Assignment: Time Series Forecasting with PyTorch

Here Sample of data :

```
df.head()
```

	Date Time	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	wv (m/s)	max. wv (m/s)	wd (deg)
0	01.01.2009 00:10:00	996.52	-8.02	265.40	-8.90	93.3	3.33	3.11	0.22	1.94	3.12	1307.75	1.03	1.75	152.3
1	01.01.2009 00:20:00	996.57	-8.41	265.01	-9.28	93.4	3.23	3.02	0.21	1.89	3.03	1309.80	0.72	1.50	136.1
2	01.01.2009 00:30:00	996.53	-8.51	264.91	-9.31	93.9	3.21	3.01	0.20	1.88	3.02	1310.24	0.19	0.63	171.6
3	01.01.2009 00:40:00	996.51	-8.31	265.12	-9.07	94.2	3.26	3.07	0.19	1.92	3.08	1309.19	0.34	0.50	198.0
4	01.01.2009 00:50:00	996.51	-8.27	265.15	-9.04	94.1	3.27	3.08	0.19	1.92	3.09	1309.00	0.32	0.63	214.3

1 – Data Preprocessing:

- **Here we see the statistics of the dataset, and in `wv(m/s)` and `max. wv (m/s)` it appears that the minimum value = -9999.000 and it's likely erroneous so I will replace all values that lower than zero to zero**

wv (m/s)	max. wv (m/s)
420551.000000	420551.000000
1.702224	3.056555
65.446714	69.016932
-9999.000000	-9999.000000
0.990000	1.760000
1.760000	2.960000
2.860000	4.740000
28.490000	23.500000

- As i see that the angles don't make a good model so we can make
  (I) Categorical Encoding by converting each direction to ("East", "north"...)
  (II) Circular Encoding: one for the sine and one for the cosine of the angle

  So here to make it more meaningful and interpretable way for a machine learning model i will convert wind direction and velocity columns to a wind vector using circular encoding

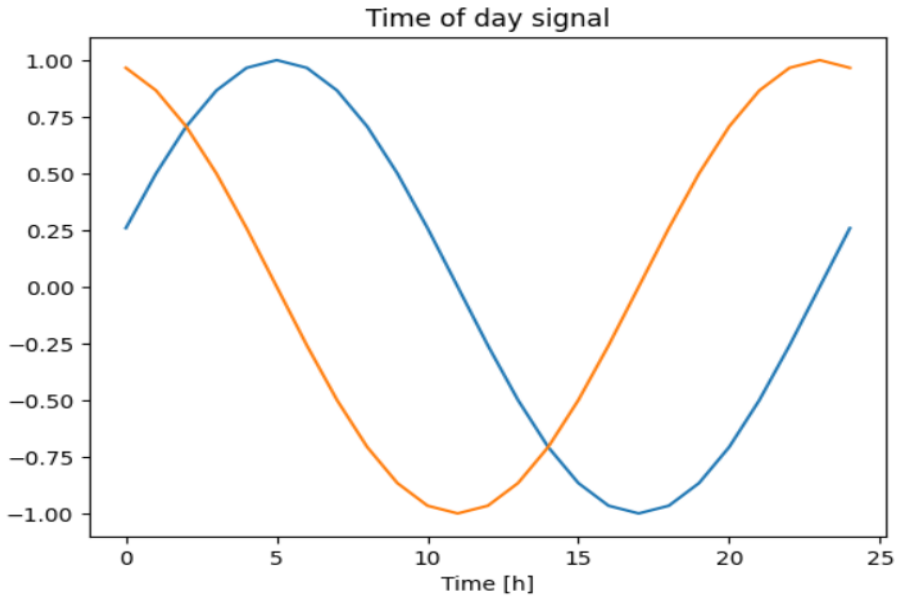
- `Year sin` and `Year cos` capture the time of year periodicity. Like the daily components, these features represent the cyclic nature of the year. They allow

the model to capture seasonal patterns in the data.

- for handling periodicity in time-related features, specifically for capturing `hourly, daily, and yearly` patterns using sine and cosine transforms

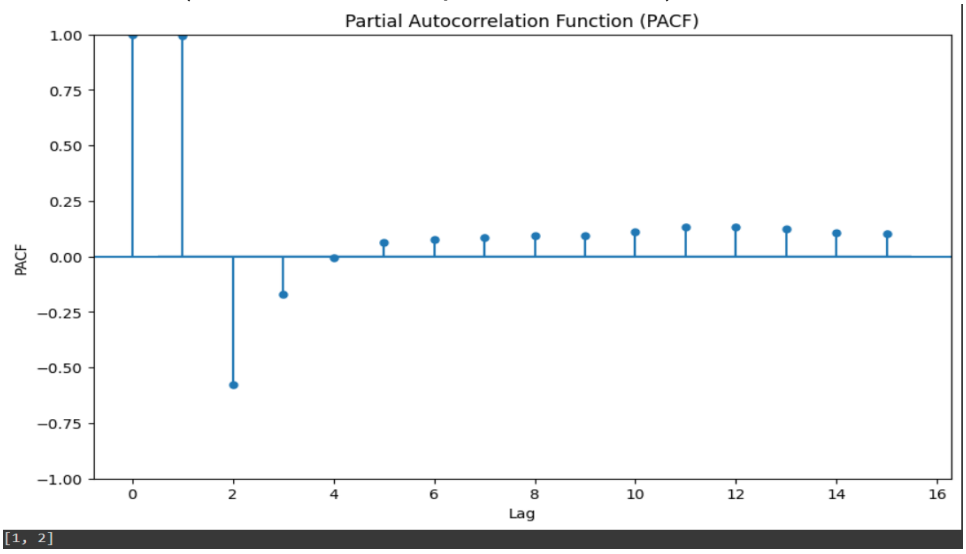
```
plt.plot(np.array(df['Day sin'])[:25])
plt.plot(np.array(df['Day cos'])[:25])
plt.xlabel('Time [h]')
plt.title('Time of day signal')

--NORMAL--
Text(0.5, 1.0, 'Time of day signal')
```



2 – Model Architecture:

- Based on Partial Autocorrelation Function (PACF) plot, I decided to choose lags as a feature (absolute value of pacf value > 0.2)




```
[1, 2]
```

- I decided to choose window moving average feature = 5 based on my dataset I choose the best moving average

- The WindowGenerator class appears to be a utility class designed to generate input-output pairs (windows) from time series data, which is particularly useful when working with recurrent neural networks (RNNs). This is crucial for training a model to predict future values based on historical observations

3 – Training and Evaluation:

- Choose these hyperparameters as default parameters
  - `hidden size = 42`
  - `output size = 1`
  - `batch size = 32`
  - `learning rate = 0.001`
  - `Num of epochs = 10`
- `Start training the model using these parameters`

4 – Hyperparameters Tuning:

- Choose these hyperparameters to be tuned
  - `learning rates = [0.01, 0.001]`
  - `batch sizes = [32, 64]`
  - `hidden sizes = [64, 128]`
- Start get the loss for each 3 pairs of these parameters and get the best parameters and get the Best Hyperparameters: {'lr': 0.01, 'batch_size': 32, 'hidden_size': 64}