Wind Energy Prediction Application

PROJECT REPORT

Team Name - Team Seven

Team Members:
1) Kapil Gautam
2) Deeksha Agarwal

Team Contributions:
1) Kapil Gautam - Data Collection, preprocessing, model building, model evaluation, deployment of models, outstanding tasks.
2) Deeksha Agarwal - UI building, Database Creation, integrating OpenWeatherMap API, integrating UI with models.

INDEX

1. INTRODUCTION
1.1 Overview
1.2 Purpose

2.LITERATURE SURVEY
2.1 Existing problem
2.2 Proposed solution

3.THEORITICAL ANALYSIS
3.1 Block diagram
3.2 Hardware / Software designing

4.EXPERIMENTAL INVESTIGATIONS

5.FLOWCHART

6.RESULT

7.ADVANTAGES & DISADVANTAGES

8.APPLICATIONS

9.CONCLUSION

10.FUTURE SCOPE

11.BIBILOGRAPHY
APPENDIX
A. Source code

1) INTRODUCTION

1.1) Overview

This problem requires building a complete application deployed on IBM cloud that includes a time series model that forecasts the wind power output based on the historical data. The application needs to predict wind energy output for a time period of 1 hour to 72 hours into the future. The application also needs to recommend the best time interval for wind power generation from the time interval predicted.

1.2) Purpose

The purpose of building one such application stems from a single goal of increasing dependency on renewable sources of energy and decreasing use of non-renewable sources. This is the primary goal in today's time for many countries in the world and this can be seen in the push for these sources such as solar and wind energy.
In India alone the installed wind power capacity has grown from 7,850 MW in the year 2006-2007 to 37,669 MW in the year 2020.
Keeping this goal in mind several papers have been published based on wind energy generation data, to predict more accurately. One such paper based on the data from **7 wind energy sites in Tamil Nadu, is Analysis of wind power generation and prediction using ANN,** by **M. Carolin Mabel , E. Fernandez [1].** This paper clearly shows the wind power generation potential in India.

2) LITERATURE SURVEY

2.1) Existing Problem

The challenges for predicting the accurate wind power output on a particular farm site are many folds. Based on the research papers on the same topic, we have identified some of the challenges. These challenges are -

- The biggest challenge that exists in predicting the wind energy output lies in the

non stationary nature of weather conditions. It's hard to predict when your input changes constantly. This problem is clearly noted in **Predicting the Energy Output of Wind Farms Based on Weather Data by Katya Vladislavleva, Tobias Friedrich [2].**

- Another problem is finding the dataset that helps to train the model on important parameters. Few datasets are available that have wind output records with suitable parameters.
- Prediction time also poses challenge for the model to forecast into the future when weather conditions are of unstable nature. Latest model published by Google only predicts weather conditions upto 36 hours **[3]**. The input data is crucial for accurately predicting the output.
- Another challenge is deciding which methods to use for this forecasting problem. We tried several different models as mentioned in our solution proposal and will explain our findings in upcoming sections.

2.2) <u>Proposed Solution</u>

We have tried solving the above listed challenges by finding research papers for the current solutions and handpicking several important implementation points to finally build the application. We developed an application that meets the requirements posed by the problem statement. The most important and some unique points about our solution are -

- Dataset - We trained our model on the dataset based on wind energy sites in Greece from the period of 01/01/2016 to 04/05/2018. This dataset contains the recordings at the period of every 10 minutes. Also, as proposed in our solution, this dataset contains 4 fields namely timestamp, wind speed, wind direction and temperature. As stated in **[2],** temperature also plays an important role in wind power generation and this seasonality can be learnt by our model using this field. (**unique proposition)**

- Preprocessing - This dataset contained the records for 10 wind farm sites for every 10 minutes. We combined the data to form single column values for each

feature. Also, we took the mean and converted into hourly data at each row.

- Model Building - In our solution, we proposed to try different models and implement the models that performed the best. Our final application predicts the wind energy output based on two different models. The need for combining different approaches and its benefits have been described in several research papers. Some of the most important points to note regarding ensemble methods are -
    - In **[2],** the authors clearly note that the **final model is a set of individual models of high accuracy.** Prediction of individual models are taken and computed mean or median for final prediction. This helps in "**a more robust prediction of the response (since overfitting is further mitigated by choosing models of different accuracy and complexity into an ensemble), and second, more trustworthy predictions**."

    - In **Current Methods and Advances in Forecasting of Wind Power Generation by Aoife M. Foley , Paul G. Leahy[4],** the author clearly state at page 4 that ensemble forecasting employs a number of different model runs to predict a large sample of possible future weather outcomes. The results are then evaluated by examining the distribution across ensemble 'members' of the forecast variables.
      The authors also note that ensemble methods also provide a measurement of reliability. If the output of different methods differ widely, then the forecast has huge uncertainty. However, if the individual forecasts are close, then the uncertainty is lower.

    - In **[5] Short-term Prediction by Lars Landberg and Gregor Giebel,** the author note at page 7 that using ensemble forecasting (e.g. combining forecasts from different NWP models). This will increase the overall accuracy and most likely also give better estimates of the error.

Basing our model on these research findings, we trained two machine learning models

on our dataset. The two models trained are artificial neural networks and the other one begin XGB-Regressor (**unique proposition).**

a) <u>Artificial Neural Network</u>

The effectiveness of artificial neural network has been proved in research. The ANN have proved to be performing better than the normal statistical methods like ARIMA. The ANN predicts the output based on wind speed, wind direction as well as the temperature values. We have found ANN to be working really well for our dataset.

There is not much difference between the train and test RMSE values which was the metric for evaluation for this model. The individual RMSEs are little high cause we only trained 50 epochs, because the training was taking a lot of time and also ending abruptly. However we tried training the same model on different platform with 250 epochs and the individual RMSEs came to down to under 1000. hence, we believe this model can be trained on any dataset with above stated parameters and will perform a reliable output. This model is trained and stored in IBM Cloud as required by hack challenge rules.

b) <u>XGB Regressor</u>

To create our ensemble model, we then tried searching for another high performing method that could make use of all the 3 parameters and make reliable predictions for the time period specified. This search led us to the amazing tool called Auto AI in IBM Cloud. Out of 8 different methods tested on the dataset, this regressing model performed the best and predicted most accurate values.

Rank	Name	Algorithm	RMSE (Optimized)	Enhancements	Build time
★ 1	Pipeline 4	XGB Regressor	333.533	HPO-1 FE HPO-2	00:30:49
2	Pipeline 3	XGB Regressor	337.333	HPO-1 FE	00:02:12
3	Pipeline 7	Gradient Boosting Regressor	341.554	HPO-1 FE	00:03:24
4	Pipeline 8	Gradient Boosting Regressor	341.554	HPO-1 FE HPO-2	00:01:25
5	Pipeline 1	XGB Regressor	341.882	*None*	00:00
6	Pipeline 2	XGB Regressor	341.882	HPO-1	00:07:46
7	Pipeline 5	Gradient Boosting Regressor	343.694	*None*	00:00:03
8	Pipeline 6	Gradient Boosting Regressor	343.694	HPO-1	00:00:29

Hence these two models that performed the best than the others are deployed. Both the models are deployed on IBM cloud. The predictions from these models are taken into account and then the final prediction is made.

- **<u>Node RED and User Interface:</u>**

We have built our UI in the Node RED and is also deployed on IBM cloud. The **UI is designed to be convenient for user by taking minimum inputs from the user and producing maximum results** that are useful for the user.

Find the UI here:

https://node-red-seven.eu-gb.mybluemix.net/ui/#!/0?socketid=0rhfSrtrPdiOimYiAAAT

Node red contains two pages namely - Home Page and Compare Page.

The **Home Page** consists of a user form that asks the user 4 fields -

- Country - Where the wind farm is located.
- City - The city of the wind site.
- Prediction hours - No. of hours for which the user wants to predict (max. 72).

- Recommended hours - As stated in problem, app needs to recommend the Power Grid to suggest the best time to utilize the energy from wind farm. User will Enter no. of hours for which the recommendation is required from the predicted values.

Once these values are inputted, the application generates the prediction of power output values based on the collection of weather data from **OpenWeatherMap APIs** for the prediction time period and feeds it into the models for prediction. These values are then displayed to the user **in form of Min - Max range.** This range gives an idea of the power output to the user. Also, the **power graph is displayed** based on the outputs of both the models, along with the graphs of the Weather Parameters namely, Wind Speed and Temperature. Also, **User can download the csv file** containing Timestamp, all the weather parameters used, along with the predicted Power Range in a single click. Also, **the best time period for power generation is displayed calculated on the predicted values using the Kadane's Algorithm** which is optimal for finding the maximum subarray from a given array. Also, **exception handling** is done in the UI fields.

The **Compare Page** takes the input location for two sites. It then uses the weather data and displays the predicted Power Output Values in a Table for the next 72 hours. Also, Timeseries Graph is displayed comparing Power Output from both the Sites.

Team Seven Home Page
Compare Two Locations Pov
Country1
set global.country1
City1
set global.city1
Country2
set global.country2
City2
set global.city2
Taking Input from Users in UI
Predict And Compare Power Output
msg.payload
Processing for location 1
pass city, country
Get Weather Params
split
take relevant features from weather data
join
array length modifier
split
PreToken
Pre Prediction
http request
http request m
msg.pa
Using Machine Learning Model To Predict Power Output
Processing for location 2
pass city, country
Get Weather Params
split
take relevant features from weather data
join
array length modifier
split
PreToken
Pre Prediction
http request
http request mode

Node-RED
Deploy
Team Seven Home Page
Compare Two Locations Pov
Pre Prediction
set power output
http request model2
join
msg.payload
setting msg.parts
Using Machine Learning Model To Predict Power Output
setting msg.parts
Pre Prediction
set power output
http request model2
join
join
msg.payload
power fetch for table
Power Output Table
Displaying Comparative Results in UI
Power Fetch For 2 Cities
Wind Power Energy (in kW) Output TimeSeries Graph

These are some of the UI screenshots:

Home Page:

Compare Page:

3) THEORETICAL ANALYSIS

3.1) Block Diagram:

3.2) Hardware / Software Designing

Our application is built using IBM node-red service and is deployed on IBM cloud. For forecasting models, we used IBM Watson Studio for developing artificial neural network from scratch and used IBM Auto AI service for developing the XGB regressor model. Both the models are deployed on IBM cloud and are trained on the dataset containing three parameters of wind speed, wind direction and temperature from 2016 to half of 2018 from wind farms in Greece. This dataset after taking the mean of values on hourly basis contains 20,000+ rows.

4) EXPERIMENTAL INVESTIGATIONS

In our research for understanding the problem and challenges associated with the problem, we have identified few conclusions. Many of these conclusions are in line with the research papers referred in this report. Some findings were new to us. These

findings are -

- ANN are really efficient for this problem. This conclusion is also supported by a study conducted on wind farms in Tamil Nadu **[1].**

- Including one more parameter will indeed help in making better predictions. We included an extra field of temperature. In **[2],** temperature is showed to have importance in predicting the power output.

- Using ensemble methods really works better instead of using a single model for prediction. It has many benefits as mentioned in solution description.

- Initially, we proposed to use the model **Prophet** by Facebook, but during our evaluation the model was not found to be performing on the dataset. Also, **by default prophet fits a straight line to data** and hence is not performing well. It also predicts based on the timestamps and was not useful as we wanted to predict based on 3 parameters instead of time. Results of prophet on our dataset are found to be -

calability, consistency & naming convention, exception handling in the code, emphasis on the UI/UX design/framework implementation.

	yhat	yhat_lower	yhat_upper	y
ds				
2016-01-01	1951.333120	-961.718583	4695.147368	752.951389
2016-01-02	1787.017110	-1024.061617	4813.557258	791.271528
2016-01-03	2004.229737	-877.839700	4853.189468	5636.693750
2016-01-04	2932.499264	134.239418	6102.059863	3256.499306
2016-01-05	1942.905880	-1170.798896	4923.298814	2310.003472

As it can be seen from the table that predicted values are far off from the actual power output values. The graphs plotted on forecasted values are -

As we can see the actual values lie far away from the predicted ones.

- Using ARIMA - In our proposed solution we also proposed to test ARIMA for forecasting problem as done in some of the research papers. One disadvantage of ARIMA is that it only predicts based on one column of timestamp rather than the actual parameters like wind speed. Also, the data needs to be stationary for ARIMA to be used, which is not the case for our dataset.

5) FLOWCHART

Input Location, Prediction Hours
Prediction Hours<48
Yes
No
hourly weather data from One Call API
Weather Data from Five Day Forecast API (in interval of 3 hours)
Formatting size of array on basis of no. of hours
Predicting Power using Machine Learning Model
Using kadane's algorithm to calculate best time to utilize Energy
Saving in Database
Results in User Interface

6) RESULT

An end to end application is developed and deployed on IBM cloud which predicts the wind power output based on the data it is trained on, which predicts power output based on three important parameters namely wind speed, wind direction and temperature. The application displays the power output in form of min-max range (just like the weather data is predicted) and displays the graph of power output values as well the wind speed and temperature. The user can also download the CSV files of power predictions. The application also recommends the most productive time period in terms of power generation for inputted time period for recommendation by user. User can also compare the power output for two sites by just inputting simple information. The weather data is collected from Open Weather API and is used by two best performing model, artificial neural network as well as XGB regressor to predict the final values.

7) ADVANTAGES AND DISADVANTAGES

- Advantages of this model includes that it **predicts the power output based on three parameters** that are found to be important in research.
- Also, this model **works on the concept of ensemble model** and uses two best performing models from a pool of models to predict the final output. This **increases the reliability and accuracy of the predicted values.**
- This application provides a database to store the current weather data as well as power outputs that can later be used to train the model for better predictions.
- **User can use the downloaded CSV file** which has predicted Power Output Data to append actual Power Output Values and it can be **used for feedback for the model.**
- Insights from Compare Tab can **help user to choose new site for installation of Wind Mills** and in case of existing Wind Farm , **can help in making informed decisions with regards to distribution of the electricity load**.
- The **model can be trained on the historical data of any site** and can be used to predict the values for that site. The model is deployed in the cloud and can be easily used and extended.
- Disadvantages of the model include the fact that the ANN is not trained on large enough iterations over the data due to the limitation of our IBM Cloud Lite

account. The RMSE will further decrease on training it on more iterations.

- Also, this model trained on only the dataset from wind farms from Greece, so to deploy it to any site in India or other country, we need to train it on the historical dataset for accurate predictions.

8) APPLICATIONS

- The above model can be easily deployed and extended to work according to the weather conditions at any of the 9 places in India where the wind energy is being produced. Different sites can keep training the model with their own data with the unique IBM Watson Studio feature called **Continuous Learning** and improve personalized forecasts.
- Accurate energy forecasts can help in more informed decisions with regards to electricity transmission sizing, sizing of nearby DERs, distribution of the electricity load, sizing of energy storage systems, and other issues related to renewable energy integration in a smart grid or microgrid.
- Our country is pursuing ambitious wind energy goals. This model's prediction can be used by energy suppliers to make better decisions and collaborate with more stable energy sources producers for efficient production reducing overproduction and costs related to that overproduction.

9) CONCLUSION

The wind energy forecast is a really important milestone for promoting the use of renewable energy sources. Most of the countries are pushing the use of energy produced from these sources. In India itself, the Prime Minister has launched an ambitious project for promoting the use of solar generated power. If we want the wind energy to be reliable at that scale, this forecasting challenge need to be overcome. This project implemented best techniques that proved to work in various researches across the globe using a dataset from Greece. This performance can be carried out and further

improved for other wind farm sites in India and other countries.

10) FUTURE SCOPE

In future, this model can be integrated with other machine learning or deep learning methods which prove suitable for the nature of the problem. Also, this model can be trained for various sites and used to predict values for that particular site.

11) BIBLIOGRAPHY

1) Analysis of wind power generation and prediction using ANN, by M. Carolin Mabel , E. Fernandez. Find the paper here.

2) Predicting the Energy Output of Wind Farms Based on Weather Data by Katya Vladislavleva, Tobias Friedrich. Find the paper here.

3)https://analyticsindiamag.com/from-weather-forecasting-to-nowcasting-how-google-is-using-ml-to-predict-precipitation/

4) Current Methods and Advances in Forecasting of Wind Power Generation by Aoife M. Foley , Paul G. Leahy. Find the paper here.

5) Short-term Prediction by Lars Landberg and Gregor Giebel. Find the paper here.

6) ANN-based evaluation of wind power generation: A case study in Kutahya, Turkey Find the paper here.

Source Code

https://github.com/SmartPracticeschool/SBSPS-Challenge-918-Wind-Energy-Prediction-App

Code For ANN

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import types
```

```
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_e713e4258c5b42f3a00b89fcaa820e9b = ibm_boto3.client(service_name='s3',
ibm_api_key_id='7MQvdbzKXZRO5GYyWITSdxd0vvKJFjQg9VvK2Dxmo4xS',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
endpoint_url='https://s3-api.us-geo.objectstorage.service.networklayer.com')

body = client_e713e4258c5b42f3a00b89fcaa820e9b.get_object(Bucket='windoutput-donotdelete-pr-dxhrchiibgtonl',Key='windoutput.xlsx')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

```

```
data = pd.read_excel(body)
data.head()
import keras
from keras.models import Sequential
from keras.layers import Dense
model=Sequential()
model.add(Dense(input_dim=3,init="random_uniform",activation="relu",output_dim=3))
model.add(Dense(output_dim=3,init="random_uniform",activation="relu"))
model.add(Dense(output_dim=1,init="random_uniform"))
model.compile(optimizer="adam", loss="mse", metrics=['mse'])
from sklearn.model_selection import train_test_split
X=data.drop(['TotalPower[kW]','TimeStamp'],axis=1)
y=data['TotalPower[kW]']
X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.3,random_state=0)
print(X_train.head())
model.fit(X_train, y_train, epochs=50, batch_size=64)
y_predict = model.predict(X_test)
import math
from sklearn.metrics import mean_squared_error
rmse = math.sqrt(mean_squared_error(y_test, y_predict))
print('Test RMSE: %.3f' % rmse)
y_train_predict = model.predict(X_train)
rmse = math.sqrt(mean_squared_error(y_train, y_train_predict))
print('Train RMSE: %.3f' % rmse)
```

Code for XGB

```
from watson_machine_learning_client.helpers import DataConnection, S3Connection, S3Location

training_data_reference = [DataConnection(
    connection=S3Connection(
api_key='Qr3p3KN9EEcG5pUBAqvNvKnHJIU4C5SJYfBZzr8ZZ6_a',
auth_endpoint='https://iam.bluemix.net/oidc/token/',
endpoint_url='https://s3-api.us-geo.objectstorage.softlayer.net'
    ),
        location=S3Location(
        bucket='windoutput-donotdelete-pr-dxhrchiibgtonl',
        path='Windoutput.csv'
    ))
]
training_result_reference = DataConnection(
    connection=S3Connection(
api_key='Qr3p3KN9EEcG5pUBAqvNvKnHJIU4C5SJYfBZzr8ZZ6_a',
auth_endpoint='https://iam.bluemix.net/oidc/token/',
endpoint_url='https://s3-api.us-geo.objectstorage.softlayer.net'
    ),
    location=S3Location(
        bucket='windoutput-donotdelete-pr-dxhrchiibgtonl',
path='auto_ml/d58fad55-608a-42f8-84dd-f66db1f6c5f1/wml_data/6bcf9a7d-8fc3-4e5e-8173-884abb54c099/data/automl',
model_location='auto_ml/d58fad55-608a-42f8-84dd-f66db1f6c5f1/wml_data/6bcf9a7d-8fc3-4e5e-8173-884abb54c099/data/automl
```

```
/hpo_c_output/Pipeline1/model.pickle',
training_status='auto_ml/d58fad55-608a-42f8-84dd-f66db1f6c5f1/wml_data/6bcf9a7d-8fc3-4e5e-8173-884abb54c099/training-status.json'
    ))
experiment_metadata = dict(
    prediction_type='regression',
    prediction_column='TotalPower[kW]',
    test_size=0.15,
    scoring='neg_root_mean_squared_error',
    csv_separator=',',
    excel_sheet=0,
    max_number_of_estimators=2,
    training_data_reference = training_data_reference,
    training_result_reference = training_result_reference)

pipeline_name='Pipeline_4'
from watson_machine_learning_client.experiment import AutoAI

optimizer = AutoAI().runs.get_optimizer(metadata=experiment_metadata)
pipeline_model = optimizer.get_pipeline(pipeline_name=pipeline_name)
pipeline_model.pretty_print(combinators=False, ipython_display=True)
pipeline_model.visualize()
training_df, holdout_df = optimizer.get_data_connections()[0].read(with_holdout_split=True)

train_X = training_df.drop([experiment_metadata['prediction_column']], axis=1).values
train_y = training_df[experiment_metadata['prediction_column']].value
```

```
s

test_X = holdout_df.drop([experiment_metadata['prediction_column']], axis=1).values
y_true = holdout_df[experiment_metadata['prediction_column']].values
from sklearn.metrics import r2_score

predictions = pipeline_model.predict(test_X)
score = r2_score(y_true=y_true, y_pred=predictions)
print('r2_score: ', score)

```

Code for Preprocessing Data

```
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_e713e4258c5b42f3a00b89fcaa820e9b = ibm_boto3.client(service_name='s3',
ibm_api_key_id='7MQvdbzKXZRO5GYyWITSdxd0vvKJFjQg9VvK2Dxmo4xS',
ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
```

```
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3-api.us-geo.objectstorage.service.networklayer.com')

body = client_e713e4258c5b42f3a00b89fcaa820e9b.get_object(Bucket='windoutput-donotdelete-pr-dxhrchiibgtonl',Key='Windfarm data.xlsx')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType(__iter__, body)

df_data_0 = pd.read_excel(body)
df_data_0.head()
def preprocess(df):
    df = df.copy()
    all_columns = df.columns
    total_active_power_avg = ['WTG 1-Active power avg [kW]', 'WTG 10-Active power avg [kW]',
        'WTG 11-Active power avg [kW]', 'WTG 2-Active power avg [kW]',
        'WTG 3-Active power avg [kW]', 'WTG 4-Active power avg [kW]',
        'WTG 5-Active power avg [kW]', 'WTG 6-Active power avg [kW]',
        'WTG 7-Active power avg [kW]', 'WTG 8-Active power avg [kW]',
        'WTG 9-Active power avg [kW]']
    total_wind_speed_avg = ['WTG 1-Wind speed avg [m/s]', 'WTG 10-Wind speed avg [m/s]',
        'WTG 11-Wind speed avg [m/s]', 'WTG 2-Wind speed avg [m/s]',
        'WTG 3-Wind speed avg [m/s]', 'WTG 4-Wind speed avg [m/s]',
        'WTG 5-Wind speed avg [m/s]', 'WTG 6-Wind speed avg
```

```
[m/s]',
        'WTG 7-Wind speed avg [m/s]', 'WTG 8-Wind speed avg [m/s]',
        'WTG 9-Wind speed avg [m/s]']
    total_wind_direction_avg = ['WTG 1-Wind direction avg [°]', 'WTG 10-Wind direction avg [°]',
        'WTG 11-Wind direction avg [°]', 'WTG 2-Wind direction avg [°]',
        'WTG 3-Wind direction avg [°]', 'WTG 4-Wind direction avg [°]',
        'WTG 5-Wind direction avg [°]', 'WTG 6-Wind direction avg [°]',
        'WTG 7-Wind direction avg [°]', 'WTG 8-Wind direction avg [°]',
        'WTG 9-Wind direction avg [°]']
    total_ambient_temperature_avg = ['WTG 1-Ambient temperature avg [°C]', 'WTG 10-Ambient temperature avg [°C]',
        'WTG 11-Ambient temperature avg [°C]', 'WTG 2-Ambient temperature avg [°C]',
        'WTG 3-Ambient temperature avg [°C]', 'WTG 4-Ambient temperature avg [°C]',
        'WTG 5-Ambient temperature avg [°C]', 'WTG 6-Ambient temperature avg [°C]',
        'WTG 7-Ambient temperature avg [°C]', 'WTG 8-Ambient temperature avg [°C]',
        'WTG 9-Ambient temperature avg [°C]']
    df = df.interpolate()
    df[total_active_power_avg] = df[total_active_power_avg].clip(0, 850)
    df[total_wind_speed_avg] = df[total_wind_speed_avg].clip(0, 30)
    df[total_wind_direction_avg] = df[total_wind_direction_avg].clip(0, 360)
    df[total_ambient_temperature_avg] = df[total_ambient_temperature_avg].clip(-10, 40)

```

```
    df["TotalPower[kW]"] = df[total_active_power_avg].sum(axis = 1)
    df["TotalWindSpeed[m/s]"] = df[total_wind_speed_avg].mean(axis = 1)
    df["TotalTemperature[°C]"] = df[total_ambient_temperature_avg].mean(axis = 1)
    df["TotalWindDirection[°]"] = df[total_wind_direction_avg].mean(axis = 1)
    df.drop(all_columns, axis = 1, inplace = True)
    df = df.resample('H').mean()
    return df

```