# Final Assignment - Specialized Models: Time Series and Survival Analysis - IBM Machine Learning Professional Certificate

## Main Objective

A company that **generates electrical energy** by different means wants to **optimize** its **resources**. One of the means that it has to generate electrical energy are **windmills** that have been operating for last three years. We have been asked to build a model that is **able to predict** how much **energy** the windmills will **generate in the next days**. The generation of energy using windmills depends on several factors and they will provide us with data from these years, so we have decided to use **Deep Learning algorithms to solve this problem** taking a step beyond more traditional techniques such as ARIMA. A **good prediction will help the company to schedule the energy production** of other sources to complete it to avoid power failure due to consumer demand. Thus, they can optimize their resources by **saving unnecessary spending on other sources, which also has the advantage for the environment of optimizing production using non-ecological sources as little as possible**.

## Brief Description of the Data

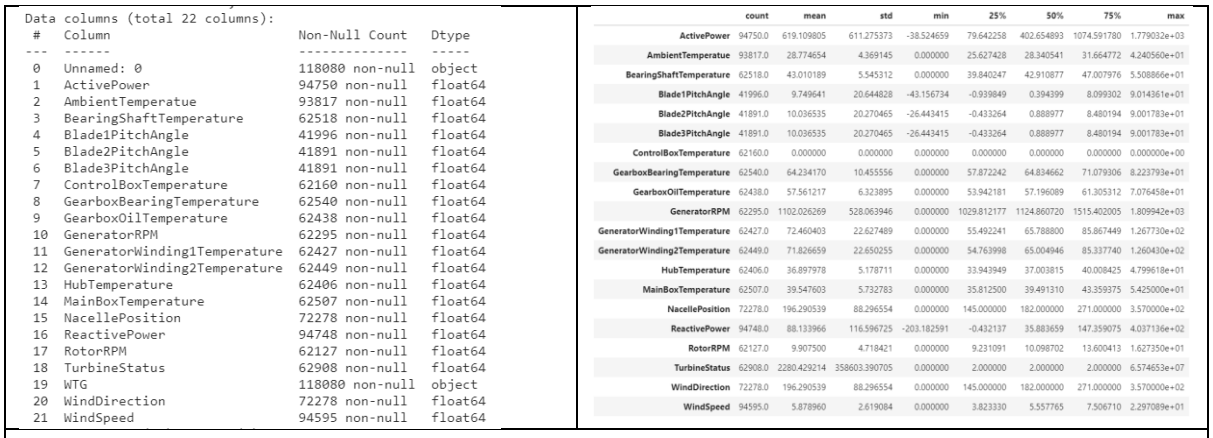
For this purpose, the company has provided us with a dataset of the generation of energy by a windmill at one of its stations with data from the last three years. The data set is a time series that collects measurements **from January 1, 2018 to March 30, 2020 taken every 10 minutes**, then each of the **118080 rows** of which the data set is composed represents one of these measured in time. Furthermore, the data set is made up of **22 columns that represent the different characteristics related to energy production**. Characteristics are the following (**figure 1)**:

- Timestamp. Date of the measurement.
- ActivePower, the power generated
- AmbientTemperature, the outside temperature
- BearingShaftTemperature, the motor operating temperature
- Blade1PitchAngle, the pitch angle for blade 1 of the turbine
- Blade2PitchAngle, the pitch angle for blade 2 of the turbine
- Blade2PitchAngle, the pitch angle for blade 3 of the turbine
- ControlBoxTemperature, temperature in control box
- GearBoxBearingTemperatue, temperature in turbine gear system
- GearBoxOilTemperature, temperature of the oil in the gear system
- GeneratorRPM
- GeneratorWinding1Temperature, temperature of stator windings 1

- GeneratorWinding1Temperature, temperature of stator windings 2
- HubTemperature
- MainBoxTemperature
- NacellePosition, the nacelle is where the mechanism that transform mechanical energy to electricity is located.
- ReactivePower
- RotorPRM
- TurbineStatus
- WTG, windmill name
- WindDirection
- WindSpeed

Where the column **ActivePower** is the one that we want to **predict** with our model.

After inspecting the data, we can see that **all the columns are numeric except for the Timestamp and WTG columns which are an object type**. Also, **there are columns with many null values**, but **date values are all different from null**. The column **WTG has one unique value for all the rows** since the data provided regards to the same windmill.



```
Data columns (total 22 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   Unnamed: 0                      118080 non-null  object
 1   ActivePower                     94750 non-null   float64
 2   AmbientTemperatue               93817 non-null   float64
 3   BearingShaftTemperature         62518 non-null   float64
 4   Blade1PitchAngle                41996 non-null   float64
 5   Blade2PitchAngle                41891 non-null   float64
 6   Blade3PitchAngle                41891 non-null   float64
 7   ControlBoxTemperature           62160 non-null   float64
 8   GearboxBearingTemperature       62540 non-null   float64
 9   GearboxOilTemperature           62438 non-null   float64
 10  GeneratorRPM                    62295 non-null   float64
 11  GeneratorWinding1Temperature    62427 non-null   float64
 12  GeneratorWinding2Temperature    62449 non-null   float64
 13  HubTemperature                  62406 non-null   float64
 14  MainBoxTemperature              62507 non-null   float64
 15  NacellePosition                 72278 non-null   float64
 16  ReactivePower                   94748 non-null   float64
 17  RotorRPM                        62127 non-null   float64
 18  TurbineStatus                   62908 non-null   float64
 19  WTG                             118080 non-null  object
 20  WindDirection                   72278 non-null   float64
 21  WindSpeed                       94595 non-null   float64
```

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| ActivePower | 94750.0 | 619.109805 | 611.275373 | -38.524659 | 79.642258 | 402.654893 | 1074.591780 | 1.779032e+03 |
| AmbientTemperatue | 93817.0 | 28.774654 | 4.369145 | 0.000000 | 25.627428 | 28.340541 | 31.664772 | 4.240560e+01 |
| BearingShaftTemperature | 62518.0 | 43.010189 | 5.545312 | 0.000000 | 39.840247 | 42.910877 | 47.007976 | 5.508866e+01 |
| Blade1PitchAngle | 41996.0 | 9.749641 | 20.644828 | -43.156734 | -0.939849 | 0.394399 | 8.099302 | 9.014361e+01 |
| Blade2PitchAngle | 41891.0 | 10.036535 | 20.270465 | -26.443415 | -0.433264 | 0.888977 | 8.480194 | 9.001783e+01 |
| Blade3PitchAngle | 41891.0 | 10.036535 | 20.270465 | -26.443415 | -0.433264 | 0.888977 | 8.480194 | 9.001783e+01 |
| ControlBoxTemperature | 62160.0 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000e+00 |
| GearboxBearingTemperature | 62540.0 | 64.234170 | 10.455556 | 0.000000 | 57.872242 | 64.834662 | 71.079306 | 8.223793e+01 |
| GearboxOilTemperature | 62438.0 | 57.561217 | 6.323895 | 0.000000 | 53.942181 | 57.196089 | 61.305312 | 7.076458e+01 |
| GeneratorRPM | 62295.0 | 1102.026269 | 528.063946 | 0.000000 | 1029.812177 | 1124.860720 | 1515.402005 | 1.809942e+03 |
| GeneratorWinding1Temperature | 62427.0 | 72.460403 | 22.627489 | 0.000000 | 55.492241 | 65.788800 | 85.867449 | 1.267730e+02 |
| GeneratorWinding2Temperature | 62449.0 | 71.826659 | 22.650255 | 0.000000 | 54.763998 | 65.004946 | 85.337740 | 1.260430e+02 |
| HubTemperature | 62406.0 | 36.897978 | 5.178711 | 0.000000 | 33.943949 | 37.003815 | 40.008425 | 4.799618e+01 |
| MainBoxTemperature | 62507.0 | 39.547603 | 5.732783 | 0.000000 | 35.812500 | 39.491310 | 43.359375 | 5.425000e+01 |
| NacellePosition | 72278.0 | 196.290539 | 88.296554 | 0.000000 | 145.000000 | 182.000000 | 271.000000 | 3.570000e+02 |
| ReactivePower | 94748.0 | 88.133966 | 116.596725 | -203.182591 | -0.432137 | 35.883659 | 147.359075 | 4.037136e+02 |
| RotorRPM | 62127.0 | 9.907500 | 4.718421 | 0.000000 | 9.231091 | 10.098702 | 13.600413 | 1.627350e+01 |
| TurbineStatus | 62908.0 | 2280.429214 | 358603.390705 | 0.000000 | 2.000000 | 2.000000 | 2.000000 | 6.574653e+07 |
| WindDirection | 72278.0 | 196.290539 | 88.296554 | 0.000000 | 145.000000 | 182.000000 | 271.000000 | 3.570000e+02 |
| WindSpeed | 94595.0 | 5.878960 | 2.619084 | 0.000000 | 3.823330 | 5.557765 | 7.506710 | 2.297089e+01 |

**Figure 1**. Summary of the data Set. Left, columns with data types and number of non-null value. Right, stats summary.

Also, we can observe the **seasonality** on the data set. **Figure 2** shows how is the time series in four different periods. In each of one it is clearly the periodicity on the events.

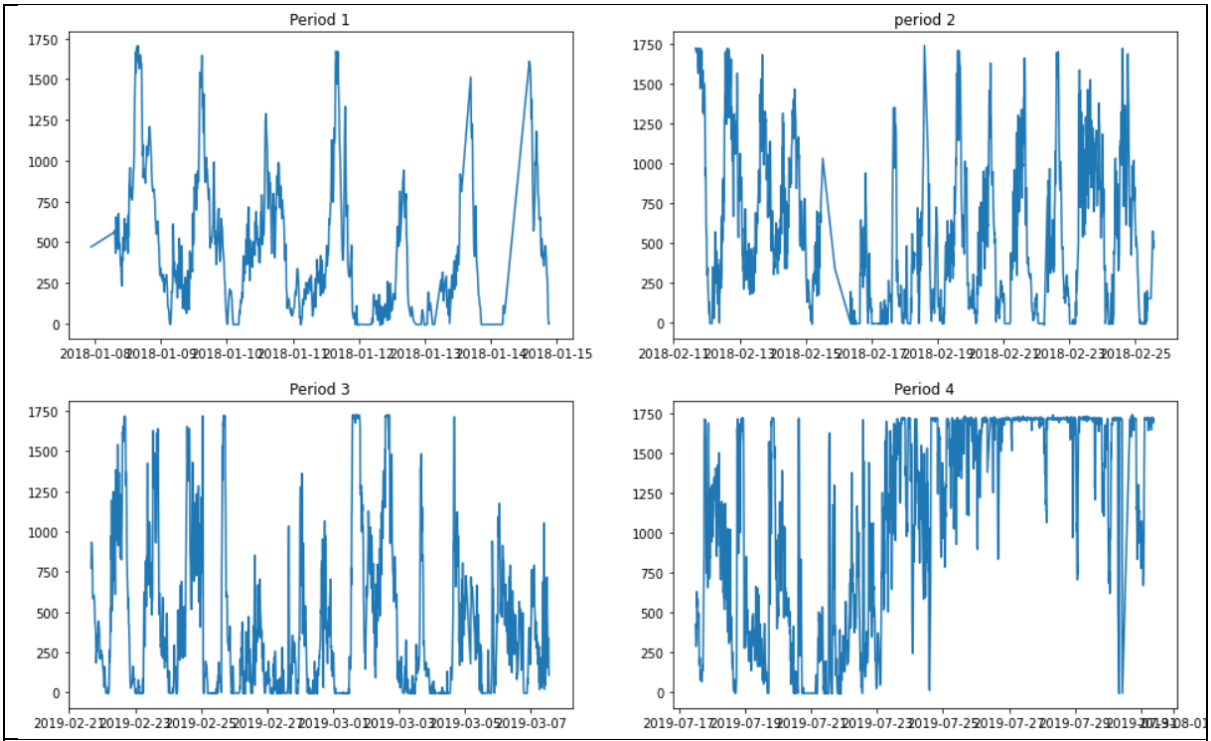



**Figure 2**. Time series for 4 different periods.

Furthermore, we can observe that there is autocorrelation on the time series by plotting the **ACF and the PACF** plots (**figure 3**)

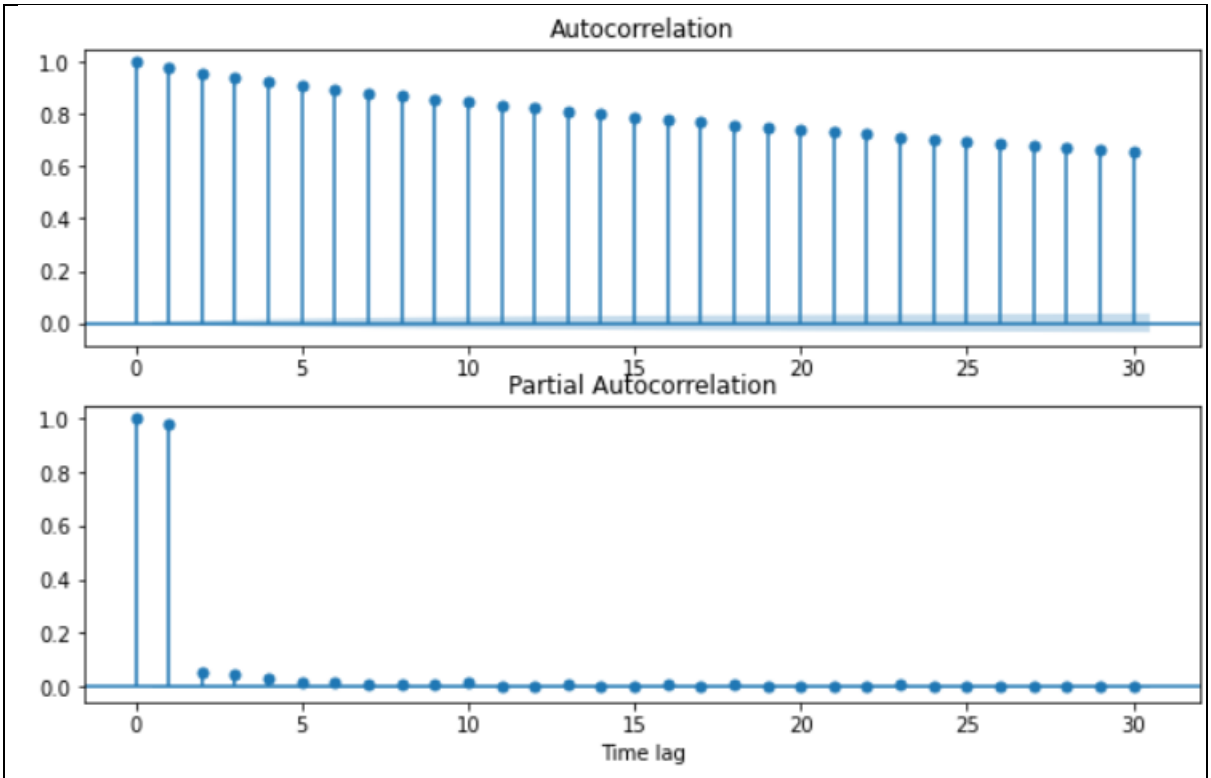



**Figure 3**. ACF and PACF plots.

## Brief Summary of the Data Pre-processing.

First, we can **drop** not useful columns such us **WTG** which has **only a single value** and is not adding any information. We also decide to **drop** the **columns with more than the 50% of missing values** since any other treatment could lead to wrong predictions. So, we will **keep** just the following columns, **Timestamp, ActivePower, NacellePosition, ReactivePower, WindDirection and WindSpeed**.

Since we are working with a Time Series, we need to **set the date as the index** dataframe, to do this we need to convert the **Timestamp dtype from object to dateTime**.

We still need to **deal with missing values**. In this case we are not interested in losing any row to keep the same time spacing between the measurements, so we **apply interpolate method on the dataframe with a linear method**. Thus, the NaN values will be filling with a linear relation.

Next, we have a look on the feature distributions and correlations between the features using a pairplot (**figure 4**):

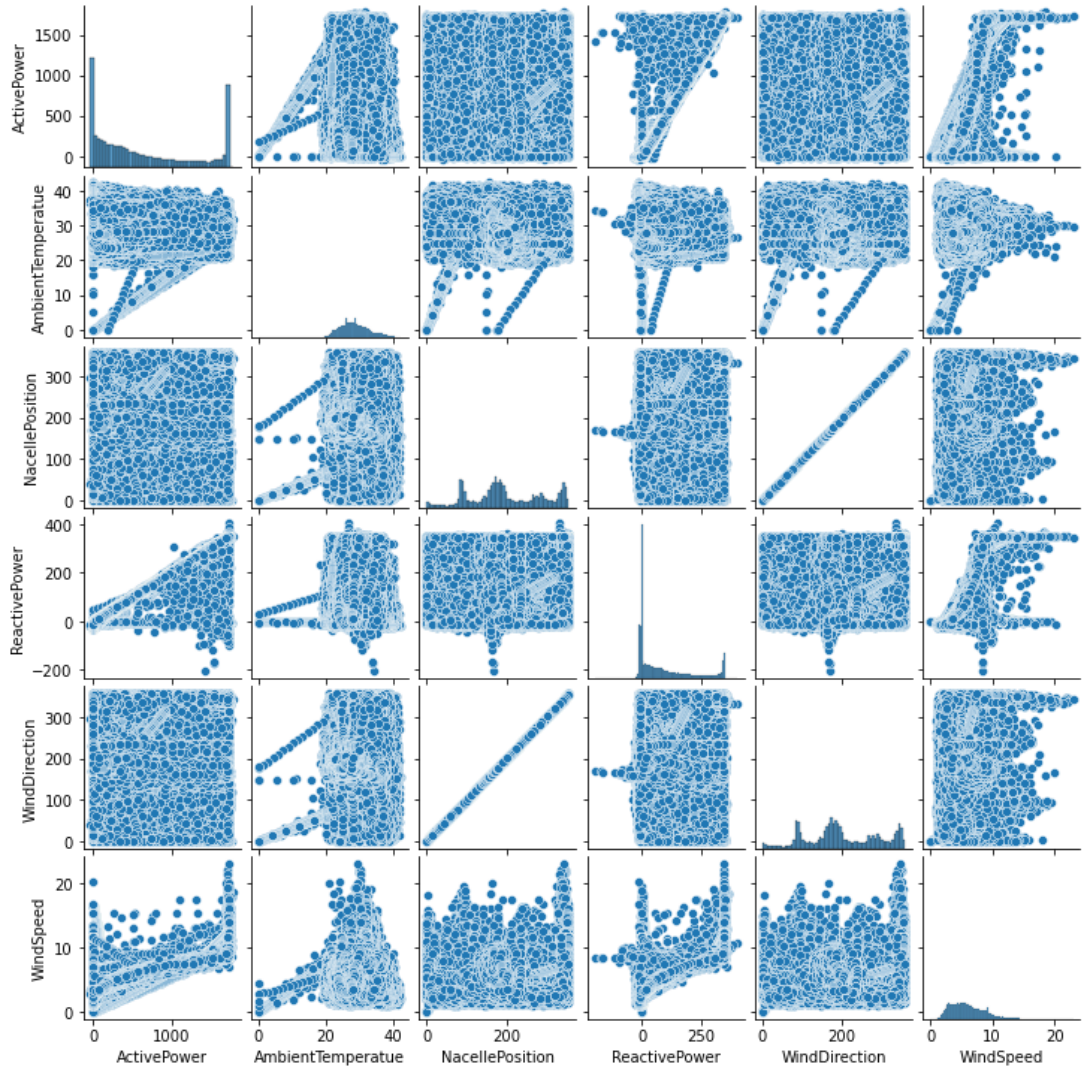



**Figure 4**. Distribution and correlation.

Having a look on this plot, we can see clearly how **NacellePosition** and **WindDirection** are **strongly correlated**. So, we **drop NacellePosition** column.

## The deep learning models

ARIMA models are the most common models used for energy generation forecast, but we decide to go a step ahead I we will perform the **forecast with Deep Learning**. Neural networks offer several benefits over traditional time series forecasting models such as. Neural Networks **automatically learn how to incorporate series characteristics like trend, seasonality, and autocorrelation into predictions**. Also, the neural network is able to capture very complex patterns.

So, we can take advantage of the fact that a model builds with a neural network deal automatically with the series characteristics. **So, to forecast the power generation we don't need to do any transformation due to the trend, seasonality or the autocorrelation**.

**Base model**

As base model we will use a **Recurrent Neural Networks** (RNN) with **a simple RNN layer** composed by **10 nodes and a full connected layer that will have a single output**, the prediction on the power generation. We will use to train the model the measurements of the **last 10 days, and we will train the model through 10 different epochs**.

As we can see on **figure 5**, this model couldn't predict at all the values

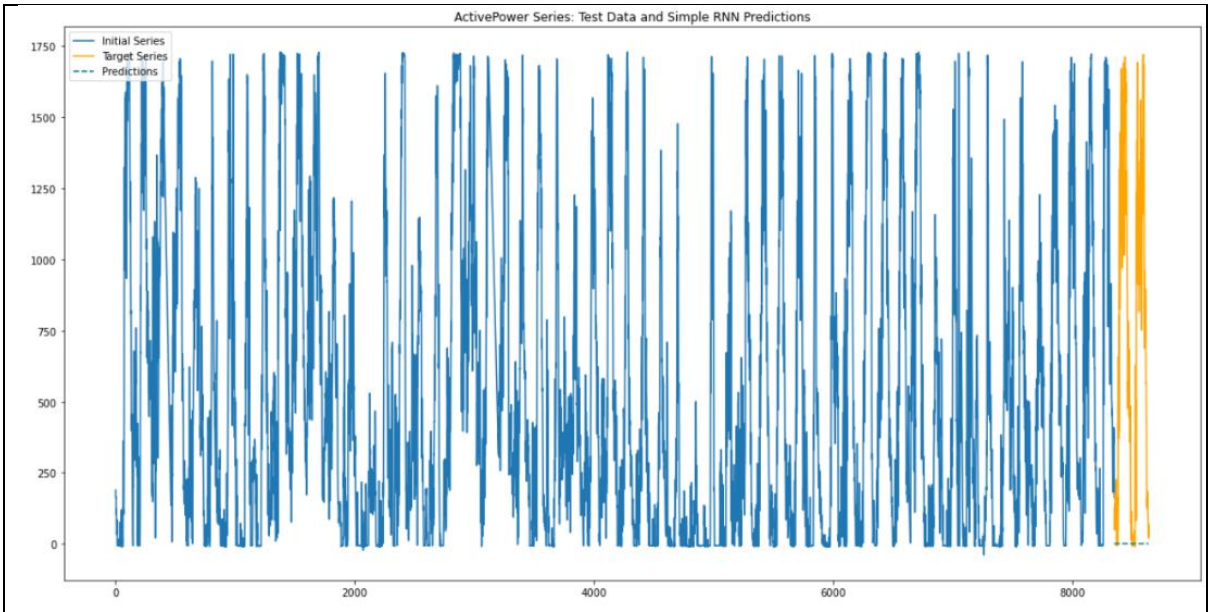



**Figure 5**. Predictions Baseline

**Second model**

We can improve by making the model more expressive, **increasing the number of nodes, for example 30**. We can also pass over the training data many more times, **increasing epochs for example 100**, giving the model more opportunity to learn the patterns in the data.

In this case we can see that the model starts to predict better but still have many problems (**figure 6**)

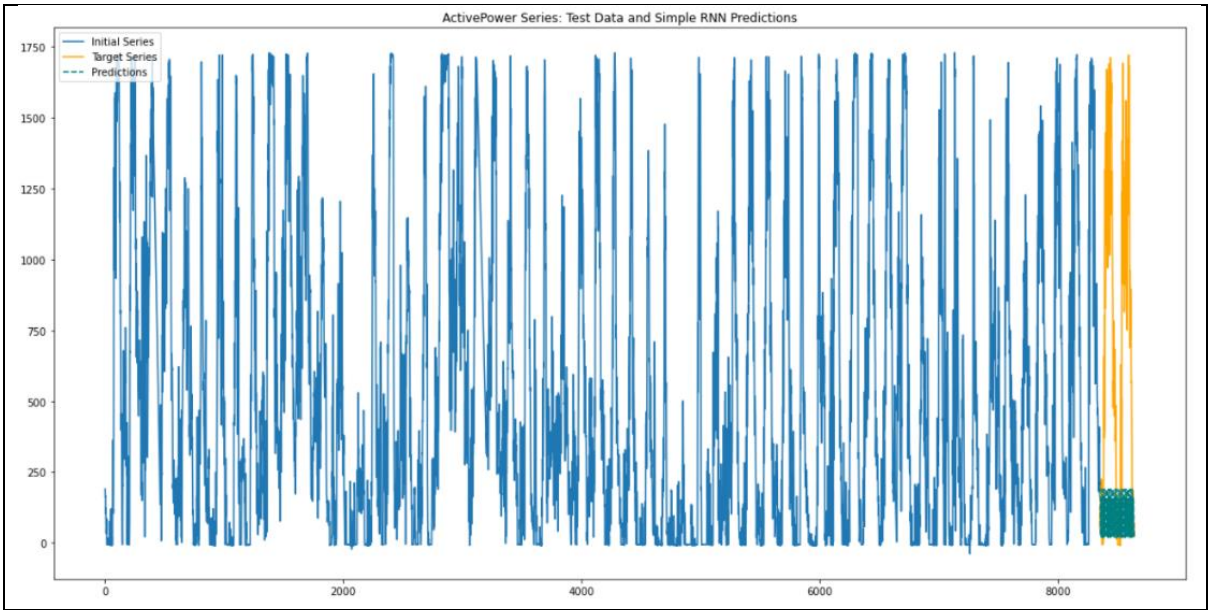



**Figure 6**. Predictions second model

**Third model**

RNNs often struggle to process long input sequences. It is mathematically difficult for RNNs to capture long-term dependencies over many time steps, which is a problem for Time Series, as sequences are often hundreds of steps. **Another type of Neural Networks, Long short-term memory networks** (LSTMs) can mitigate these issues with a better memory system. So, in this case we keep the same structure but we only change the simple RNN layer with a LSTM layer (**figure 7**)

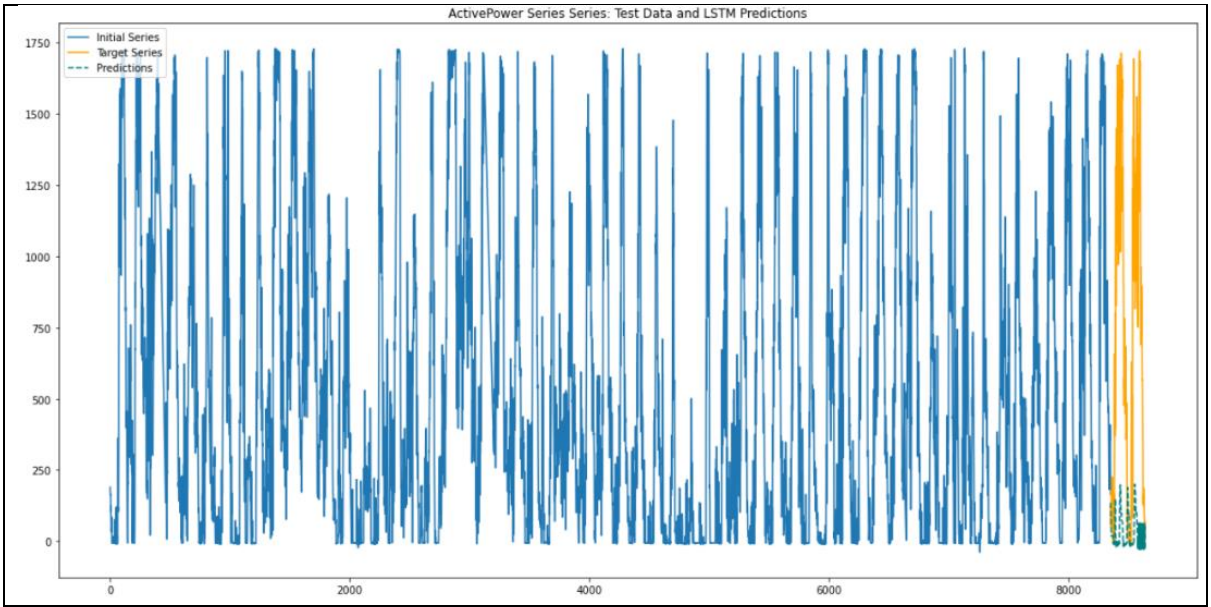



**Figure 7**. Predictions third model

Despite we are still far from a satisfactory forecast, we can observe an improvement on the predictions, which is at least predicting a similar behaviour.

So, **we recommend to use the LSTM model to forecast the energy production rather than the RNN model**. This mechanism regulate information flow and memory storage better so can improve the results in relation with the ones obtained with an RNN model.

## Key findings

Having a look on the data we find that the time series has both a seasonality and autocorrelation. So, **working with deep learning able us to avoid dealing with these characteristics**.

Furthermore, a **LSTM model due to its mechanism that deal better with the memory performs a better forecast than RNN**.

## Next steps

There are many **flaws** on this model, the **dataset finally used is not so large**, only the **last sixty days** were used and a deep learning model performs better with more data. We should **increase** the number of **days used to train the model**, ideally the last 3 years. The more data the better the forecast but on the other side training a deep learning model for so large datasets and features **needs more resources**. Also, we should try to **increase the number of nodes** used in the hidden layer **and the number of epochs** giving a larger opportunity to the model to learn the patterns. If you **don't have access to GPUs maybe we could perform the forecast with less data but using an ARIMA model**.