Weather-Based Prediction of Wind Turbine Energy Output: A Next-Generation Approach to Renewable Energy Management

Team ID : LTVIP2026TMIDS78411

Team Leader : Kalam Sudheer Akash

Team member : Sankar Reddy Ambati

Team member : Attili Avinash

Team member : Bommisetti Bala Tripura Naga Maheswari

Introduction:

The project aims to predict the energy output of a wind turbine based on weather conditions. This is valuable for energy companies and grid operators to better manage and optimize energy production. By analyzing historical data of weather conditions and energy output, machine learning models can be trained to predict the energy output of a wind turbine given current weather conditions.
Scenario 1: Energy Production Forecasting Energy companies want to forecast the energy production of their wind turbines for a given period. They can use machine learning models to predict the energy output based on weather forecasts, helping them make informed decisions about energy distribution and pricing.
Scenario 2: Maintenance Planning Wind farm operators want to plan maintenance schedules for their turbines to minimize downtime and maximize energy production. By predicting energy output based on weather conditions, they can schedule maintenance during periods of low wind activity.
Scenario 3: Grid Integration Grid operators want to integrate wind energy into the grid efficiently. By predicting the energy output of wind turbines, they can better balance the grid by adjusting the output of other energy sources accordingly.

Project Objectives:

- You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
- You will be able to know how to pre-process/clean the data using different data preprocessing techniques.
- You will able to analyze or get insights into data through visualization.
- Applying different algorithms according to the dataset and based on visualization.
- You will be able to know how to build a web application using the Flask framework.

Project Flow:

- User interacts with the UI (User Interface) to enter Data
- The entered data is analyzed by the model which is integrated
- Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities and tasks listed below

- Data Collection.
    - Collect the dataset or Create the dataset
- Data Preprocessing.
    - Import the Libraries.
    - Importing the dataset.
    - Checking for Null Values.
    - Data Visualization.
    - Taking care of Missing Data.
    - Label encoding.
    - One Hot Encoding.
    - Feature Scaling.
    - Splitting Data into Train and Test.
- Model Building
    - Training and testing the model
    - Evaluation of Model
- Application Building
    - Create an HTML file
    - Build a Python Code

Prerequisites:

In order to develop this project we need to install the following software/packages:

Step 1:

Anaconda Navigator :

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform, package management system. Anaconda comes with great tools like JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code.

For this project, we will be using **Jupyter** notebook and **Spyder**

Step 2:

To build Machine learning models you must require the following packages

Sklearn: Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms.

NumPy: NumPy is a Python package that stands for 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object

Pandas: pandas is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.

Matplotlib: It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits

Flask: Web framework used for building Web applications.

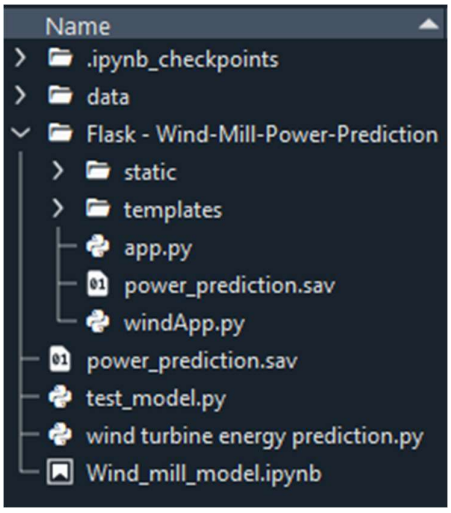
If you are using anaconda navigator, follow the below steps to download the required packages:

1. Open anaconda prompt.
2. Type “pip install numpy” and click enter.
3. Type “pip install pandas” and click enter.
4. Type “pip install matplotlib” and click enter.
5. Type “pip install scikit-learn” and click enter.
6. Type “pip install Flask” and click enter.

If you are using **Pycharm IDE**, you can install the packages through the command prompt and follow the same syntax as above.

Project Structure:

Create a Project folder which contains files as shown below



- Flask folder contains the necessary files for a web application:
- static folder contains the images for web application
- templates folder contains the HTML pages
- .sav file is the model file
- windApp.py is for server-side scripting
- .csv file is the dataset
- .py files are the training and testing files.

Data Collection:

ML depends heavily on data, without data, a machine can't learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

Download dataset /create dataset

The dataset for wind energy prediction is to be collected. The dataset which is considered here will have the environmental conditions. You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository etc.

Data pre-processing:

In this milestone, we will be preprocessing the dataset that is collected. Preprocessing includes:

1. Processing the dataset.

2. Handling the null values.
3. Handling the categorical values if any.
4. Normalize the data if required.
5. Identify the dependent and independent variables.
6. Split the dataset into train and test sets.

Import required libraries

The libraries can be imported using the import keyword.

```
# Importing Necessary Libraries
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
```

Analyze the datasets

Step 1: The datasets are imported as data frames using the pandas library. Rename the columns with suitable column names for better understanding.

*Dataset contains the wind speed and wind direction along with the power generated.

```
path = "data/T1.csv"

df = pd.read_csv(path)

df.rename(columns={'Date/Time':'Time',
                   'LV ActivePower (kW)':'ActivePower(kW)',
                   "Wind Speed (m/s)":"WindSpeed(m/s)",
                   "Wind Direction (°)":"Wind_Direction"},
                    inplace=True)
```

Step 2: Check the correlation between the columns for dimensionality reduction (knowing which columns are necessary and which are not)

```
#Data Description and Visualizing

#Plotting the pair plot, each variable in the data set is plotted with all other variables
sns.pairplot(df)

#Plotting Correlation between the variables
plt.figure(figsize=(10, 8))
corr = df.corr()
ax = sns.heatmap(corr, vmin = -1, vmax = 1, annot = True)
bottom, top = ax.get_ylim()
ax.set_ylim(bottom + 0.5, top - 0.5)
plt.show()

#Print the numerical form for the correlation
print(corr)
```

```
# The heat map clearly tells us that there's no realtion between wind direction and
# the Power generated but Wind speed, Theoritical power and Actual power generated
# have a very positive correlation

#df.drop(['Wind_Direction'],axis=1,inplace = True)
df["Time"] = pd.to_datetime(df["Time"], format = "%d %m %Y %H:%M", errors = "coerce")
```

The heat map clearly tells us that there's no relation between wind direction and the Power generated but Wind speed, Theoretical power and Actual power generated to have a very positive correlation.

Splitting data into independent and dependent variables

In this activity, the dependent and independent variables are to be identified. The independent columns are considered as x and the dependent column as y.

After identifying the dependent and independent variables, the dataset now has to be split into two sets, one set is used for training the model and the second set is used for testing how good the model is built. The split ratio we consider is 80% for training and 20% for testing.

```
#Splitting the Data into train and test :
y = df['ActivePower(kW)'] #'Theoretical_Power_Curve (KWh)'
X = df[['Theoretical_Power_Curve (KWh)','WindSpeed(m/s)']]#'ActivePower(kW)'

from sklearn.model_selection import train_test_split
train_X, val_X, train_y, val_y = train_test_split(X, y,random_state = 0)
```

Model Building:

There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms can be chosen according to the objective. As the dataset which we are using is a Regression dataset so you can use the following algorithms

- Linear Regression
- Random Forest Regression / Classification
- Decision Tree Regression / Classification

Choose the appropriate model

We will be considering the Random Forest Regressor model and fit the data.

```
#Importing libraries for Model training and fitting
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error,r2_score

forest_model = RandomForestRegressor(n_estimators = 750, max_depth = 4,
                                     max_leaf_nodes =500, random_state=1)
forest_model.fit(train_X, train_y)
```

Check the metrics of the model

Here we will be evaluating the model built. We will be using the test set for evaluation. The test set is given to the model for prediction and prediction values are stored in another variable called y_pred. The r2 score of the model is calculated and its performance is estimated.

```
#Predicting for Test Data
power_preds = forest_model.predict(val_X)

#Evaluating the score of our model
print(mean_absolute_error(val_y, power_preds))
print(r2_score(val_y,power_preds))
```

Save the model

The finalised model is now to be saved. We will be saving the model as a pickle or **pkl** file or **sav** file.

```
#Saving the model for future reference
joblib.dump(forest_model, "power_prediction.sav")
```

API Integration

Step 1: Signup for OpenWeather API for current weather forecasting.To signup click here

Step 2: After verification and subscription within 24 hours the API key will be activated.

Step 3: The API Key can be used to get the weather forecast of any of the cities known. The city is passed with parameter q and apikey is to be given with the parameter appid. An example for London city is shown below.

Application Building:

After the model is built, we will be integrating it to a web application so that normal users can also use it to predict the energy in a no-code manner. In the application, the user provides the required values and get the predictions.

Build the python flask app

In the flask application, the API requests, as well as energy prediction requests, are taken and the results are processed.

Step 1: Import required libraries

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import joblib
import requests
```

Step 2: Load the model and initialise flask app

```
app = Flask(__name__)
model = joblib.load('power_prediction.sav')
```

Step 3: Configure app.py for api requests

Flask file takes the city as input and hits the API to get the weather conditions and send it back to the UI.

```
@app.route('/')
def home():
    return render_template('intro.html')

@app.route('/predict')
def predict():
    return render_template('predict.html')

@app.route('/windapi',methods=['POST'])
def windapi():
    city=request.form.get('city')
    apikey="43ce69715e2133b2300e0f8f7289befd"
    url="http://api.openweathermap.org/data/2.5/weather?q="+city+"&appid="+apikey
    resp = requests.get(url)
    resp=resp.json()
    temp = str(resp["main"]["temp"])+" °C"
    humid = str(resp["main"]["humidity"])+" %"
    pressure = str(resp["main"]["pressure"])+" mmHG"
    speed = str(resp["wind"]["speed"])+" m/s"
    return render_template('predict.html', temp=temp, humid=humid, pressure=pressure,speed=speed)
```

Step 4: Configure the file with predictions

It takes the inputs from the UI and passes it to the model and sends the predicted output to the UI.

```
@app.route('/y_predict',methods=['POST'])
def y_predict():
    '''
    For rendering results on HTML GUI
    '''
    x_test = [[float(x) for x in request.form.values()]]

    prediction = model.predict(x_test)
    print(prediction)
    output=prediction[0]
    return render_template('predict.html', prediction_text='The energy predicted is {:.2f} KWh'.format(output))
```

Step 5: Run the app

Enter commands as shown below

```
if __name__ == "__main__":
    app.run(debug=False)
```

Build an HTML Page

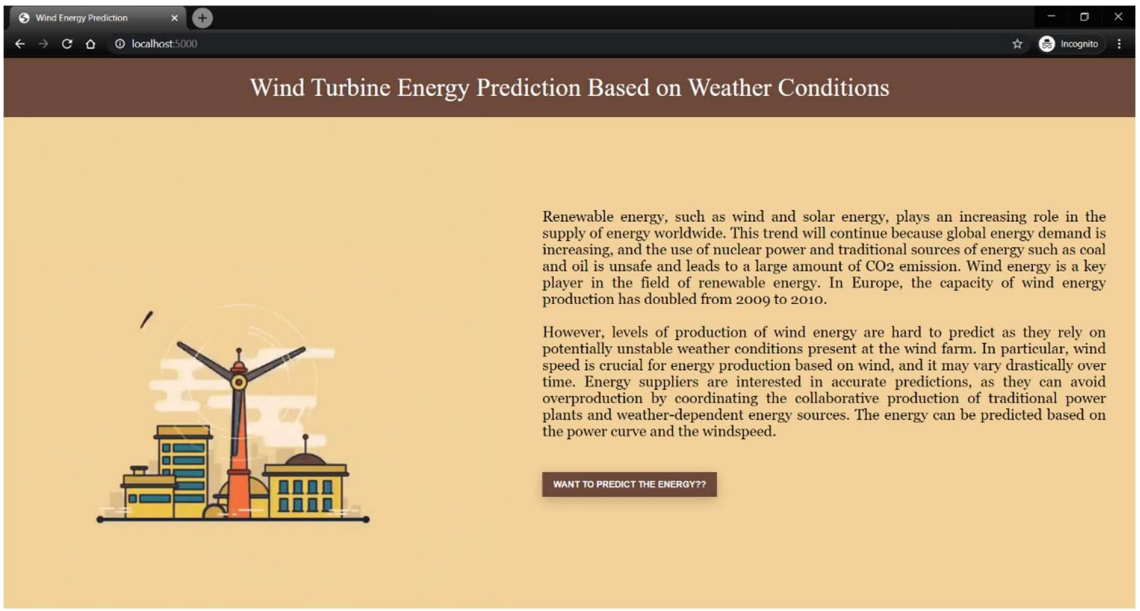
We Build an HTML page to take both the API requests for weather forecasting as well as the energy prediction forms. You can get the HTML page from project folder

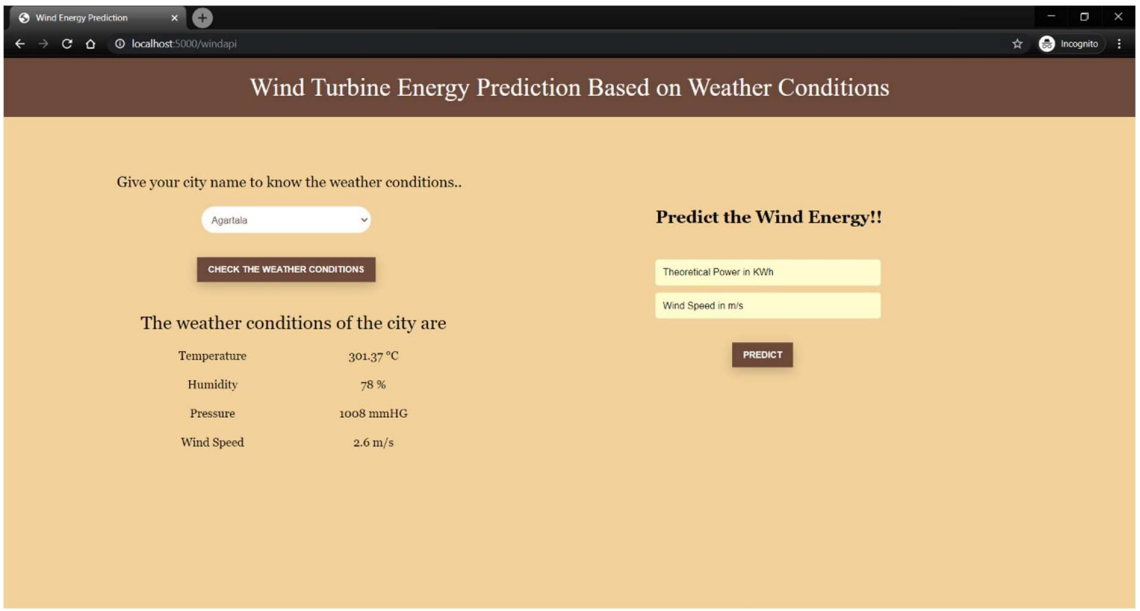
Execute and test your model

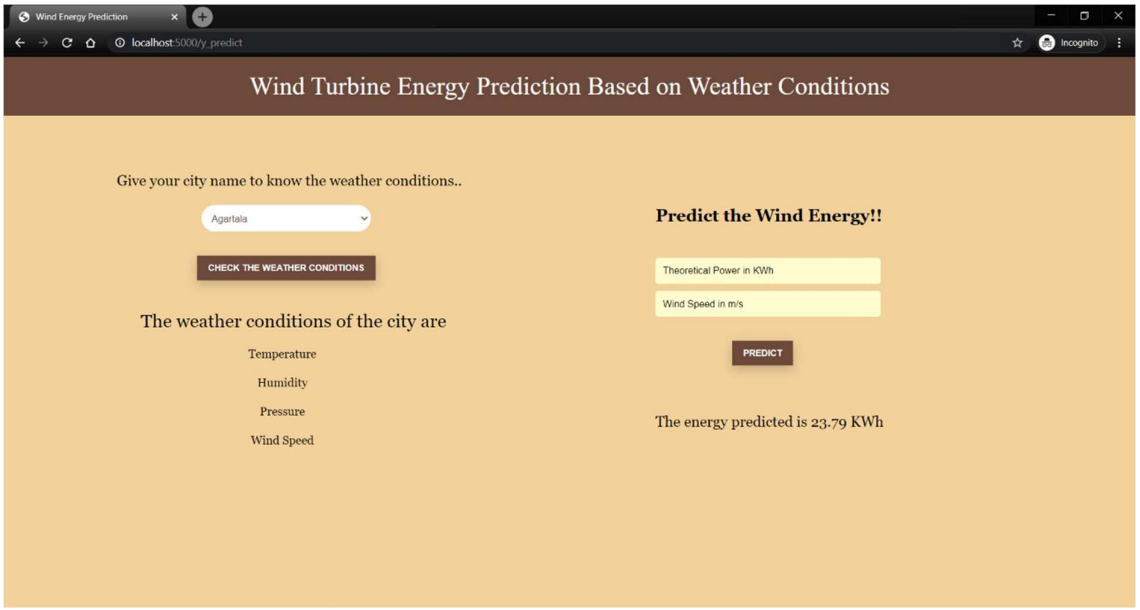
Step 1: Execute the python code and after the model is running, open localhost:5000 page and test it.

```
(myenv) D:\data science\Wind-Mill-Power-Prediction-main\Flask - Wind-Mill-Power-Prediction>python windApp.py
```

```
* Serving Flask app 'windApp' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```


Wind Energy Prediction
localhost:5000
Incognito
Wind Turbine Energy Prediction Based on Weather Conditions
Renewable energy, such as wind and solar energy, plays an increasing role in the supply of energy worldwide. This trend will continue because global energy demand is increasing, and the use of nuclear power and traditional sources of energy such as coal and oil is unsafe and leads to a large amount of CO2 emission. Wind energy is a key player in the field of renewable energy. In Europe, the capacity of wind energy production has doubled from 2009 to 2010.
However, levels of production of wind energy are hard to predict as they rely on potentially unstable weather conditions present at the wind farm. In particular, wind speed is crucial for energy production based on wind, and it may vary drastically over time. Energy suppliers are interested in accurate predictions, as they can avoid overproduction by coordinating the collaborative production of traditional power plants and weather-dependent energy sources. The energy can be predicted based on the power curve and the windspeed.
WANT TO PREDICT THE ENERGY??


Wind Energy Prediction
localhost:5000/windapi
Incognito
Wind Turbine Energy Prediction Based on Weather Conditions
Give your city name to know the weather conditions..
Agartala
CHECK THE WEATHER CONDITIONS
The weather conditions of the city are
Temperature 301.37 °C
Humidity 78 %
Pressure 1008 mmHG
Wind Speed 2.6 m/s
Predict the Wind Energy!!
Theoretical Power in KWh
Wind Speed in m/s
PREDICT


Wind Energy Prediction
localhost:5000/y_predict
Incognito
Wind Turbine Energy Prediction Based on Weather Conditions
Give your city name to know the weather conditions..
Agartala
CHECK THE WEATHER CONDITIONS
The weather conditions of the city are
Temperature
Humidity
Pressure
Wind Speed
Predict the Wind Energy!!
Theoretical Power in KWh
Wind Speed in m/s
PREDICT
The energy predicted is 23.79 KWh