MANAI MORTADHA _ 3GII/SSE

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
df=pd.read_excel("/content/production.xlsx")
df.head()
```

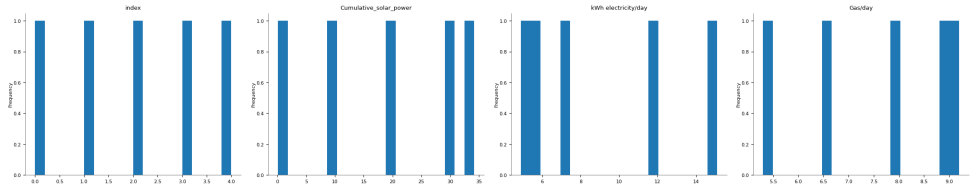
1 to 5 of 5 entries Filter

index	date	Cumulative_solar_power	kWh electricity/day	Gas/day
0	2011-10-26 00:00:00	0.1	15.1	9.0
1	2011-10-27 00:00:00	10.2	7.4	9.2
2	2011-10-28 00:00:00	20.2	5.8	8.0
3	2011-10-29 00:00:00	29.6	4.9	6.6
4	2011-10-30 00:00:00	34.2	11.7	5.3

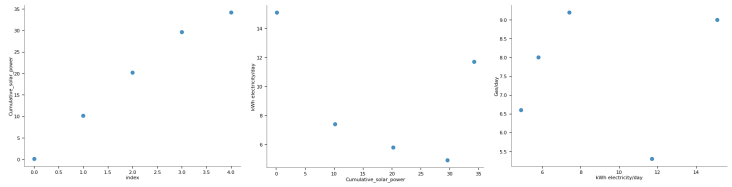
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

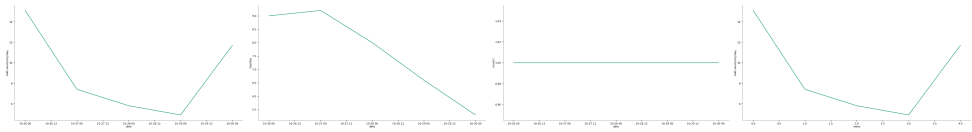
Distributions



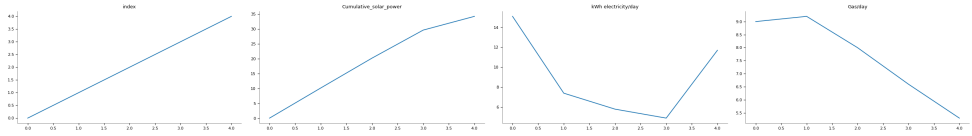
2-d distributions



Time series



Values



```
def load_data(data, seq_len, column_index ):
    x = []
    y = []
    for i in range(seq_len, len(data)):
        x.append(data.iloc[i-seq_len : i, 2:3])
        y.append(data.iloc[i, 2:3])
    return np.array(x), np.array(y)
```

```
# Predict 'kWh electricity/day'
training_size = int(len(df)*0.8)
x, y = load_data(df, 20, 2)

x_train = x[:training_size]
y_train = y[:training_size]
x_test = x[training_size:]
y_test = y[training_size:]

reg1 = LinearRegression().fit(x_train.reshape(x_train.shape[0], -1), y_train)
y_pred1 = reg1.predict(x_test.reshape(x_test.shape[0], -1))


x_train
```

```
array([[[15.1],
        [ 7.4],
        [ 5.8],
        ...,
        [12.9],
        [10.7],
        [10.2]],

       [[ 7.4],
        [ 5.8],
        [ 4.9],
        ...,
        [10.7],
        [10.2],
        [ 7.2]],

       [[ 5.8],
        [ 4.9],
        [11.7],
        ...,
        [10.2],
        [ 7.2],
        [ 9.4]],

       ...,

       [[14. ],
        [18. ],
        [14. ],
        ...,
        [14. ],
        [10. ],
        [18. ]],

       [[18. ],
        [14. ],
        [16. ],
        ...,
        [10. ],
        [18. ],
        [22. ]],

       [[14. ],
        [16. ],
        [11. ],
        ...,
        [18. ],
        [22. ],
        [10. ]]])
```

```
y_train
```

```
array([[7.2],
       [9.4],
       [10.1],
       ...,
       [22.0],
       [10.0],
       [12.0]], dtype=object)
```

```
y_test
```

```
array([[14.0],
       [17.0],
       [8.0],
       [13.0],
       [6.0],
       [14.0],
       [11.0],
       [6.0],
       [6.0],
       [9.0],
```

```
               [9.0],
               [8.0],
               [6.0],
               [15.0],
               [8.0],
               [3.0],
               [10.0],
               [4.0],
               [4.0],
               [7.0],
               [16.0],
               [13.0],
               [15.0],
               [15.0],
               [18.0],
               [7.0],
               [9.0],
               [9.0],
               [15.0],
               [15.0],
               [9.0],
               [10.0],
               [9.0],
               [11.0],
               [12.0],
               [12.0],
               [15.0],
               [4.0],
               [4.0],
               [9.0],
               [9.0],
               [-6.0],
               [-4.0],
               [15.0],
               [-11.0],
               [9.0],
               [6.0],
               [6.0],
               [8.0],
               [-4.0],
               [1.0],
               [1.0],
               [-3.0],
               [10.0],
               [2.0],
               [7.0],
               [3.0],
               [3.0]
```

```
x_test
```

```
    array([[[16.],
            [11.],
            [11.],
            ...,
            [22.],
            [10.],
            [12.]],

           [[11.],
            [11.],
            [13.],
            ...,
            [10.],
            [12.],
            [14.]],

           [[11.],
            [13.],
            [18.],
            ...,
            [12.],
            [14.],
            [17.]],

           ...,

           [[16.],
            [16.],
            [20.],
            ...,
            [12.],
            [16.],
            [13.]],

           [[16.],
            [20.],
            [12.],
            ...,
            [16.],
```

```
       [13.],
       [12.]],

      [[20.],
       [12.],
       [16.],
       ...,
       [13.],
       [12.],
       [14.]]])
```

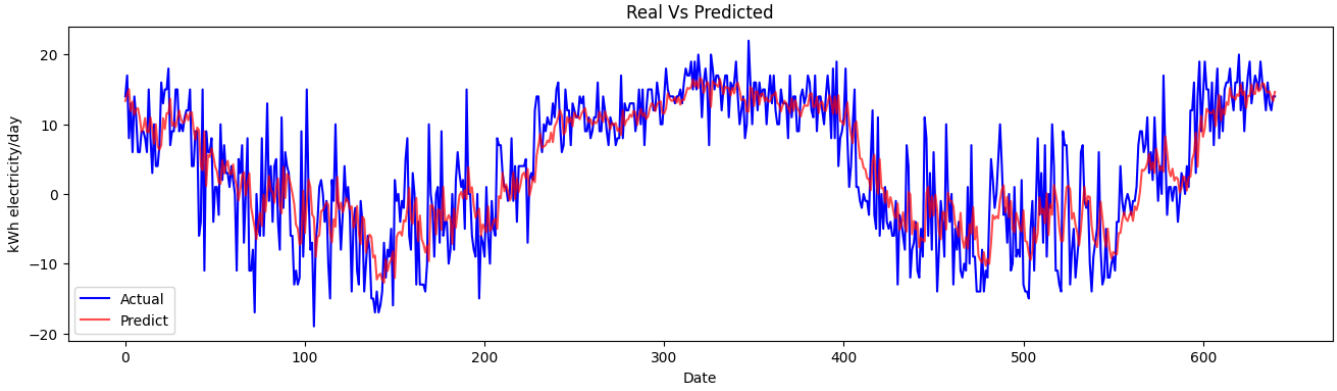
reg1

```
▾ LinearRegression
LinearRegression()
```

y_pred1

```
      [ 3.62098581e+00],
      [ 1.91026746e+00],
      [ 2.40027304e+00],
      [ 2.16577432e+00],
      [ 2.40336563e-01],
      [ 8.52991045e-01],
      [ 2.52350784e+00],
      [ 2.32390663e+00],
      [ 4.98821845e-01],
      [ 2.42067474e+00],
      [ 2.45963131e+00],
      [ 4.93812823e+00],
      [ 5.84545644e+00],
      [ 8.79538821e+00],
      [ 3.91038243e+00],
      [ 9.05967292e+00],
      [ 1.11950132e+01],
      [ 8.20278037e+00],
      [ 9.43399312e+00],
      [ 1.22218256e+01],
      [ 1.19077032e+01],
      [ 1.15679631e+01],
      [ 1.16073601e+01],
      [ 1.20680307e+01],
      [ 9.72389969e+00],
      [ 1.19512626e+01],
      [ 1.42190531e+01],
      [ 1.05477529e+01],
      [ 1.22008451e+01],
      [ 1.14020681e+01],
      [ 1.31670009e+01],
      [ 1.27755084e+01],
      [ 1.52104625e+01],
      [ 1.38349320e+01],
      [ 1.39995891e+01],
      [ 1.35244334e+01],
      [ 1.42931830e+01],
      [ 1.40732407e+01],
      [ 1.57183736e+01],
      [ 1.42102248e+01],
      [ 1.47103749e+01],
      [ 1.23828681e+01],
      [ 1.40795039e+01],
      [ 1.40794599e+01],
      [ 1.57347255e+01],
      [ 1.43949271e+01],
      [ 1.43301047e+01],
      [ 1.58168360e+01],
      [ 1.47303019e+01],
      [ 1.49774409e+01],
      [ 1.59162395e+01],
      [ 1.57625536e+01],
      [ 1.49166149e+01],
      [ 1.45190558e+01],
      [ 1.47240866e+01],
      [ 1.39581383e+01],
      [ 1.33251799e+01],
      [ 1.46322449e+01]])
```

```
plt.figure(figsize=(16,4))
plt.plot(y_test, color='blue',label='Actual')
plt.plot(y_pred1, alpha=0.7, color='red',label='Predict')
plt.title('Real Vs Predicted')
plt.xlabel('Date')
plt.ylabel('kWh electricity/day')
plt.legend()
plt.show()
```




```
# Predict 'Gas/day'
x, y = load_data(df, 20, 3)

x_train = x[:training_size]
y_train = y[:training_size]
x_test = x[training_size:]
y_test = y[training_size:]

reg2 = LinearRegression().fit(x_train.reshape(x_train.shape[0], -1), y_train)
y_pred2 = reg2.predict(x_test.reshape(x_test.shape[0], -1))
```

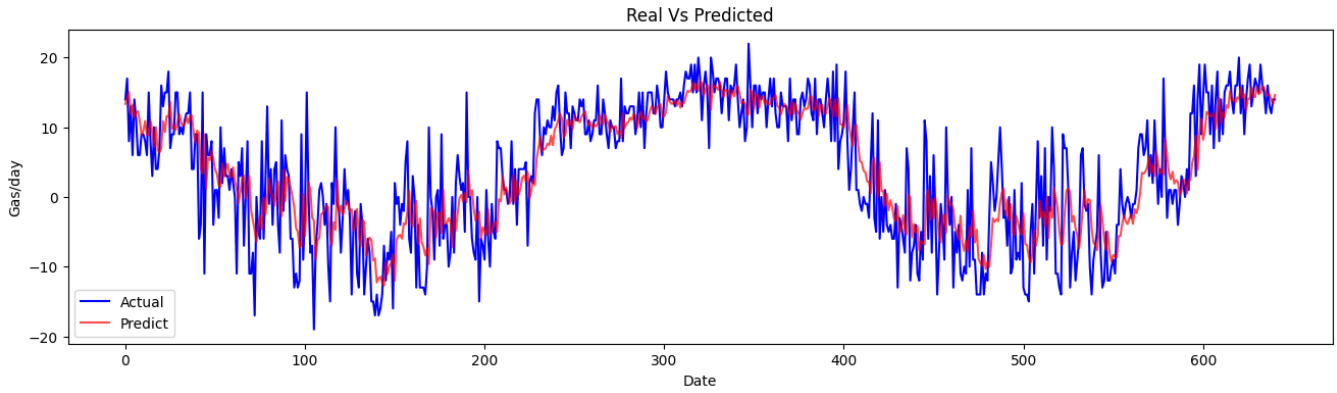
```
reg2
```

▾ LinearRegression
LinearRegression()

```
y_pred2
```

```
[ 1.35244534e+01],
[ 1.42931830e+01],
[ 1.40732407e+01],
[ 1.57183736e+01],
[ 1.42102248e+01],
[ 1.47103749e+01],
[ 1.23828681e+01],
[ 1.40795039e+01],
[ 1.40794599e+01],
[ 1.57347255e+01],
[ 1.43949271e+01],
[ 1.43301047e+01],
[ 1.58168360e+01],
[ 1.47303019e+01],
[ 1.49774409e+01],
[ 1.59162395e+01],
[ 1.57625536e+01],
[ 1.49166149e+01],
[ 1.45190558e+01],
[ 1.47240866e+01],
[ 1.39581383e+01],
[ 1.33251799e+01],
[ 1.46322449e+01]])
```

```
plt.figure(figsize=(16,4))
plt.plot(y_test, color='blue',label='Actual')
plt.plot(y_pred2, alpha=0.7, color='red',label='Predict')
plt.title('Real Vs Predicted')
plt.xlabel('Date')
plt.ylabel('Gas/day')
plt.legend()
plt.show()
```




```
rmse = np.sqrt(mean_squared_error(y_test, y_pred1)).round(2)
# Exclude zero values
mask = y_test != 0
mape = np.round(np.mean(np.abs(y_test[mask]-y_pred1[mask])/y_test[mask])*100,2)

#mape = np.round(np.mean(np.abs(y_test-y_pred)/y_test)*100,2)

print('RMSE:', rmse)
print('MAPE:', mape)
```

```
RMSE: 5.66
MAPE: 7.12
```

```
rmse = np.sqrt(mean_squared_error(y_test, y_pred2)).round(2)
# Exclude zero values
mask = y_test != 0
mape = np.round(np.mean(np.abs(y_test[mask]-y_pred2[mask])/y_test[mask])*100,2)

#mape = np.round(np.mean(np.abs(y_test-y_pred)/y_test)*100,2)

print('RMSE:', rmse)
print('MAPE:', mape)
```

```
RMSE: 5.66
MAPE: 7.12
```