    | 0.069    | 6.405   | 0.000 | 0.306    | 0.576    |
| ar.S.L12 | 0.0354    | 0.014    | 2.588   | 0.010 | 0.009    | 0.062    |
| ma.S.L12 | -0.8885   | 0.012    | -71.530 | 0.000 | -0.913   | -0.864   |
| sigma2   | 5.408e+05 | 1572.037 | 344.042 | 0.000 | 5.38e+05 | 5.44e+05 |
| -----    | -----     | -----    | -----   | ----- | -----    | -----    |

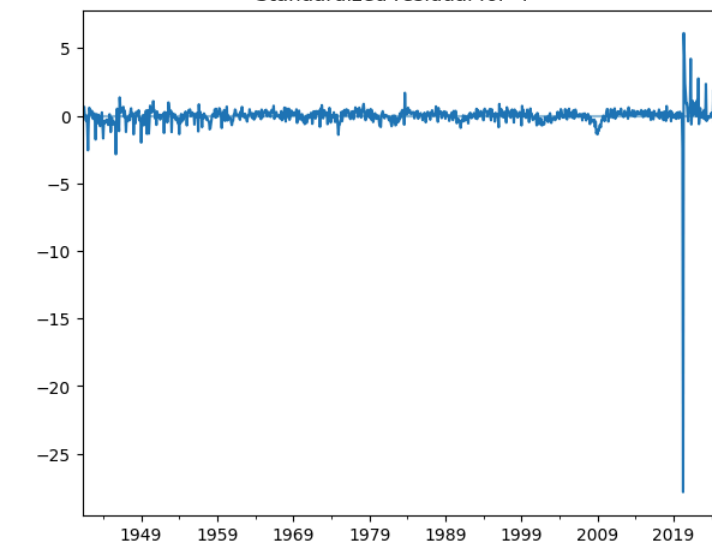
|                         |       |                   |             |
|-------------------------|-------|-------------------|-------------|
| Ljung-Box (L1) (Q):     | 0.20  | Jarque-Bera (JB): | 12753058.81 |
| Prob(Q):                | 0.65  | Prob(JB):         | 0.00        |
| Heteroskedasticity (H): | 11.29 | Skew:             | -19.88      |
| Prob(H) (two-sided):    | 0.00  | Kurtosis:         | 554.25      |

```
=====
```

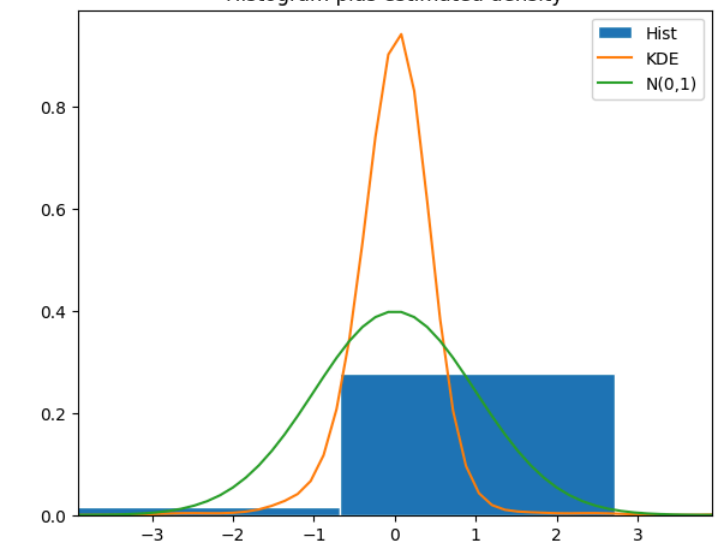
Warnings:  
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

```
In [22]: results.plot_diagnostics(figsize=(15, 12))  
plt.show()
```

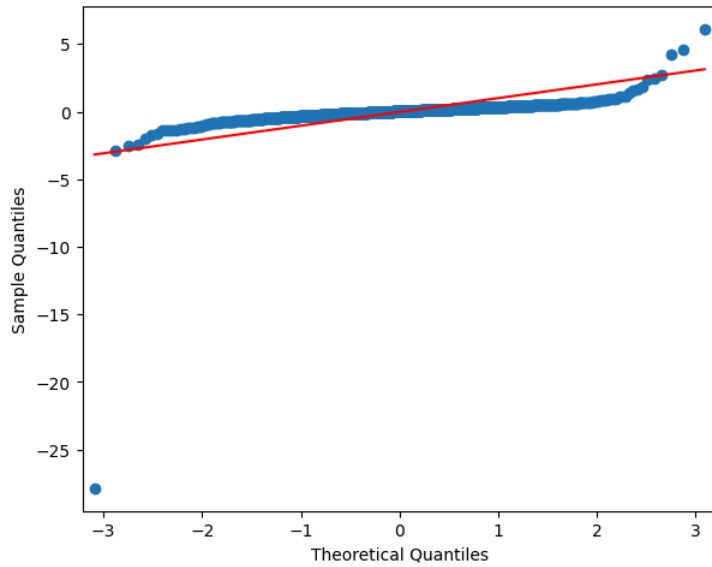
Standardized residual for "P"



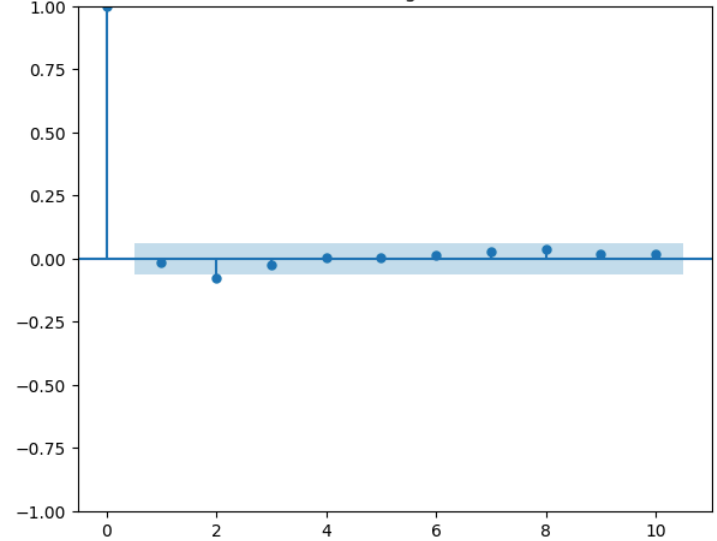
Histogram plus estimated density



Normal Q-Q



Correlogram



```
In [24]: forecast = results.get_forecast(steps=1)
forecast_mean = forecast.predicted_mean
forecast_conf_int = forecast.conf_int()
print(f"Forecasted Payroll for Next Month: {forecast_mean.iloc[0]:.2f}")
print(f"95% Confidence Interval: [{forecast_conf_int.iloc[0, 0]:.2f}, {forecast_conf_int.iloc[0, 1]:.2f}]"
```

Forecasted Payroll for Next Month: 160218.76  
95% Confidence Interval: [158777.36, 161660.17]

```
In [27]: # Input the actual value for September, 2024
actual_value = 254000
predicted_value = forecast_mean.iloc[0]

# Calculate the percentage difference
percentage_difference = ((actual_value - predicted_value) / actual_value) * 100
print(f"Forecasted value for September 2024: {predicted_value}")
print(f"Actual value for September 2024: {actual_value}")
print(f"Percentage difference: {percentage_difference:.2f}%"
```

Forecasted value for September 2024: 160218.7640215512  
Actual value for September 2024: 254000  
Percentage difference: 36.92%

```
In [28]: mae = mean_absolute_error([actual_value], [predicted_value])
mse = mean_squared_error([actual_value], [predicted_value])
rmse = np.sqrt(mse)

print(f'Mean Absolute Error (MAE): {mae:.2f}')
print(f'Root Mean Squared Error (RMSE): {rmse:.2f}')
```

Mean Absolute Error (MAE): 93781.24  
Root Mean Squared Error (RMSE): 93781.24

To find out the optimal SARIMAX model

```
In [30]: # Define p,d,q ranges
p = q = range(0, 3)
d = [1]
pdq = list(itertools.product(p, d, q))
```

```

# Define seasonal P, D, Q ranges
P = Q = range(0, 2)
D = [1]
seasonal_pdq = [(x[0], x[1], x[2], 12) for x in itertools.product(P, D, Q)]

# Grid search
aic_values = []
params = []

for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(
                non_farm_data['PAYNSA'],
                order=param,
                seasonal_order=param_seasonal,
                enforce_stationarity=False,
                enforce_invertibility=False
            )
            results = mod.fit(dispatch=False)
            aic_values.append(results.aic)
            params.append((param, param_seasonal))
            print(f'SARIMA{param}x{param_seasonal[:-1]}12 - AIC:{results.aic}')
        except:
            continue

# Find the parameters with minimal AIC
min_aic = min(aic_values)
best_params = params[aic_values.index(min_aic)]

print(f'\nBest SARIMA{best_params[0]}x{best_params[1][:-1]}12 model - AIC:{min_aic}')

```

```

C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
SARIMA(0, 1, 0)x(0, 1, 0)12 - AIC:16907.07285691912
SARIMA(0, 1, 0)x(0, 1, 1)12 - AIC:16153.73368517433
SARIMA(0, 1, 0)x(1, 1, 0)12 - AIC:16462.0010543683
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
SARIMA(0, 1, 0)x(1, 1, 1)12 - AIC:16154.308740202585
SARIMA(0, 1, 1)x(0, 1, 0)12 - AIC:16889.094778865132
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
SARIMA(0, 1, 1)x(0, 1, 1)12 - AIC:16133.571833353479
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
SARIMA(0, 1, 1)x(1, 1, 0)12 - AIC:16458.724315809657
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
SARIMA(0, 1, 1)x(1, 1, 1)12 - AIC:16134.365176096991
SARIMA(0, 1, 2)x(0, 1, 0)12 - AIC:16860.388691650347

```

[illegible]



SARIMA(1, 1, 2)x(0, 1, 0)12 - AIC:16861.989644547306



```
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood optimization failed to converge. Check mle_retvals
    warnings.warn("Maximum Likelihood optimization failed to ")
SARIMA(2, 1, 2)x(1, 1, 0)12 - AIC:16427.622653703143

C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
SARIMA(2, 1, 2)x(1, 1, 1)12 - AIC:16111.174415047739

Best SARIMA(0, 1, 2)x(0, 1, 1)12 model - AIC:16107.588405127168
```

```
In [31]: # Best parameters from grid search
best_order = best_params[0]
best_seasonal_order = best_params[1]

# Refit the model
model = sm.tsa.statespace.SARIMAX(
    non_farm_data['PAYNSA'],
    order=best_order,
    seasonal_order=best_seasonal_order,
    enforce_stationarity=False,
    enforce_invertibility=False
)
results = model.fit()
print(results.summary())
```

```
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base
\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
    self._init_dates(dates, freq)
```

| SARIMAX Results         |                                  |                   |                   |           |          |         |
|-------------------------|----------------------------------|-------------------|-------------------|-----------|----------|---------|
| =====                   |                                  |                   |                   |           |          |         |
| Dep. Variable:          | PAYNSA                           |                   | No. Observations: | 1029      |          |         |
| Model:                  | SARIMAX(0, 1, 2)x(0, 1, [1], 12) |                   | Log Likelihood    | -8049.794 |          |         |
| Date:                   | Mon, 14 Oct 2024                 |                   | AIC               | 16107.588 |          |         |
| Time:                   | 23:25:12                         |                   | BIC               | 16127.223 |          |         |
| Sample:                 | 01-01-1939                       |                   | HQIC              | 16115.051 |          |         |
|                         | - 09-01-2024                     |                   |                   |           |          |         |
| Covariance Type:        | opg                              |                   |                   |           |          |         |
| =====                   |                                  |                   |                   |           |          |         |
|                         | coef                             | std err           | z                 | P> z      | [0.025   | 0.975]  |
| -----                   |                                  |                   |                   |           |          |         |
| ma.L1                   | 0.0787                           | 0.007             | 11.593            | 0.000     | 0.065    | 0.092   |
| ma.L2                   | -0.1113                          | 0.013             | -8.844            | 0.000     | -0.136   | -0.087  |
| ma.S.L12                | -0.8830                          | 0.008             | -107.695          | 0.000     | -0.899   | -0.867  |
| sigma2                  | 5.367e+05                        | 1554.336          | 345.316           | 0.000     | 5.34e+05 | 5.4e+05 |
| =====                   |                                  |                   |                   |           |          |         |
| Ljung-Box (L1) (Q):     | 0.01                             | Jarque-Bera (JB): | 13306595.20       |           |          |         |
| Prob(Q):                | 0.92                             | Prob(JB):         | 0.00              |           |          |         |
| Heteroskedasticity (H): | 10.54                            | Skew:             | -20.43            |           |          |         |
| Prob(H) (two-sided):    | 0.00                             | Kurtosis:         | 566.36            |           |          |         |
| =====                   |                                  |                   |                   |           |          |         |

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Looking for the historical maximum differences

```
In [36]: # Training window
train_window = 120

# Forecast horizon
forecast_horizon = 1
```

```
In [37]: actual_values = []
predicted_values = []
differences = []
dates = []
```

```
In [38]: # Start the rolling forecast
for i in tqdm(range(train_window, len(non_farm_data) - forecast_horizon + 1)):
    # Define training and test sets
    train_data = non_farm_data.iloc[i - train_window:i]['PAYNSA']
    test_data = non_farm_data.iloc[i:i + forecast_horizon]['PAYNSA']

    # Fit SARIMA model on training data
    model = SARIMAX(train_data,
                     order=(1, 1, 1),
                     seasonal_order=(1, 1, 1, 12),
                     enforce_stationarity=False,
                     enforce_invertibility=False)
    results = model.fit(dispatch=False)
```

```
# Forecast
forecast = results.forecast(steps=forecast_horizon)

# Store the results
actual = test_data.iloc[0]
predicted = forecast.iloc[0]
difference = abs(actual - predicted)

actual_values.append(actual)
predicted_values.append(predicted)
differences.append(difference)
dates.append(test_data.index[0])
```

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]



[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

[illegible]



[illegible]



[illegible]

[illegible]

```

self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
99%|██████████| 902/909 [05:21<00:03, 2.22it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
99%|██████████| 903/909 [05:22<00:02, 2.23it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
99%|██████████| 904/909 [05:22<00:02, 2.14it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
100%|██████████| 905/909 [05:22<00:01, 2.25it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
100%|██████████| 906/909 [05:23<00:01, 2.19it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
100%|██████████| 907/909 [05:23<00:00, 2.37it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
100%|██████████| 908/909 [05:24<00:00, 2.32it/s]C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
C:\Users\67003\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
self._init_dates(dates, freq)
100%|██████████| 909/909 [05:24<00:00, 2.80it/s]

```

```

In [54]: results_df = pd.DataFrame({
        'Date': dates,
        'Actual': actual_values,
        'Predicted': predicted_values,
        'Difference': differences
    }).set_index('Date')

```

```

In [56]: # Handle potential division by zero
results_df['Actual'].replace(0, np.nan, inplace=True)

# Calculate percentage differences
results_df['Percentage Difference'] = 100 * results_df['Difference'] / results_df['Actual']

```

C:\Users\67003\AppData\Local\Temp\ipykernel\_9948\4175090180.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.

```
results_df['Actual'].replace(0, np.nan, inplace=True)
```

```

In [58]: # Find maximum percentage difference
max_percentage_difference = results_df['Percentage Difference'].max()
max_percentage_date = results_df['Percentage Difference'].idxmax()

print(f"The maximum percentage difference is {max_percentage_difference:.2f}%")
print(max_percentage_date)

```

The maximum percentage difference is 132.22%  
2020-05

```

In [45]: # Find the maximum difference
max_difference = results_df['Difference'].max()
max_difference_date = results_df['Difference'].idxmax()

print(f"The maximum difference is {max_difference:.2f} on {max_difference_date.date()}")

```

The maximum difference is 176416.26

```
In [79]: sorted_results = results_df.sort_values(by='Percentage Difference', ascending=False)
top_5_differences = sorted_results.head(5)
print("Top 5 Percentage Differences:")
top_5_differences
```

Top 5 Percentage Differences:

```
Out[79]:
```

|         | Actual | Predicted     | Difference    | Percentage Difference |
|---------|--------|---------------|---------------|-----------------------|
| Date    |        |               |               |                       |
| 2020-05 | 133422 | -42994.262741 | 176416.262741 | 132.224268            |
| 2020-04 | 130253 | 151735.273467 | 21482.273467  | 16.492728             |
| 2022-04 | 151434 | 131247.903574 | 20186.096426  | 13.329963             |
| 2021-04 | 144402 | 149675.012136 | 5273.012136   | 3.651620              |
| 2020-06 | 138507 | 134201.514845 | 4305.485155   | 3.108496              |

```
In [51]: # Error metrics
mae = mean_absolute_error(results_df['Actual'], results_df['Predicted'])
rmse = np.sqrt(mean_squared_error(results_df['Actual'], results_df['Predicted']))
mape = np.mean(results_df['Percentage Difference'].dropna())

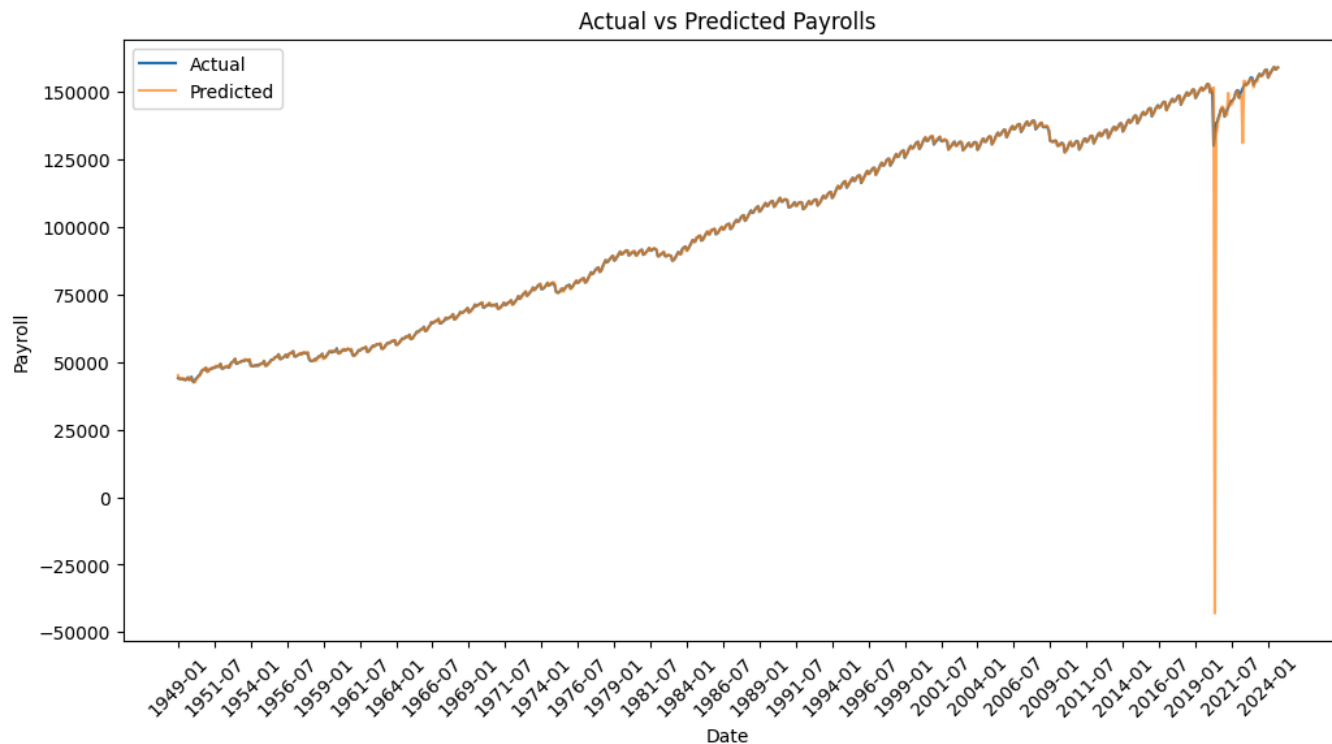
print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
print(f"Mean Absolute Percentage Error (MAPE): {mape:.2f}%")
```

Mean Absolute Error (MAE): 426.69

Root Mean Squared Error (RMSE): 5943.13

Mean Absolute Percentage Error (MAPE): 0.40%

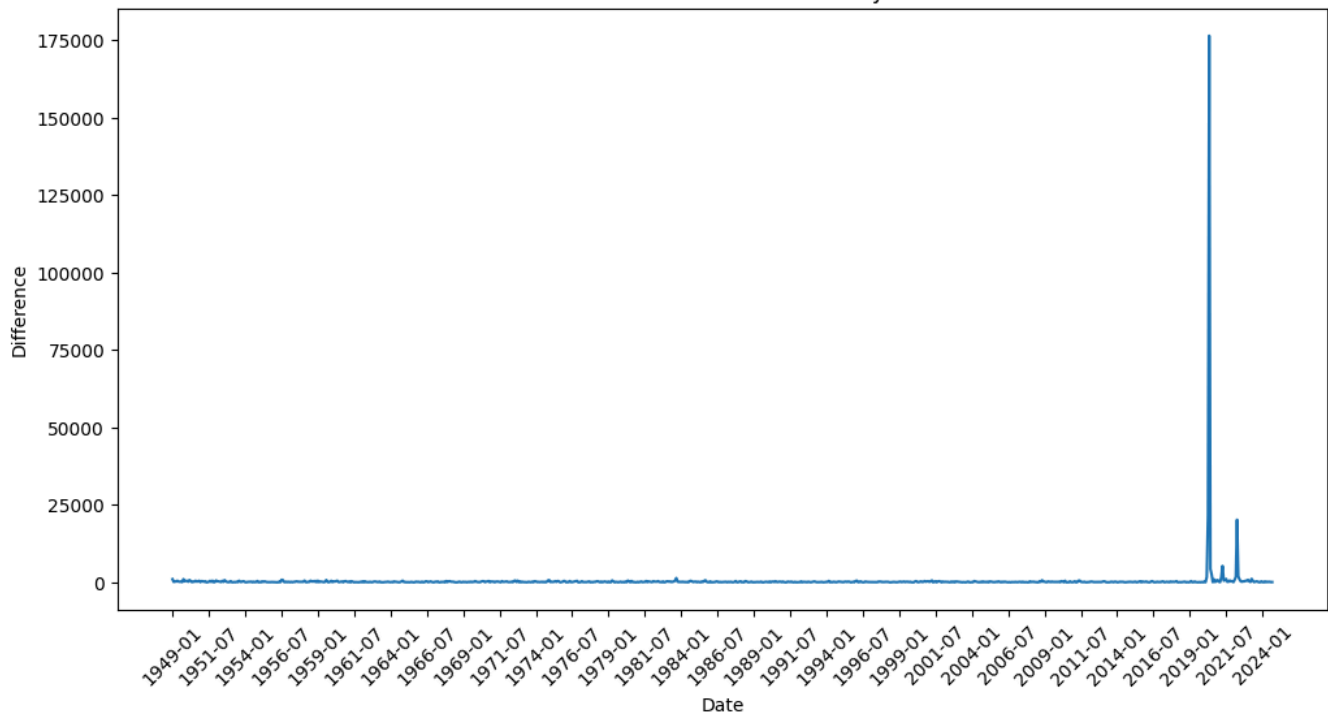
```
In [46]: plt.figure(figsize=(12, 6))
plt.plot(results_df.index, results_df['Actual'], label='Actual')
plt.plot(results_df.index, results_df['Predicted'], label='Predicted', alpha=0.7)
plt.title('Actual vs Predicted Payrolls')
plt.xticks(results_df.index[::30], rotation = 45)
plt.xlabel('Date')
plt.ylabel('Payroll')
plt.legend()
plt.show()
```



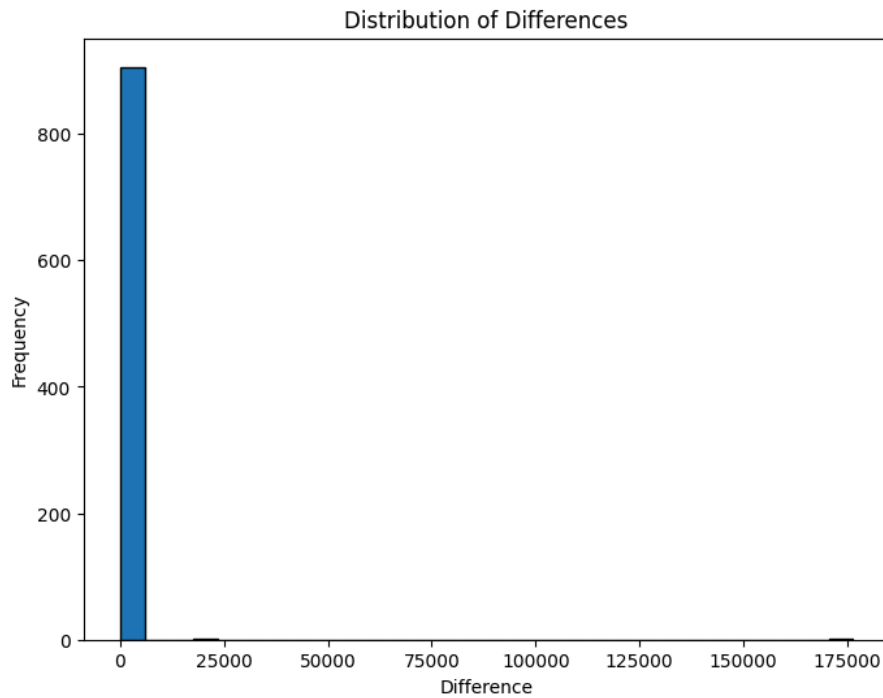
```
In [47]: plt.figure(figsize=(12, 6))
plt.plot(results_df.index, results_df['Difference'])
plt.title('Difference Between Actual and Predicted Payrolls Over Time')
plt.xticks(results_df.index[::30], rotation = 45)
plt.xlabel('Date')
plt.ylabel('Difference')
plt.show()
```



Difference Between Actual and Predicted Payrolls Over Time



```
In [43]: plt.figure(figsize=(8, 6))
plt.hist(results_df['Difference'], bins=30, edgecolor='k')
plt.title('Distribution of Differences')
plt.xlabel('Difference')
plt.ylabel('Frequency')
plt.show()
```



```
In [80]: # Set a threshold for significant differences (e.g., top 5% differences)
threshold = results_df['Difference'].quantile(0.95)

# Get dates with significant differences
significant_errors = results_df[results_df['Difference'] >= threshold]

print("Dates with significant differences:")
significant_errors
```

Dates with significant differences:

| Out[80]: | Actual | Predicted     | Difference    | Percentage Difference |
|----------|--------|---------------|---------------|-----------------------|
| Date     |        |               |               |                       |
| 1949-01  | 44159  | 45158.346759  | 999.346759    | 2.263065              |
| 1949-10  | 43447  | 44457.945966  | 1010.945966   | 2.326849              |
| 1950-03  | 43410  | 42663.146727  | 746.853273    | 1.720464              |
| 1951-10  | 48550  | 48033.628376  | 516.371624    | 1.063587              |
| 1952-01  | 47659  | 48242.340580  | 583.340580    | 1.223988              |
| 1952-08  | 49077  | 48357.854559  | 719.145441    | 1.465341              |
| 1956-07  | 51847  | 52634.885947  | 787.885947    | 1.519637              |
| 1956-08  | 52802  | 52040.614830  | 761.385170    | 1.441963              |
| 1958-02  | 50588  | 51138.592144  | 550.592144    | 1.088385              |
| 1959-01  | 51721  | 51201.900628  | 519.099372    | 1.003653              |
| 1959-08  | 53636  | 54376.837391  | 740.837391    | 1.381232              |
| 1972-08  | 74054  | 73491.899536  | 562.100464    | 0.759041              |
| 1972-10  | 75290  | 74759.281198  | 530.718802    | 0.704899              |
| 1974-11  | 79060  | 79643.367238  | 583.367238    | 0.737879              |
| 1974-12  | 78419  | 79133.471072  | 714.471072    | 0.911094              |
| 1979-04  | 89313  | 89914.798129  | 601.798129    | 0.673808              |
| 1980-05  | 90763  | 91303.813548  | 540.813548    | 0.595852              |
| 1983-08  | 89998  | 90584.201528  | 586.201528    | 0.651350              |
| 1983-09  | 91719  | 90349.821247  | 1369.178753   | 1.492797              |
| 1985-09  | 98476  | 99144.113354  | 668.113354    | 0.678453              |
| 1996-02  | 117364 | 116827.328356 | 536.671644    | 0.457271              |
| 2001-04  | 132335 | 132935.476748 | 600.476748    | 0.453755              |
| 2008-11  | 136744 | 137416.315355 | 672.315355    | 0.491660              |
| 2011-06  | 132871 | 132309.614049 | 561.385951    | 0.422504              |
| 2020-03  | 149952 | 151649.181059 | 1697.181059   | 1.131816              |
| 2020-04  | 130253 | 151735.273467 | 21482.273467  | 16.492728             |
| 2020-05  | 133422 | -42994.262741 | 176416.262741 | 132.224268            |
| 2020-06  | 138507 | 134201.514845 | 4305.485155   | 3.108496              |
| 2020-07  | 139105 | 136519.883385 | 2585.116615   | 1.858392              |
| 2020-09  | 141957 | 141029.657825 | 927.342175    | 0.653256              |
| 2020-11  | 144116 | 144704.801472 | 588.801472    | 0.408561              |
| 2020-12  | 143604 | 144333.450688 | 729.450688    | 0.507960              |
| 2021-03  | 143308 | 141595.231872 | 1712.768128   | 1.195166              |
| 2021-04  | 144402 | 149675.012136 | 5273.012136   | 3.651620              |
| 2021-05  | 145392 | 144869.160438 | 522.839562    | 0.359607              |
| 2021-06  | 146626 | 145982.700050 | 643.299950    | 0.438735              |
| 2021-07  | 146619 | 145540.228109 | 1078.771891   | 0.735765              |
| 2021-10  | 149605 | 149022.879886 | 582.120114    | 0.389105              |
| 2022-02  | 149606 | 148820.938486 | 785.061514    | 0.524753              |
| 2022-03  | 150411 | 148629.547625 | 1781.452375   | 1.184390              |
| 2022-04  | 151434 | 131247.903574 | 20186.096426  | 13.329963             |
| 2022-05  | 152264 | 154176.299002 | 1912.299002   | 1.255910              |
| 2022-06  | 153175 | 154134.754179 | 959.754179    | 0.626574              |
| 2023-01  | 152688 | 151886.276995 | 801.723005    | 0.525073              |
| 2023-04  | 155201 | 154090.307338 | 1110.692662   | 0.715648              |
| 2023-05  | 156132 | 155612.037826 | 519.962174    | 0.333027              |