mnist_sin

February 17, 2021

[134]:
```
from mnist import MNIST
from collections import Counter
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
sys.path.insert(0, '../../Libraries')
import JacksonsTSPackage as jts
from statsmodels.tsa.api import VAR
from statsmodels.tsa.ar_model import AutoReg
```

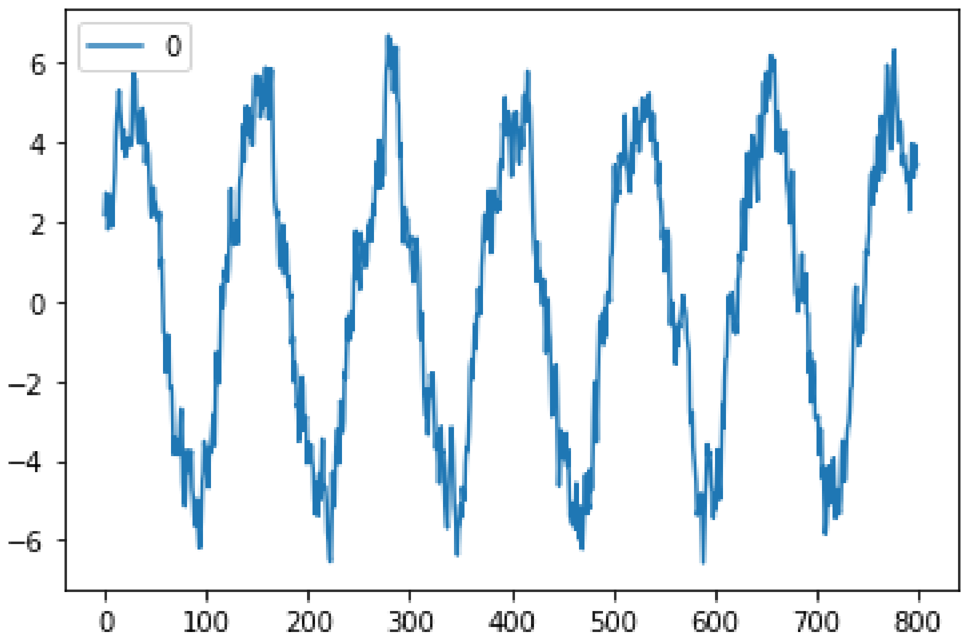
[135]:
```
np.random.seed(123)
data = [np.random.uniform(0, 1) + 5]

N = 800

e = 1
for i in range(1, N + 10):
    data.append(0.8*data[i-1] + np.random.uniform(-e, e) + np.sin(i/ 20))

data = pd.DataFrame(data[10:(N + 10)])
```

[136]:
```
data
```

[136]:
```
            0
0    2.182413
1    2.726717
2    2.623160
3    1.823070
4    1.898763
..        …
795  3.832744
796  3.116977
797  3.945307
798  3.349211
799  3.447519

[800 rows x 1 columns]
```

[137]:
```
data.plot()
```

[137]: <AxesSubplot:>



[138]:
```
N_test = 80
N_train = N - N_test
print(f"N: {N}")
print(f"N_train: {N_train}")
print(f"N_test: {N_test}")
```

```
N: 800
N_train: 720
N_test: 80
```

[139]:
```
rounded_data = data.to_numpy()
min_val = min(rounded_data)
max_val = max(rounded_data)
print(f"Min: {min_val}")
print(f"Max: {max_val}")
```

```
Min: [-6.57611761]
Max: [6.68426652]
```

[140]:
```
rounded_data = [i + 7 for i in rounded_data]
rounded_data = [(i*9)/(max_val - min_val) for i in rounded_data]
```

[141]:
```
rounded_data = [np.round(i) for i in rounded_data]
rounded_data = [int(i) for i in rounded_data]
rounded_data[0:5]
```

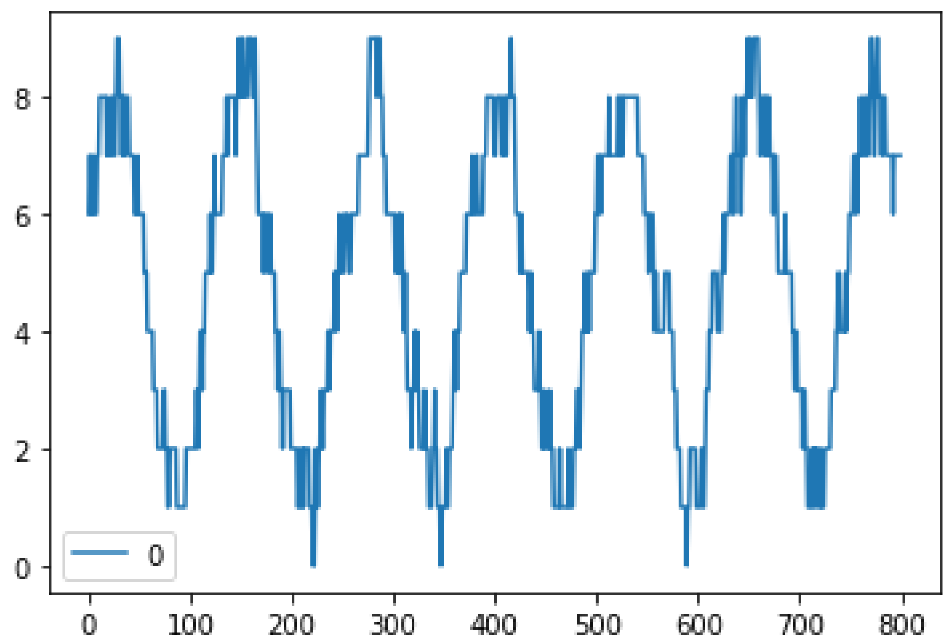
[141]:
```
[6, 7, 7, 6, 6]
```

[142]:
```
Counter(rounded_data)
```

[142]:
```
Counter({6: 105,
         7: 127,
         8: 111,
         9: 33,
         5: 91,
         4: 71,
         3: 71,
         2: 108,
         1: 79,
         0: 4})
```

[143]:
```
pd.DataFrame(rounded_data).plot()
```

[143]: <AxesSubplot:>

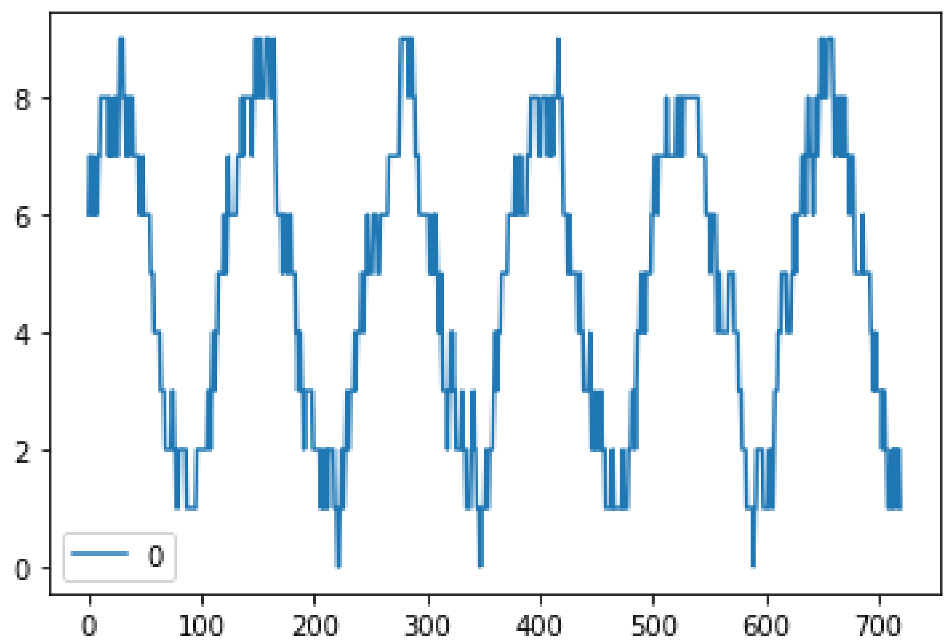


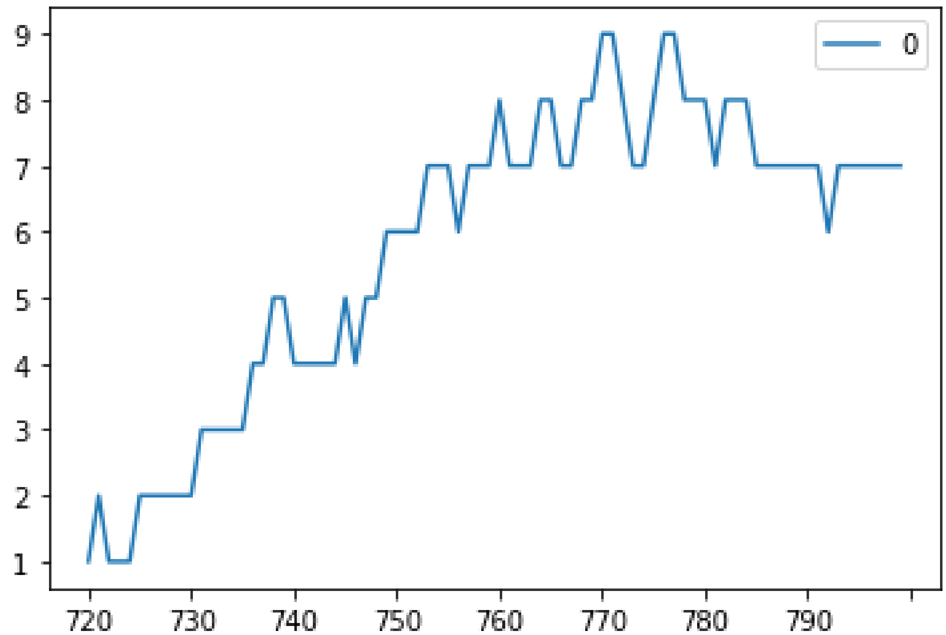
[144]:
```
Counter(rounded_data)
```

[144]: Counter({6: 105,
             7: 127,
             8: 111,
             9: 33,
             5: 91,
             4: 71,
             3: 71,
             2: 108,
             1: 79,
             0: 4})

[145]:
```
train = pd.DataFrame(rounded_data[:N_train])
test = pd.DataFrame(rounded_data[N_train:], index = [str(i) for i in␣
 ↪range(N_train, N)])
train.plot()
test.plot()
```

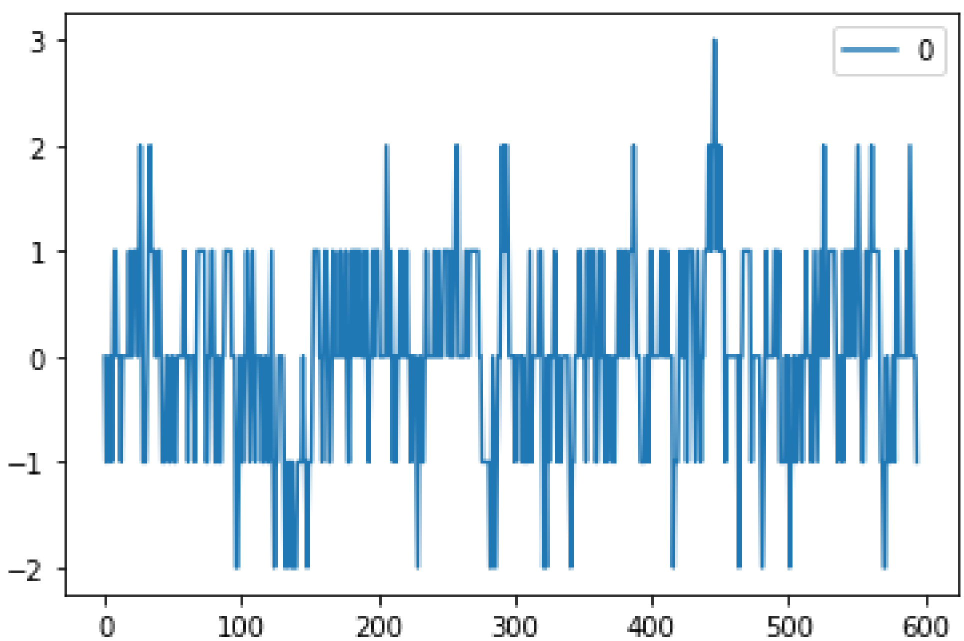
[145]: <AxesSubplot:>





[146]:
```
interval = 125
train_diff = jts.calc_seasonal_diff(train, interval)
train_diff.plot()
```

[146]: <AxesSubplot:>



[194]:
```
jts.mts_adf(train_diff)
```

```
   Augmented Dickey-Fuller Test on "0"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "1"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "2"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "3"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
```

```
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "4"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "5"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3538
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "6"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -5.8745
No. Lags Chosen       = 8
Critical value 1%     = -3.442
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "7"
```

```
   ----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -23.8847
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "8"
   ----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -7.1336
No. Lags Chosen       = 11
Critical value 1%     = -3.442
Critical value 5%     = -2.867
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "9"
   ----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3389
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "10"
   ----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3352
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
```

```
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "11"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3311
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "12"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3311
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "13"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "14"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3311
No. Lags Chosen       = 0
```

```
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "15"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3311
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "16"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.332
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "17"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3716
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "18"
   -----------------------------------------------
```

```
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -24.3721
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "19"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "20"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "21"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.
```

```
   Augmented Dickey-Fuller Test on "22"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "23"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "24"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "25"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
```

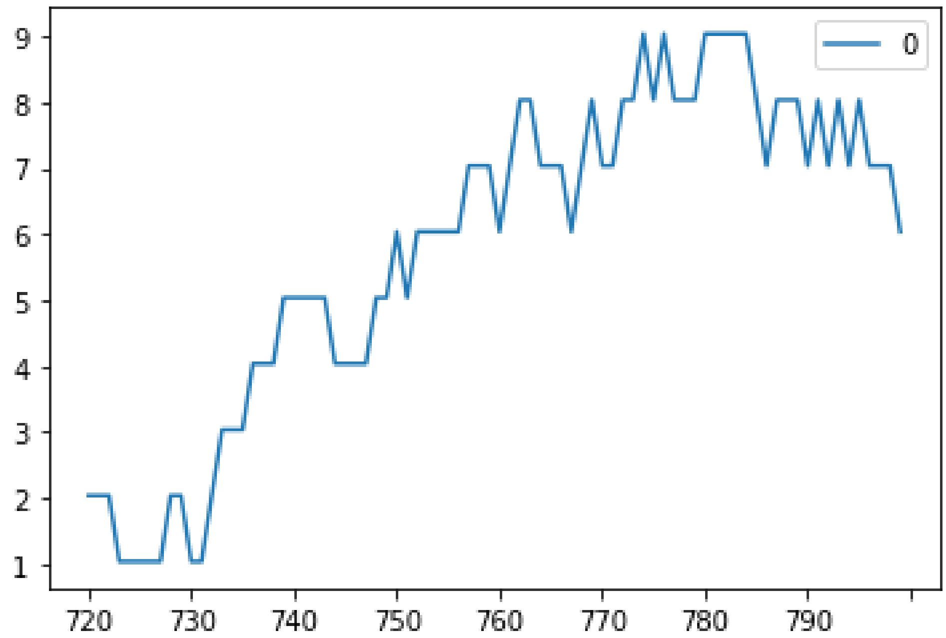
```
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "26"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.


   Augmented Dickey-Fuller Test on "27"
   -----------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = nan
No. Lags Chosen       = 0
Critical value 1%     = -3.441
Critical value 5%     = -2.866
Critical value 10%    = -2.569
=> P-Value = nan. Weak evidence to reject the Null Hypothesis.
=> Series is Non-Stationary.
```

[147]:
```
model = AutoReg(train_diff.values, lags = 1)
fit = model.fit()
pred_diff = pd.DataFrame(fit.predict(start = N_train, end = N - 1), index =
 ↪[str(i) for i in range(N_train, N)])
```

[149]:
```
pred = jts.invert_diff_transformation(pred_diff, train, interval)
```

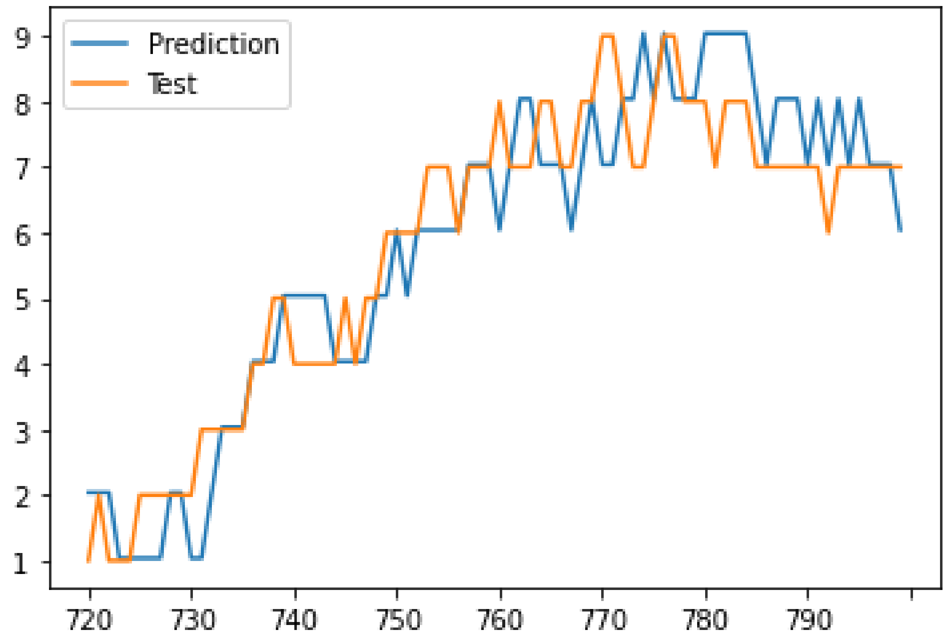
[150]:
```
pred.plot()
```

[150]: <AxesSubplot:>



[151]:
```
pred.columns = ["Prediction"]
test.columns = ["Test"]
df = pd.concat([pred, test], axis = 1)
df.plot()
```

[151]: <AxesSubplot:>



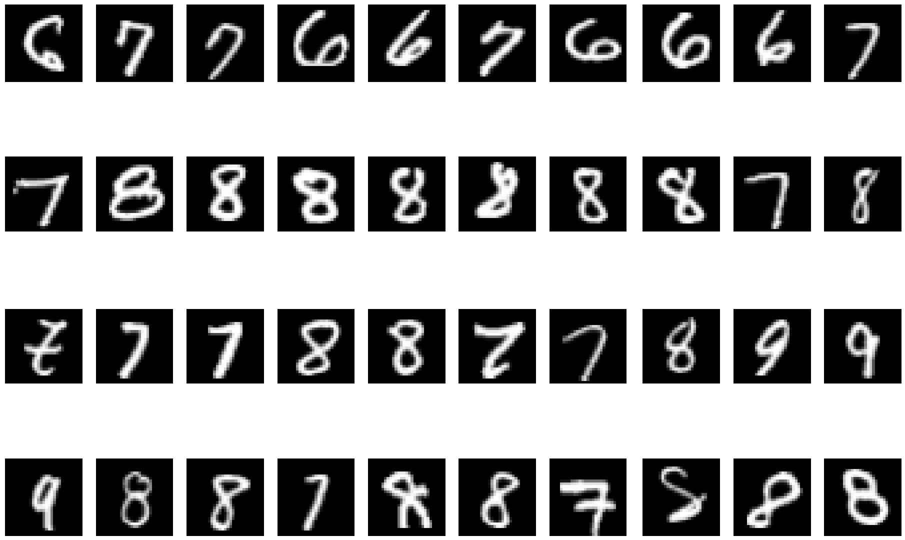
[152]:
```
jts.forecast_accuracy(pred, test)
```

```
Results
----------------------------------------------------------------------
          ME            MSE            MAE            MAPE
Test    0.013          0.800          0.666          14.797%
```

[153]:
```
mndata = MNIST('./Data')
images, labels = mndata.load_training()
labels = labels.tolist()
images = [x for _,x in sorted(zip(labels,images))]
labels.sort()

label_counts = list(Counter(labels).values())

tensor_shape = (N, 28, 28)
data_tensor = np.empty(tensor_shape)

for i in range(N):

    curr_num = rounded_data[i]

    # Pick a random number within that number range
    offset = sum(label_counts[:(curr_num)])
    ran_index = np.random.randint(offset, offset + label_counts[curr_num])

    # Set that random images inside our tensor
    tmp = np.asarray(images[ran_index])
    tmp.resize((tensor_shape[1], tensor_shape[2]))
    data_tensor[i] = tmp
```

[154]:
```
jts.implot_tensor(data_tensor, (4, 10))
np.array(rounded_data[0:40])
```

[154]:
```
array([6, 7, 7, 6, 6, 7, 6, 6, 6, 7, 7, 8, 8, 8, 8, 8, 8, 8, 7, 8, 7, 7,
       7, 8, 8, 7, 7, 8, 9, 9, 9, 8, 8, 7, 8, 8, 7, 8, 8, 8])
```



[155]:
```
train_tensor = jts.extract_train_tensor(data_tensor, N_train)
test_tensor = jts.extract_test_tensor(data_tensor, N_train, N_test)
```

[156]:
```
train_dwt = jts.apply_dwt_to_tensor(train_tensor)
train_dwt
```

[156]:
```
array([[[0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        …,
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.]],

       [[0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        …,
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.]],

       [[0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        …,
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
```

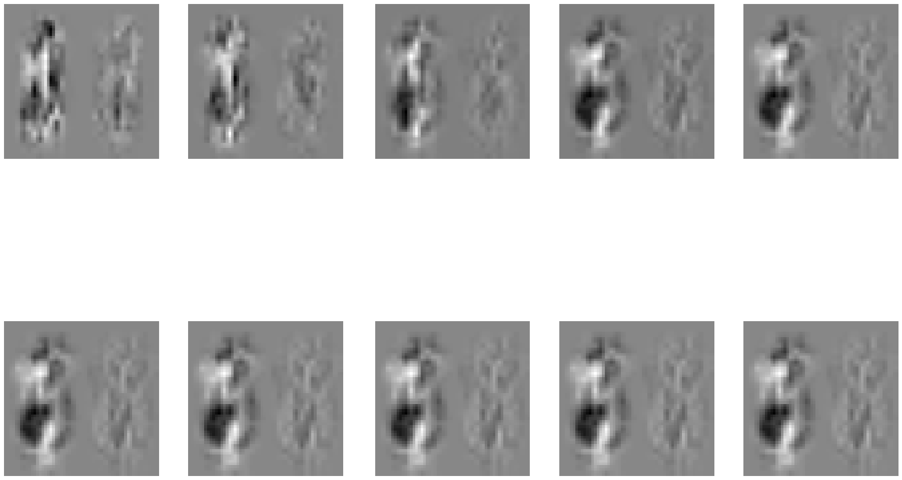
```
        [0., 0., 0., …, 0., 0., 0.]],

       …,

       [[0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        …,
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.]],

       [[0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        …,
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.]],

       [[0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        …,
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.],
        [0., 0., 0., …, 0., 0., 0.]]])
```

[157]:
```
train_model_sets = jts.split_cols_into_model_sets(train_dwt, N_train)
test_model_sets = jts.split_cols_into_model_sets(test_tensor, N_test)
```

[202]:
```
result_model_sets = np.empty((tensor_shape[2], N_test, tensor_shape[1]))
result_model_sets_diff = np.empty((tensor_shape[2], N_test, tensor_shape[1]))

for i in range(28):
    train_df = pd.DataFrame(train_model_sets[i])
    test_df = pd.DataFrame(test_model_sets[i])
    train_diff = jts.calc_seasonal_diff(train_df, interval)
    model = VAR(train_diff)
    fit = model.fit(1)

    test_df.columns = test_df.columns[:].astype(str)

    results_diff = jts.forecast(fit, train_diff, test_df, N_test, calc_conf =␣
 ↪False)

    result_model_sets_diff[i] = results_diff
```

```
    result_model_sets[i] = jts.invert_diff_transformation(results_diff, train_df, interval)
```

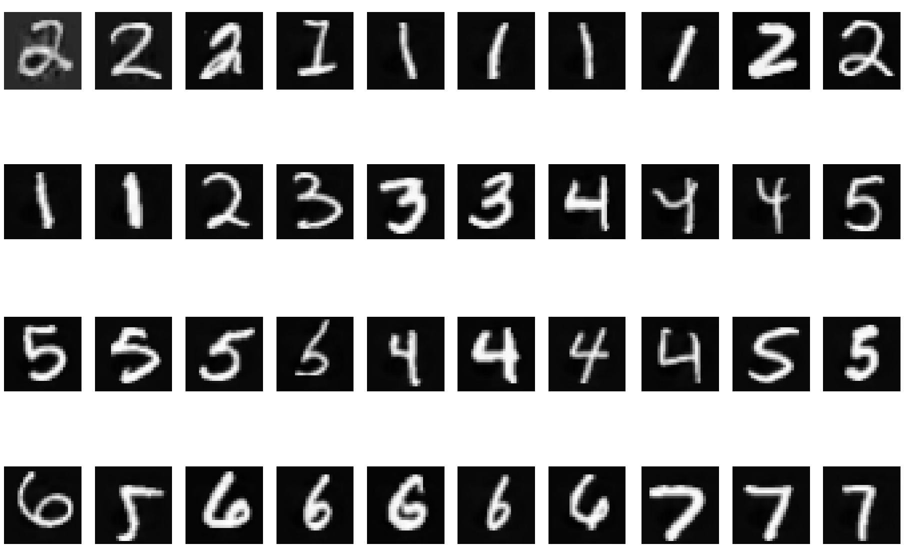
[203]:
```
jts.implot_tensor(jts.collect_result_cols_into_tensor(result_model_sets_diff, N_test), (2, 5))
```



[204]:
```
result_dwt_tensor = jts.collect_result_cols_into_tensor(result_model_sets, N_test)
```

[205]:
```
result_tensor = jts.apply_inverse_dwt_to_tensor(result_dwt_tensor)
```

[206]:
```
jts.implot_tensor(result_tensor, (4, 10))
```



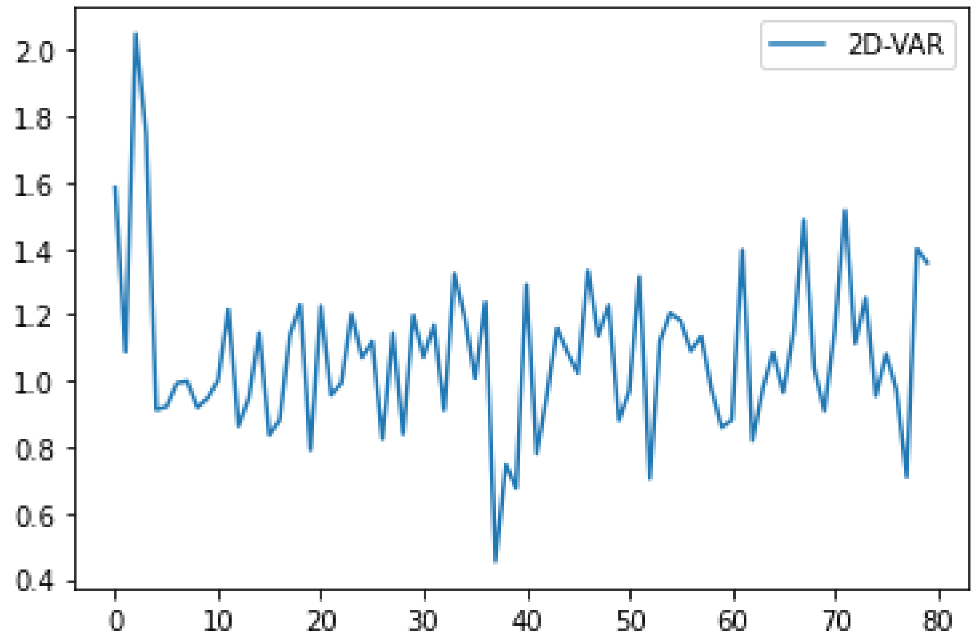
[207]:
```python
jts.implot_tensor(test_tensor, (4, 10))
```

[209]:
```python
error = jts.calc_mape_per_matrix(test_tensor, result_tensor)
error = error.rename(columns={"MAPE": "2D-VAR"})
error
```

[209]:
```
      2D-VAR
0    1.58563
1    1.08817
2    2.05009
3     1.7479
4    0.91321
..       …
75   1.08118
76  0.973078
77  0.709374
78   1.40055
79   1.35728

[80 rows x 1 columns]
```

[210]:
```python
    error.plot()
```

[210]:
```
<AxesSubplot:>
```


2D-VAR
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0
10
20
30
40
50
60
70
80