arima

May 29, 2023

1 PREDICTION WITH ARIMA

[2]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from math import sqrt
from sklearn.metrics import mean_squared_error
from statsmodels.tsa.arima.model import ARIMA
```

[3]:
```
# read dataset may2023
df = pd.read_pickle("../../data/20230319_RTU_Dataset_PPC-Lab/combined_may2023.
↪pkl")
```

[4]:
```
df
```

[4]:

	MEM_USAGE	CPU_USAGE	PS1_V	TEMP
0	35.555417	27.343750	5.435294	28.687
1	35.555417	6.367041	5.435294	28.687
2	35.555417	7.142857	5.435294	28.687
3	35.555417	27.306273	5.435294	28.687
4	35.555417	5.639098	5.435294	28.687
…	…	…	…	…
3798	25.962425	8.396947	5.383530	29.562
3799	25.962425	6.766917	5.383530	29.562
3800	25.962425	6.000000	5.383530	29.562
3801	25.962425	8.045977	5.383530	29.562
3802	25.962425	13.229572	5.383530	29.562

[3733 rows x 4 columns]

[73]:
```
LAG = 12   # ---------- 1H
```

[56]:
```
def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100
```

2 TEMP

[57]:
```
training_size = int(len(df) * 0.8)

train = [[i] for i in df["TEMP"]][:training_size]
test = [[i] for i in df["TEMP"]][training_size:]

len(train)
```

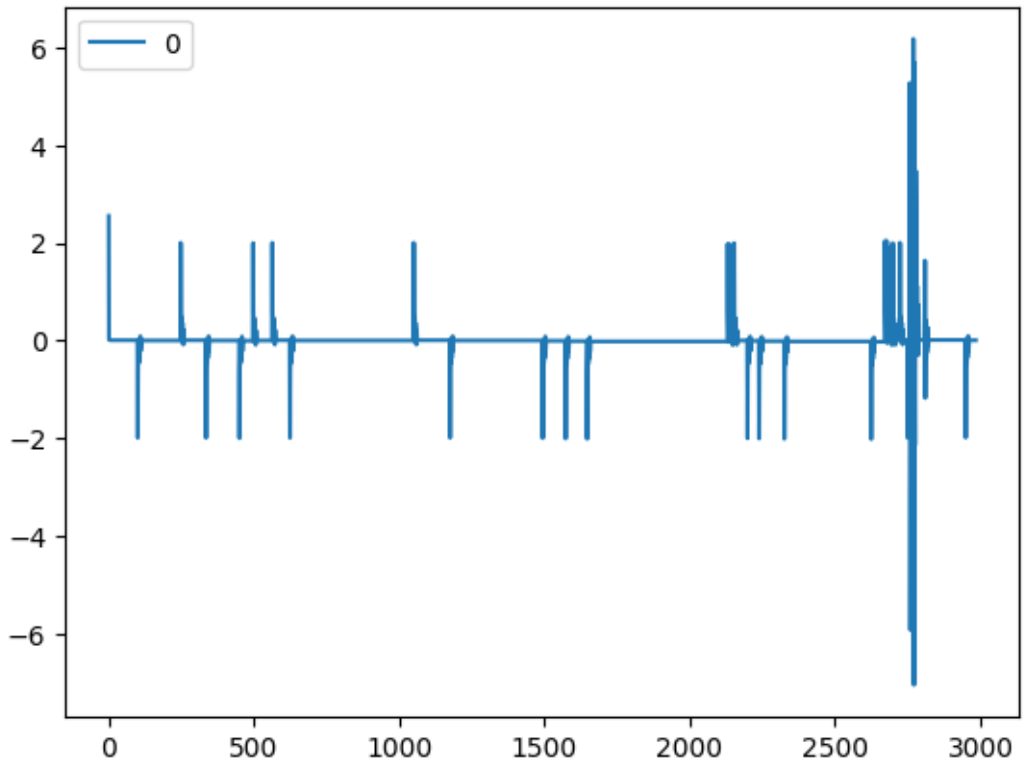
[57]: 2986

[59]:
```
arima = ARIMA(train, order=(LAG,0,0))
arima_fit = arima.fit()
print(arima_fit.summary())
```

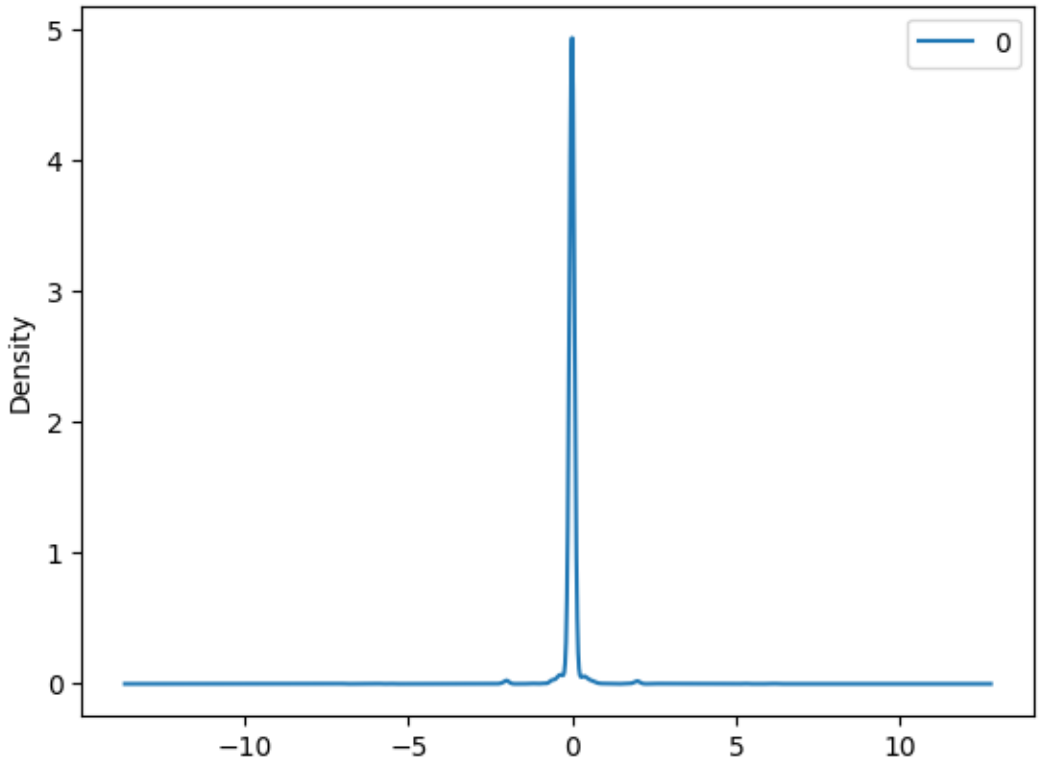
```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                 2986
Model:                ARIMA(12, 0, 0)   Log Likelihood               -1211.768
Date:                Thu, 25 May 2023   AIC                           2451.536
Time:                        15:01:07   BIC                           2535.560
Sample:                             0   HQIC                          2481.766
                                 - 2986
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         26.1354      2.420     10.799      0.000      21.392      30.879
ar.L1          0.6926      0.005    138.863      0.000       0.683       0.702
ar.L2          0.1343      0.003     38.752      0.000       0.127       0.141
ar.L3          0.1785      0.004     39.917      0.000       0.170       0.187
ar.L4         -0.2374      0.007    -33.110      0.000      -0.251      -0.223
ar.L5          0.2011      0.007     28.659      0.000       0.187       0.215
ar.L6         -0.1500      0.006    -23.960      0.000      -0.162      -0.138
ar.L7          0.1480      0.009     16.494      0.000       0.130       0.166
ar.L8          0.0408      0.011      3.831      0.000       0.020       0.062
ar.L9          0.0333      0.015      2.287      0.022       0.005       0.062
ar.L10        -0.1503      0.012    -12.327      0.000      -0.174      -0.126
ar.L11         0.0682      0.008      8.184      0.000       0.052       0.085
ar.L12         0.0377      0.009      4.369      0.000       0.021       0.055
sigma2         0.1316      0.001    158.893      0.000       0.130       0.133
==============================================================================
===
Ljung-Box (L1) (Q):                   0.18   Jarque-Bera (JB):
3359306.30
Prob(Q):                              0.67   Prob(JB):
0.00
Heteroskedasticity (H):               9.74   Skew:
```

```
-0.80
Prob(H) (two-sided):                  0.00   Kurtosis:
167.31
================================================================================
===

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

[60]:
```
residuals = pd.DataFrame(arima_fit.resid)
residuals.plot()
plt.show()
```



[61]:
```
# density plot of residuals
residuals.plot(kind='kde')
plt.show()
# summary stats of residuals
print(residuals.describe())
```

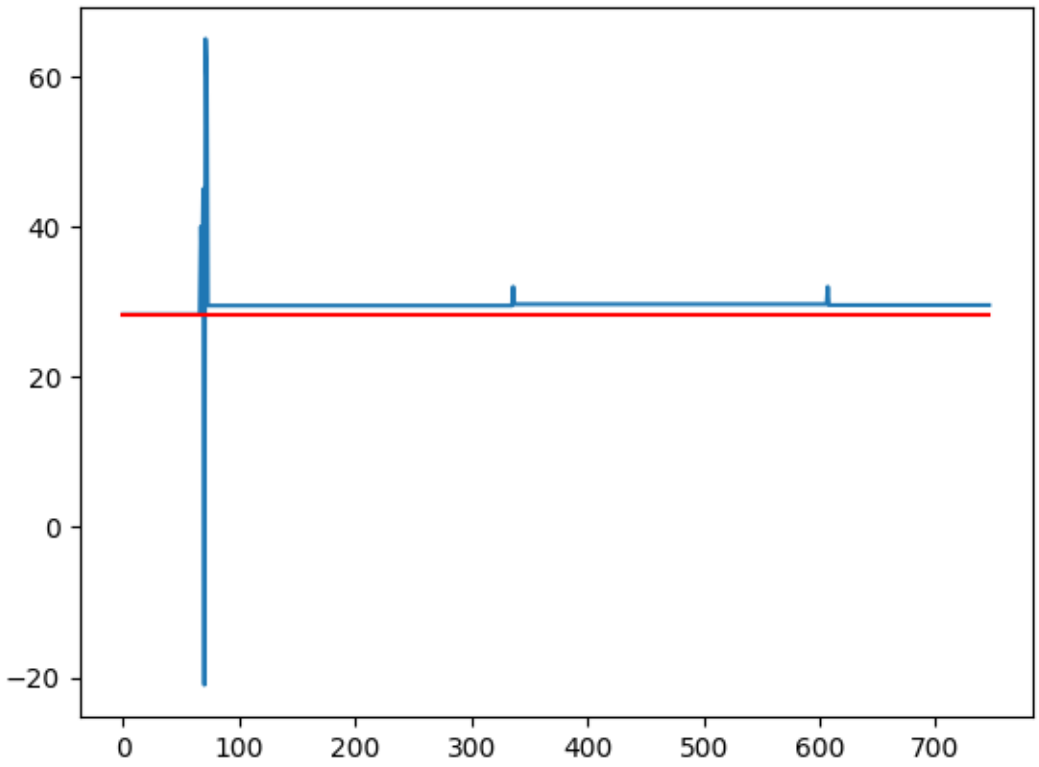


	0
count	2986.000000
mean	-0.002585
std	0.365788
min	-7.048656
25%	-0.016902
50%	0.001711
75%	0.001711
max	6.177039

[62]:
```
history = [x for x in train]
predictions = list()
for t in range(len(test)):
    output = arima_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test[t]
    history.append(obs)
    # print('predicted', yhat, obs)
    # print('actual', obs)
```

[63]:
```
# evaluate forecasts
rmse = sqrt(mean_squared_error(test, predictions))
mape = mean_absolute_percentage_error(test, predictions)
print('Test MAPE: %.3f' % mape)
print('Test RMSE: %.3f' % rmse)
# plot forecasts against actual outcomes
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

```
Test MAPE: 4.497
Test RMSE: 2.915
```



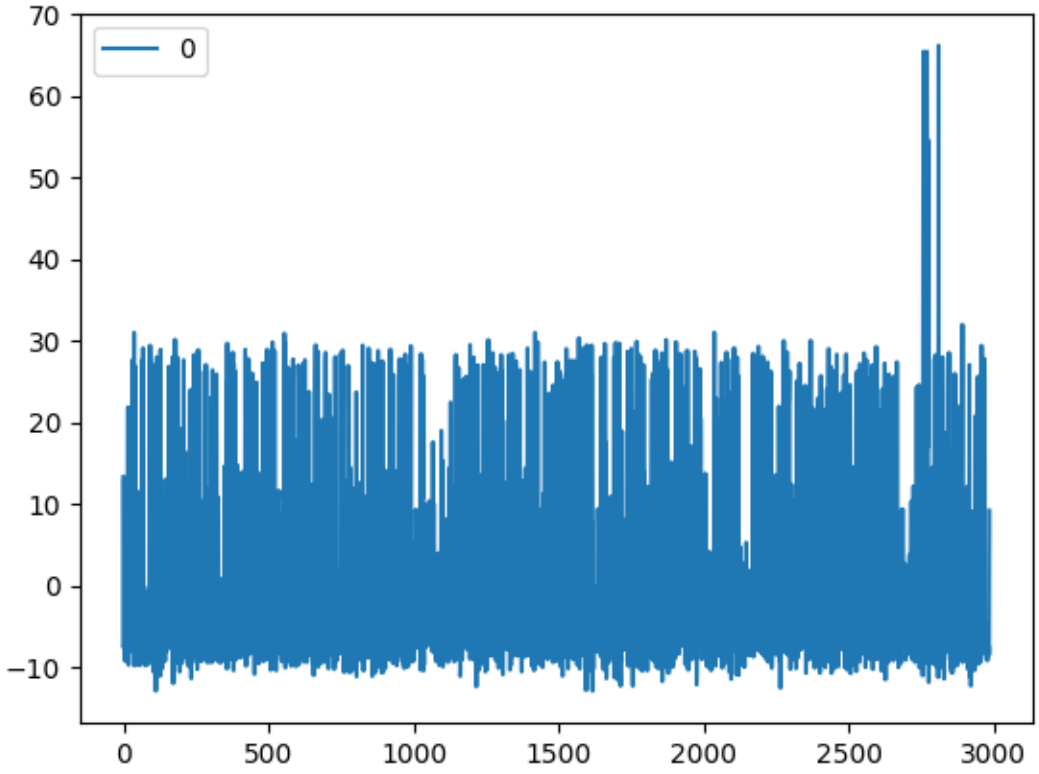
3 CPU

[64]:
```
training_size = int(len(df) * 0.8)

train = [[i] for i in df["CPU_USAGE"]][:training_size]
test = [[i] for i in df["CPU_USAGE"]][training_size:]

arima = ARIMA(train, order=(LAG,0,0))
```

```
arima_fit = arima.fit()
print(arima_fit.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                          y   No. Observations:                 2986
Model:                 ARIMA(12, 0, 0)   Log Likelihood              -11431.464
Date:                Thu, 25 May 2023   AIC                          22890.929
Time:                        15:01:25   BIC                          22974.952
Sample:                             0   HQIC                         22921.159
                                 - 2986
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         13.9781      0.376     37.143      0.000      13.241      14.716
ar.L1         -0.0007      0.016     -0.045      0.964      -0.031       0.030
ar.L2          0.0124      0.017      0.713      0.476      -0.022       0.046
ar.L3          0.0285      0.017      1.703      0.089      -0.004       0.061
ar.L4         -0.0149      0.018     -0.825      0.410      -0.050       0.021
ar.L5          0.0385      0.018      2.146      0.032       0.003       0.074
ar.L6          0.0415      0.018      2.287      0.022       0.006       0.077
ar.L7         -0.0138      0.019     -0.728      0.467      -0.051       0.023
ar.L8          0.0102      0.018      0.565      0.572      -0.025       0.046
ar.L9         -0.0203      0.018     -1.103      0.270      -0.056       0.016
ar.L10         0.0187      0.018      1.032      0.302      -0.017       0.054
ar.L11         0.0062      0.017      0.370      0.711      -0.026       0.039
ar.L12         0.0071      0.016      0.454      0.650      -0.023       0.038
sigma2       123.8167      3.700     33.463      0.000     116.565     131.069
==============================================================================
===
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
2150.56
Prob(Q):                              1.00   Prob(JB):
0.00
Heteroskedasticity (H):               1.10   Skew:
1.65
Prob(H) (two-sided):                  0.13   Kurtosis:
5.52
==============================================================================
===

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

[65]:
```
residuals = pd.DataFrame(arima_fit.resid)
residuals.plot()
plt.show()
```

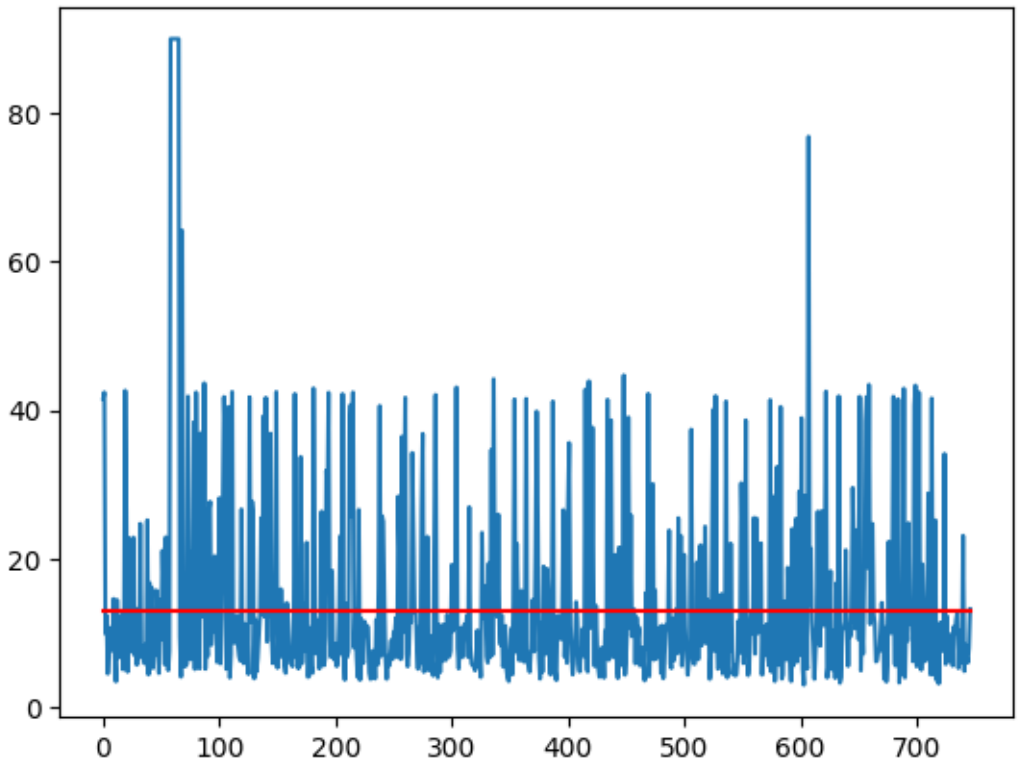


[66]:
```
history = [x for x in train]
predictions = list()
for t in range(len(test)):
    output = arima_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test[t]
    history.append(obs)
    # print('predicted', yhat, obs)
    # print('actual', obs)
```

[67]:
```
# evaluate forecasts
rmse = sqrt(mean_squared_error(test, predictions))
mape = mean_absolute_percentage_error(test, predictions)
print('Test MAPE: %.3f' % mape)
print('Test RMSE: %.3f' % rmse)
# plot forecasts against actual outcomes
plt.plot(test)
```

```
plt.plot(predictions, color='red')
plt.show()
```

```
Test MAPE: 76.357
Test RMSE: 13.574
```



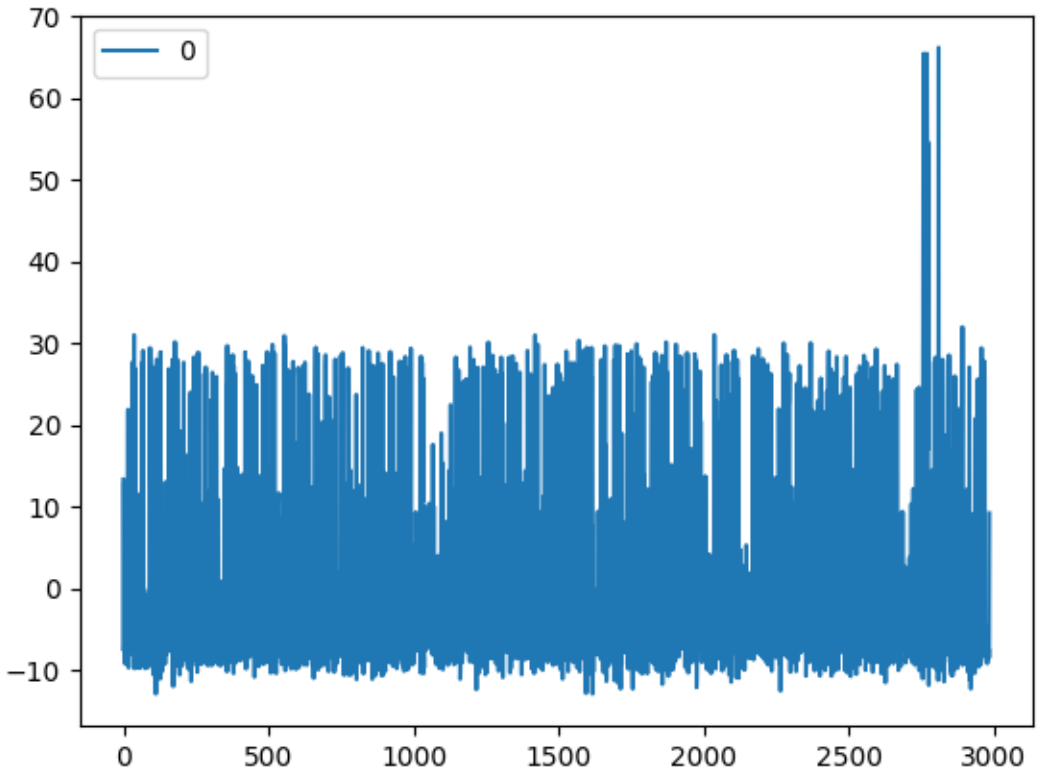
4 MEM_USAGE

[69]:
```
training_size = int(len(df) * 0.8)

train = [[i] for i in df["CPU_USAGE"]][:training_size]
test = [[i] for i in df["CPU_USAGE"]][training_size:]

arima = ARIMA(train, order=(LAG,0,0))
arima_fit = arima.fit()
print(arima_fit.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                           y   No. Observations:                 2986
Model:                 ARIMA(12, 0, 0)   Log Likelihood              -11431.464
```

```
Date:                Thu, 25 May 2023   AIC                          22890.929
Time:                        15:01:53   BIC                          22974.952
Sample:                             0   HQIC                         22921.159
                                 - 2986
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         13.9781      0.376     37.143      0.000      13.241      14.716
ar.L1         -0.0007      0.016     -0.045      0.964      -0.031       0.030
ar.L2          0.0124      0.017      0.713      0.476      -0.022       0.046
ar.L3          0.0285      0.017      1.703      0.089      -0.004       0.061
ar.L4         -0.0149      0.018     -0.825      0.410      -0.050       0.021
ar.L5          0.0385      0.018      2.146      0.032       0.003       0.074
ar.L6          0.0415      0.018      2.287      0.022       0.006       0.077
ar.L7         -0.0138      0.019     -0.728      0.467      -0.051       0.023
ar.L8          0.0102      0.018      0.565      0.572      -0.025       0.046
ar.L9         -0.0203      0.018     -1.103      0.270      -0.056       0.016
ar.L10         0.0187      0.018      1.032      0.302      -0.017       0.054
ar.L11         0.0062      0.017      0.370      0.711      -0.026       0.039
ar.L12         0.0071      0.016      0.454      0.650      -0.023       0.038
sigma2       123.8167      3.700     33.463      0.000     116.565     131.069
===================================================================================
===
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
2150.56
Prob(Q):                              1.00   Prob(JB):
0.00
Heteroskedasticity (H):               1.10   Skew:
1.65
Prob(H) (two-sided):                  0.13   Kurtosis:
5.52
===================================================================================
===

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

[70]:
```
residuals = pd.DataFrame(arima_fit.resid)
residuals.plot()
plt.show()
```

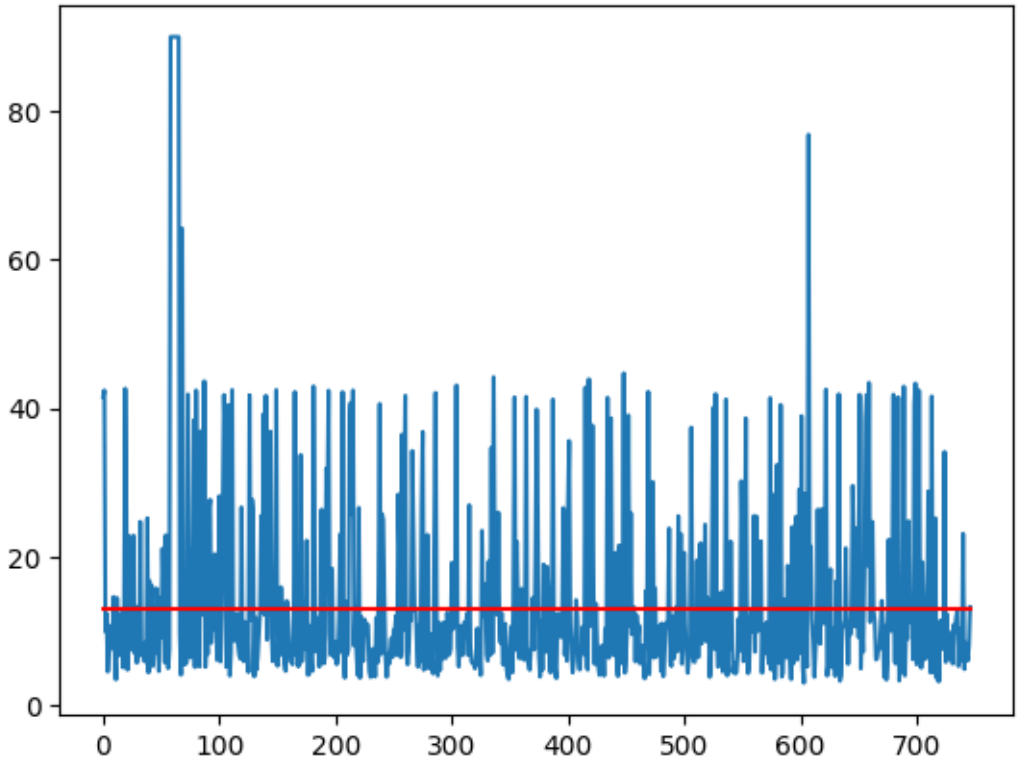


[71]:
```
history = [x for x in train]
predictions = list()
for t in range(len(test)):
    output = arima_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test[t]
    history.append(obs)
    # print('predicted', yhat, obs)
    # print('actual', obs)
```

[72]:
```
# evaluate forecasts
rmse = sqrt(mean_squared_error(test, predictions))
mape = mean_absolute_percentage_error(test, predictions)
print('Test MAPE: %.3f' % mape)
print('Test RMSE: %.3f' % rmse)
# plot forecasts against actual outcomes
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

```
Test MAPE: 76.357
```

```
Test RMSE: 13.574
```



[ ]: