```
import math
import numpy as np
import pandas as pd
from datetime import date,timedelta,datetime
from pandas.plotting import register_matplotlib_converters
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import tensorflow as tf
from sklearn.metrics import mean_absolute_error, mean_squared_error
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.preprocessing import RobustScaler, MinMaxScaler,StandardScaler
import seaborn as sns
sns.set_style('white', { 'axes.spines.right': False, 'axes.spines.top': False})
```

```
from google.colab import drive
drive.mount('/content/drive/')
```

```
     Mounted at /content/drive/
```

```
df=pd.read_csv('/content/drive/MyDrive/load_forecast_data/training_data - Week 21, May 2019.csv')
```

```
df.head()
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

								holiday	Hol
0	2015-01-31 01:00:00	962.2865	906.9580	970.3450	938.004850	1	1	0	
1	2015-01-31 02:00:00	933.3221	863.5135	912.1755	900.284075	1	1	0	

```
df.set_index('datetime')
```

	week_X-2	week_X-3	week_X-4	MA_X-4	dayOfWeek	weekend	holiday	Ho
datetime								
2015-01-31 01:00:00	962.2865	906.9580	970.3450	938.004850	1	1	0	
2015-01-31 02:00:00	933.3221	863.5135	912.1755	900.284075	1	1	0	
2015-01-31 03:00:00	903.9817	848.4447	900.2688	881.704325	1	1	0	
2015-01-31 04:00:00	900.9995	839.8821	889.9538	876.458825	1	1	0	
2015-01-31 05:00:00	904.3481	847.1073	893.6865	879.190775	1	1	0	
...	...	...	...	...	...	...	...	

```
df_plot = df.copy()
ncols=2
nrows=int(round(df_plot.shape[1] / ncols, 0))
fig, ax = plt.subplots(nrows=nrows,ncols=ncols,sharex=True, figsize=(14, 7))
for i, ax in enumerate(fig.axes):
        sns.lineplot(data = df_plot.iloc[:, i], ax=ax)
        ax.tick_params(axis="x", rotation=30, labelsize=10, length=0)
        ax.xaxis.set_major_locator(mdates.AutoDateLocator())
        ax.title.set_text(df_plot.columns[i])
fig.tight_layout()
plt.show()
```

```
KeyboardInterrupt                         Traceback (most recent call last)
```

13 frames

```
/usr/local/lib/python3.10/dist-packages/dateutil/parser/_parser.py in
_to_decimal(self, val)
   1141     def _to_decimal(self, val):
   1142         try:
-> 1143             decimal_value = Decimal(val)
   1144             # See GH 662, edge case, infinite value should not be
converted
   1145             #  via `_to_decimal`

KeyboardInterrupt:
```

SEARCH STACK OVERFLOW

```
Error in callback <function _draw_all_if_interactive at 0x7f7f8aae1000> (for post_
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/matplotlib/pyplot.py in
_draw_all_if_interactive()
    118 def _draw_all_if_interactive():
    119     if matplotlib.is_interactive():
--> 120         draw_all()
    121
    122
```

18 frames

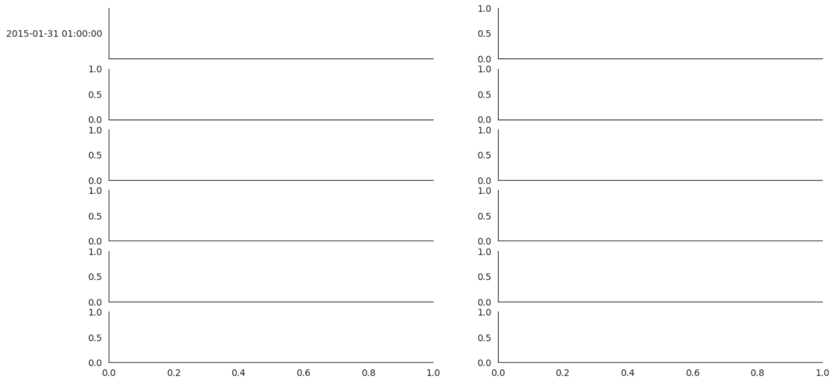
```
/usr/lib/python3.10/abc.py in __instancecheck__(cls, instance)
    115             return _abc_register(cls, subclass)
    116
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
    119             return _abc_instancecheck(cls, instance)

KeyboardInterrupt:
```

SEARCH STACK OVERFLOW



```
df.set_index('datetime')[['week_X-2','']]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-7-ca72d950a981> in <cell line: 1>()
----> 1 df.set_index('datetime')[['week_X-2','']]
```

```
cols = list(df)[1:]
print(cols)
```

```
['week_X-2', 'week_X-3', 'week_X-4', 'MA_X-4', 'dayOfWeek', 'weekend', 'holiday', 'Holiday_ID', 'hourOfDay', 'T2M_toc', 'DEMAND']
 6132            not_found = list(ensure_index(key)[missing_mask.nonzero()]
```

```
def prepare_data(df):
  FEATURES=['week_X-2', 'week_X-3', 'week_X-4', 'MA_X-4', 'dayOfWeek', 'weekend', 'holiday', 'Holiday_ID', 'hourOfDay', 'T2M_toc', 'DEMAN

  print('Feature list')

  print([f for f in FEATURES])

  df_filter=df[FEATURES]

  np_filter_unscaled=np.array(df_filter)

  print(np_filter_unscaled.shape)

  np_c_scaled = np.array(df['DEMAND']).reshape(-1,1)

  return np_filter_unscaled,np_c_scaled,df_filter
```

```
_X-4', 'dayOfWeek', 'weekend', 'holiday', 'Holiday_ID', 'hour
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
df_train.head()
```

	week_X-2	week_X-3	week_X-4	MA_X-4	dayOfWeek	weekend	holiday	Holida
datetime								
2015-01-31 01:00:00	962.2865	906.9580	970.3450	938.004850	1	1	0	
2015-01-31 02:00:00	933.3221	863.5135	912.1755	900.284075	1	1	0	

```
np_filter_unscaled, np_c_unscaled, df_filter = prepare_data(df_train)
```

```
Feature list
['week_X-2', 'week_X-3', 'week_X-4', 'MA_X-4', 'dayOfWeek', 'weekend', 'holiday', 'Holiday_ID', 'hourOfDay', 'T2M_toc', 'DEMAND']
(37728, 11)
```

```
scaler_train = MinMaxScaler()
np_scaled = scaler_train.fit_transform(np_filter_unscaled)

scaler_pred = MinMaxScaler()
np_scaled_c = scaler_pred.fit_transform(np_c_unscaled)
```

```
df_train
```

```
                 week_X-2  week_X-3  week_X-4      MA_X-4  dayOfWeek  weekend  holiday  Ho
```

```
input_sequence_length = 168
output_sequence_length = 24
index_Demand = df_filter.columns.get_loc("DEMAND")
```

```
df_filter.columns.get_loc("DEMAND")
```

```
10
```

```
df_filter
```

	week_X-2	week_X-3	week_X-4	MA_X-4	dayOfWeek	weekend	holiday	Ho
datetime								
2015-01-31 01:00:00	962.2865	906.9580	970.3450	938.004850	1	1	0	
2015-01-31 02:00:00	933.3221	863.5135	912.1755	900.284075	1	1	0	
2015-01-31 03:00:00	903.9817	848.4447	900.2688	881.704325	1	1	0	
31 04:00:00	900.9995	839.8821	889.9558	876.458825	1	1	0	
2015-01-31 05:00:00	904.3481	847.1073	893.6865	879.190775	1	1	0	
...	...	...	...	...	...	...	...	

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
train_data=np_scaled[:36720:]
test_data=np_scaled[36552:36889:]



def partition_dataset(input_sequence_length,output_sequence_length,data):
  x,y=[],[]
  data_len = data.shape[0]
  gap = 48

  for i in range(input_sequence_length,data_len-output_sequence_length,gap):
    x.append(data[i-input_sequence_length:i,:])
    y.append(data[i:i+output_sequence_length,index_Demand])


  x = np.array(x);
  y = np.array(y);

  return x,y


x_train,y_train = partition_dataset(input_sequence_length,output_sequence_length,train_data)
x_test,y_test = partition_dataset(input_sequence_length,output_sequence_length,test_data)

print(x_train.shape,y_train.shape)

# nrows=3

# fig,ax = plt.subplots(nrows=nrows,ncols=1,figsize=(16,8))

# for i,ax in enumerate(fig.axes):
#   xtrain = pd.DataFrame(x_train[i][:,index_Demand],columns={f'x_train_{i}'})
#   ytrain = pd.DataFrame(y_train[i][:output_sequence_length-1],columns={f'y_train_{i}'})
#   ytrain.index = np.arange(input_sequence_length,input_sequence_length+output_sequence_length-1)
#   xtrain_=pd.concat([xtrain,ytrain[:1].rename(columns={ytrain.columns[0]:xtrain.columns[0]})])
#   df_merge = pd.concat([xtrain_,ytrain])
#   sns.lineplot(data=df_merge,ax=ax)

# plt.show
```

```
(761, 168, 11) (761, 24)
```

```
# Model Training

model = Sequential()
n_output_neurons = output_sequence_length

n_input_neurons = x_train.shape[1]*x_train.shape[2]

print(n_input_neurons,x_train.shape[1],x_train.shape[2])

model.add(LSTM(n_input_neurons,return_sequences=True,input_shape=(x_train.shape[1],x_train.shape[2])))
model.add(LSTM(n_input_neurons,return_sequences=False))
model.add(Dense(5))
model.add(Dense(n_output_neurons))

model.compile(optimizer='adam',loss='mse')
```

```
1848 168 11
```

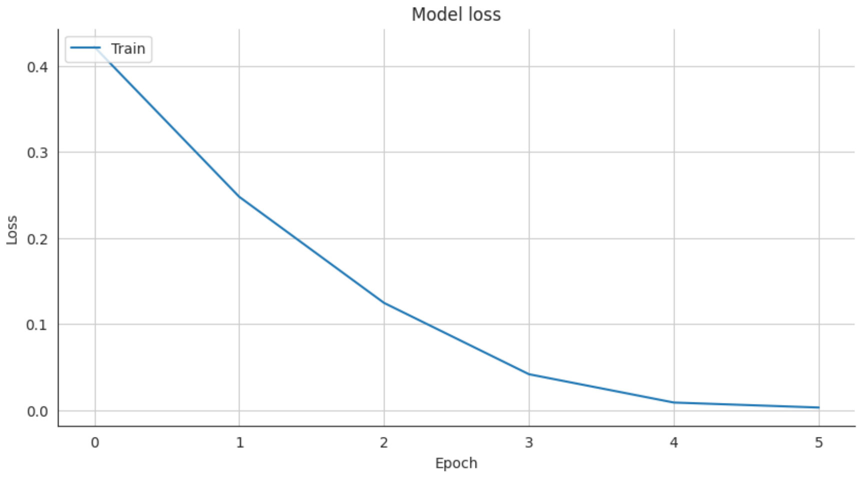
```
epochs = 6
batch_size = 16
early_stop = EarlyStopping(monitor="loss",patience=1,verbose=1)
history = model.fit(x_train,y_train,
                    batch_size=batch_size,
                    epochs=epochs)
```

```
Epoch 1/6
48/48 [==============================] - 2157s 45s/step - loss: 0.4223
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
48/48 [==============================] - 2117s 44s/step - loss: 0.1247
Epoch 4/6
48/48 [==============================] - 2131s 44s/step - loss: 0.0419
Epoch 5/6
48/48 [==============================] - 2124s 44s/step - loss: 0.0090
Epoch 6/6
48/48 [==============================] - 2109s 44s/step - loss: 0.0032
```

```
fig, ax = plt.subplots(figsize=(10, 5), sharex=True)
plt.plot(history.history["loss"])
plt.title("Model loss")
plt.ylabel("Loss")
plt.xlabel("Epoch")
ax.xaxis.set_major_locator(plt.MaxNLocator(epochs))
plt.legend(["Train"], loc="upper left")
plt.grid()
plt.show()
```



```
# Get the predicted values
y_pred_scaled = model.predict(x_test)

# Unscale the predicted values
y_pred = scaler_pred.inverse_transform(y_pred_scaled)
y_test_unscaled = scaler_pred.inverse_transform(y_test).reshape(-1, output_sequence_length)
```

```
# Mean Absolute Error (MAE)
MAE = mean_absolute_error(y_test_unscaled, y_pred)
print(f'Median Absolute Error (MAE): {np.round(MAE, 2)}')


# Mean Absolute Percentage Error (MAPE)
MAPE = np.mean((np.abs(np.subtract(y_test_unscaled, y_pred)/ y_test_unscaled))) * 100
print(f'Mean Absolute Percentage Error (MAPE): {np.round(MAPE, 2)} %')


# Median Absolute Percentage Error (MDAPE)
MDAPE = np.median((np.abs(np.subtract(y_test_unscaled, y_pred)/ y_test_unscaled)) ) * 100
print(f'Median Absolute Percentage Error (MDAPE): {np.round(MDAPE, 2)} %')


def prepare_df(i, x, y, y_pred_unscaled):
    # Undo the scaling on x, reshape the testset into a one-dimensional array, so that it fits to the pred scaler
    x_test_unscaled_df = pd.DataFrame(scaler_pred.inverse_transform((x[i]))[:,index_Demand]).rename(columns={0:'x_test'})

    y_test_unscaled_df = []
    # Undo the scaling on y
    if type(y) == np.ndarray:
        y_test_unscaled_df = pd.DataFrame(scaler_pred.inverse_transform(y)[i]).rename(columns={0:'y_test'})

    # Create a dataframe for the y_pred at position i, y_pred is already unscaled
    y_pred_df = pd.DataFrame(y_pred_unscaled[i]).rename(columns={0:'y_pred'})
    return x_test_unscaled_df, y_pred_df, y_test_unscaled_df
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
def plot_multi_test_forecast(x_test_unscaled_df, y_test_unscaled_df, y_pred_df, title):
    # Package y_pred_unscaled and y_test_unscaled into a dataframe with columns pred and true
    if type(y_test_unscaled_df) == pd.core.frame.DataFrame:
        df_merge = y_pred_df.join(y_test_unscaled_df, how='left')
    else:
        df_merge = y_pred_df.copy()

    # Merge the dataframes
    df_merge_ = pd.concat([x_test_unscaled_df, df_merge]).reset_index(drop=True)

    # Plot the linecharts
    fig, ax = plt.subplots(figsize=(20, 8))
    plt.title(title, fontsize=12)
    ax.set(ylabel = "DEMAND")
    sns.lineplot(data = df_merge_, linewidth=2.0, ax=ax)


# Creates a linechart for a specific test batch_number and corresponding test predictions

# x_test_unscaled_df, y_pred_df, y_test_unscaled_df = prepare_df(i, x_test, y_test, y_pred)
# title = "Forecasting Demand Price"
# plot_multi_test_forecast(x_test_unscaled_df, y_test_unscaled_df, y_pred_df, title)
```

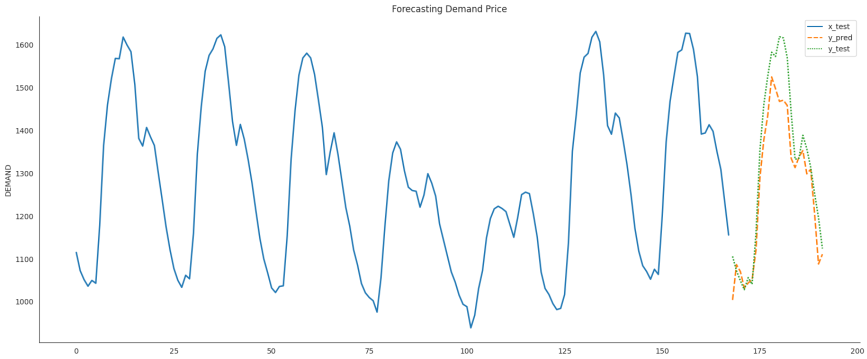
```
    1/1 [==============================] - 6s 6s/step
    Median Absolute Error (MAE): 96.67
    Mean Absolute Percentage Error (MAPE): 7.2 %
    Median Absolute Percentage Error (MDAPE): 6.7 %
```

```
x_test.shape
```

```
    (4, 168, 11)
```

```
x_test_unscaled_df, y_pred_df, y_test_unscaled_df = prepare_df(0, x_test, y_test, y_pred)
```

```
title = "Forecasting Demand Price"
plot_multi_test_forecast(x_test_unscaled_df, y_test_unscaled_df, y_pred_df, title)
```

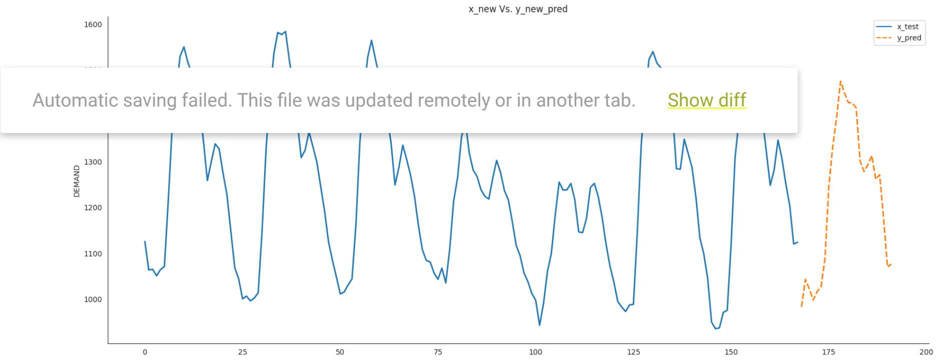


```
x_test_latest_batch = np_scaled[-168:,:].reshape(1,168,11)

# Predict on the batch
y_pred_scaled = model.predict(x_test_latest_batch)
y_pred_unscaled = scaler_pred.inverse_transform(y_pred_scaled)

# Prepare the data and plot the input data and the predictions
x_test_unscaled_df, y_test_unscaled_df, _ = prepare_df(0, x_test_latest_batch, '', y_pred_unscaled)
plot_multi_test_forecast(x_test_unscaled_df, '', y_test_unscaled_df, "x_new Vs. y_new_pred")
```

```
    1/1 [==============================] - 3s 3s/step
```



Automatic saving failed. This file was updated remotely or in another tab. Show diff