PREDICTING THE ENERGY OUTPUT OF WIND TURBINE BASED ON WEATHER CONDITIONS

Submitted By

P. JAGADEESH

39110394

INDEX

1. INTRODUCTION

a. Overview:

Wind power generation differs from conventional thermal generation due to the stochastic nature of wind. Thus, wind power forecasting plays a key role in dealing with the challenges of balancing supply and demand in any electricity system, given the uncertainty associated with the wind farm power output. Accurate wind power forecasting reduces the need for additional balancing energy and reserve power to integrate wind power. For a wind farm that converts wind energy into electricity power, a real-time prediction system of the output power is significant. In this project, a prediction system is developed with a method of combining statistical models and physical models. In this system, the inlet condition of the wind farm is forecasted by the auto regressive model.

b. Purpose:

We'll be able to understand the problem to classify if it is a regression or a classification kind of problem. We 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. We will be able to know how to build a web application using the Flask framework

2. LITERATURE SURVEY

a. Existing problem:

Now, meteorologists have to manually take down every value and then calculate the value for theoretical power. This is a very time taking process and there are chances for human errors. As this decides how much energy will be produced, any kind of error will cost a huge amount to the government. Also, there is no fixed formula for

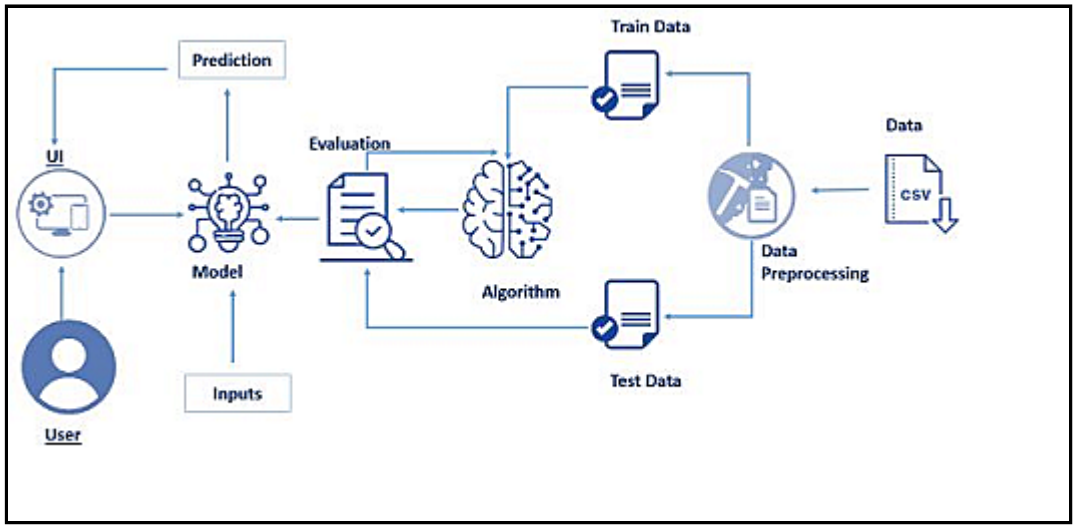
calculating Theoretical power. They depend on a number of factors. Hence, we have to come up with a solution such that the work for meteorologists is decreased and also efficiency is increased.

b. Proposed Solution:

Our aim is to map weather data to energy production. We wish to show that even data that is publicly available for weather stations close to wind farms can be used to give a good prediction of the energy output. Furthermore, we examine the impact of different weather conditions on the energy output of technique to predict the energy output of wind farms. We are building an IBM Watson Auto AI Machine Learning technique to predict the energy output of wind turbine.

2. THEORETICAL ANALYSIS

3.1 Block Diagram



b. Hardware/Software Requirements

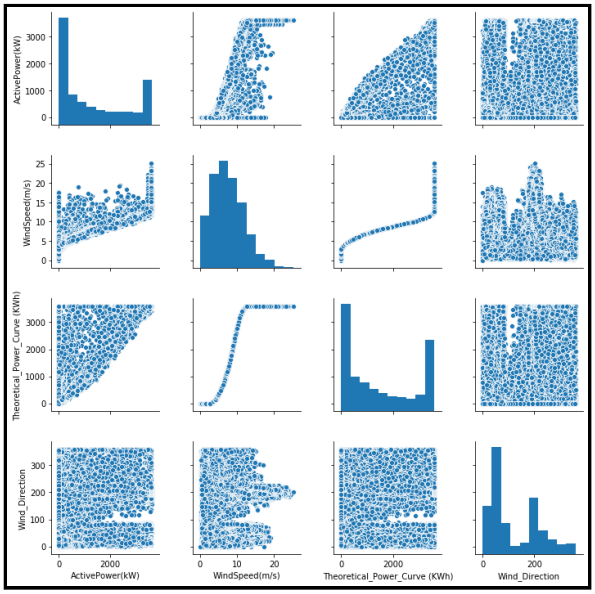
1. Cloud Tool used: IBM
2. API: open weather
1. IDE: Jupyter Notebook, Spyder, Anaconda navigator
2. Programing Language (Back-end): Python 3.7
3. Front-end: HTML, CSS
4. Framework: Flask

4. EXPERIMENTAL INVESTIGATIONS

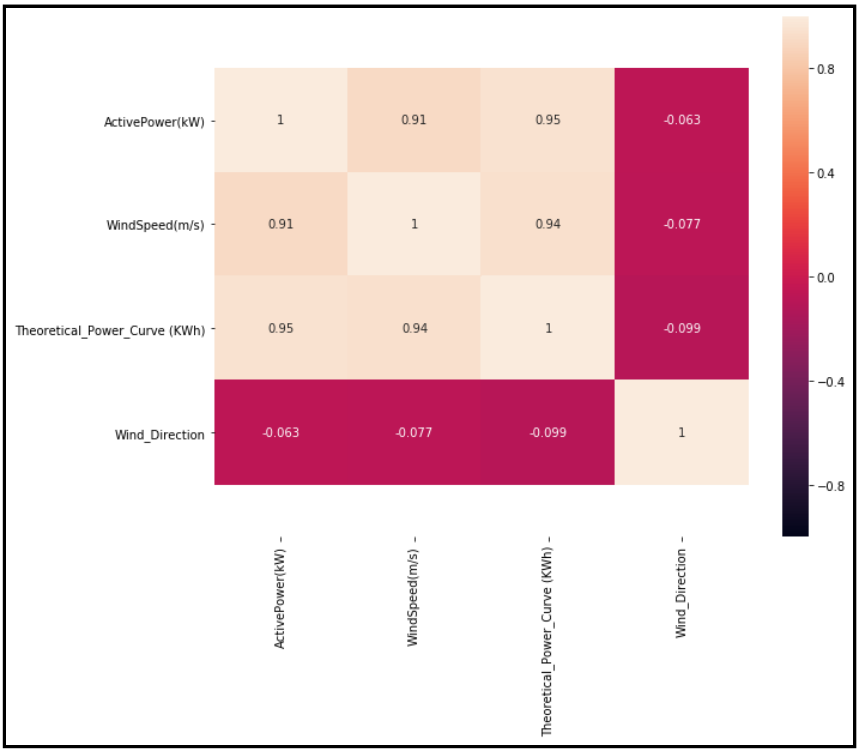
The accuracy of the current power curve method may depend on the distribution of wind speed, turbulence intensity, and shear at the test site, compared to the deployment site. If the test site conditions are similar to the deployment site, the power curve method may give good results. The regression tree method predicts wind turbine energy capture with two to three times more accuracy than the industry standard power curve method, and may be more useful for predictions of energy capture at sites that experience different conditions than the test site. To use the regression tree modelling approach to predict the energy capture of a turbine at a new site, several steps are required.

The label for the data set is PE which is a continuous variable. Data Visualization is one of the powerful parts of Data Science to infer logic form data and find some patterns.

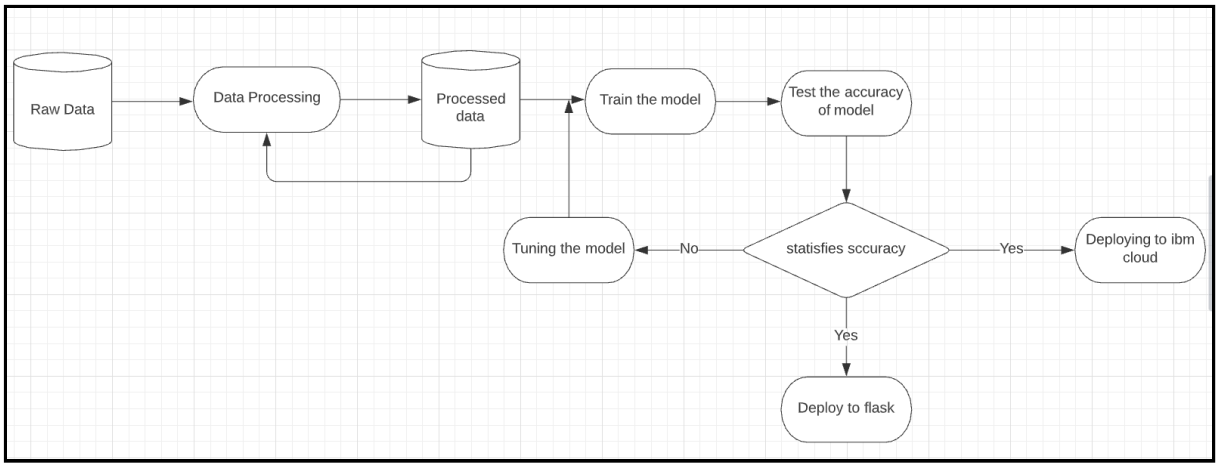
The factors affecting PE are also continuous. So, in order to analyses the dataset using graphs, we use the plots like heatmap.



Heatmap is the plot of values of correlation between the variables in data set. The correlation values are plotted using heatmap.

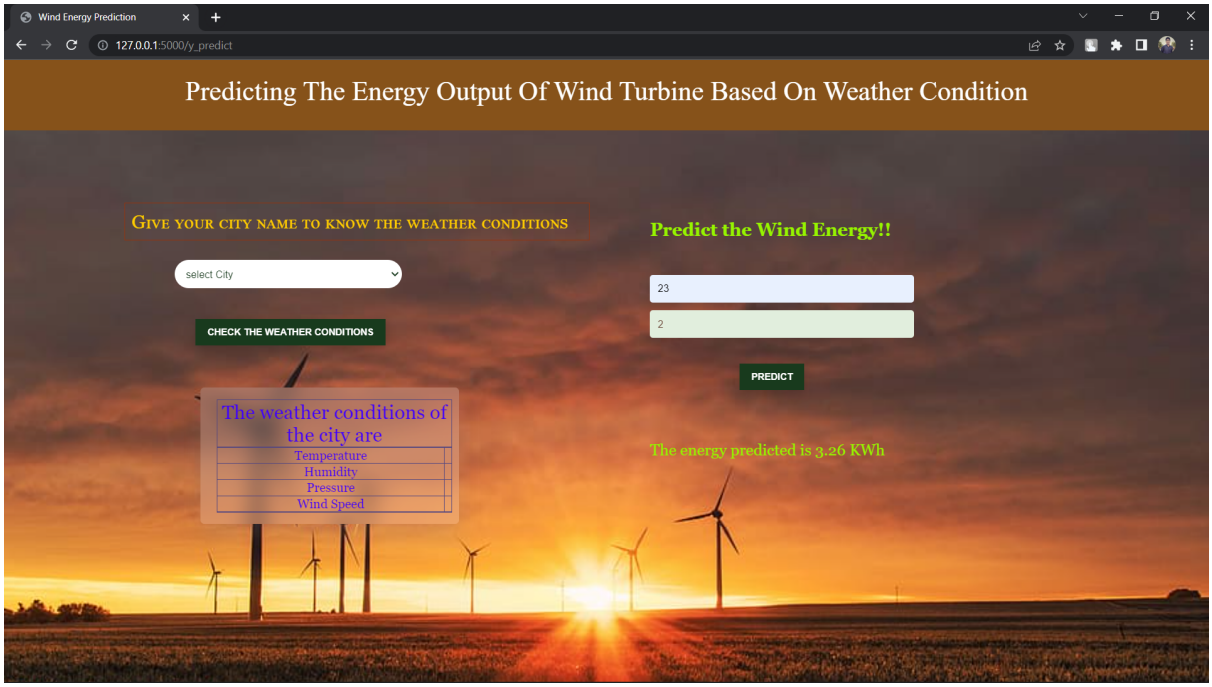


5. FLOW CHART



6. RESULT

We have successfully built the UI interface prediction of weather conditions of the city, in which user can select any of the states from India and then he will get the current climatic conditions in that region. These conditions include temperature, humidity, pressure and wind speed. Next to that we can see a tab for predicting wind energy. Here we have to enter the Active power and wind speed, in return we will get the Theoretical power i.e., the wind energy. As people are more inclined towards non-conventional energy, this UI will help them to find how much energy will be generated and accordingly user can plan their future steps.



7. ADVANTAGES

1. The interface is user friendly and easy to use and understand even to a person who has very little/no knowledge for the same.
2. The model uses a huge amount of data the results generated are accurate and reliable.
3. Easy to use & has a user-friendly interface.
4. Results can be improved by training data to our choice of parameter.
5. We can easily find the weather conditions of cities using Weather API integration.

DISADVANTAGES

1. Some complex integrations of services are required.
2. No free server available on IBM Cloud for deploying Backend.

8. APPLICATIONS

1. Better Power Output Wind power forecasts are important in efficiently using wind turbines for generating power output.

2. This model develops efficiently to predict the electrical energy output in the easy way for the peoples.

3. Efficient to predict the current weather condition of the city.

4. 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
5. Time saving & cost-efficient method as many applications are not needed to implement.
6. As the application is quite robust & resilient in its architecture, it allows us to easily navigate through different sections.

9. CONCLUSION

A Machine Learning model, has been developed to forecast the energy output of the wind turbine and current weather condition of the cities. A simple, efficient and a versatile model is built keeping in mind the diversity of data, computational complexity and overhead involved in making API calls to the model for prediction.

The product can increase the accuracy of the forecasting the output from the wind turbine. Overall accurate wind energy prediction reduces the financial and technical risk of uncertainty of wind energy production for all electricity market participants.

10.FUTURE SCOPE

1. Despite our model giving good results, we can add robustness to it by making it do the predictions for a greater time in the future.
2. The model can be developed to make predictions on other sources of data like solar energy, tidal energy etc.

11. BIBLIOGRAPHY

- Weather api - https://openweathermap.org
- Long-term wind speed and power forecasting using local recurrent neural network models IEEE Trans. Energy Convers.
- Brower M 2012 Wind Resource Assessment: A Practical Guide to Developing a Wind Project (New York: Wiley)

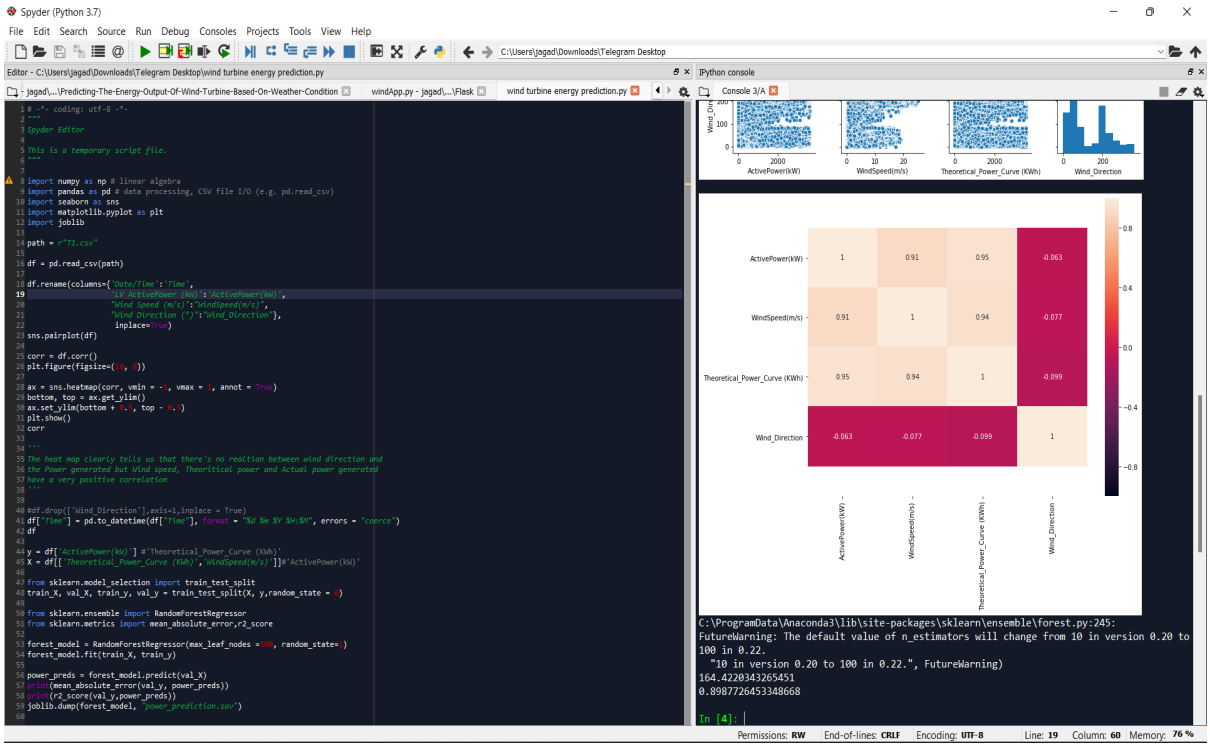
12. APPENDIX

a. Source code:

Windapp.py

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

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

@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="86b1a085e43cad23bfd9c45d5fd88fc3"
    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"]) - 273.15) +" °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)

@app.route('/y_predict',methods=['POST'])
def y_predict():
    '''
    For rendering results on HTML GUI
```

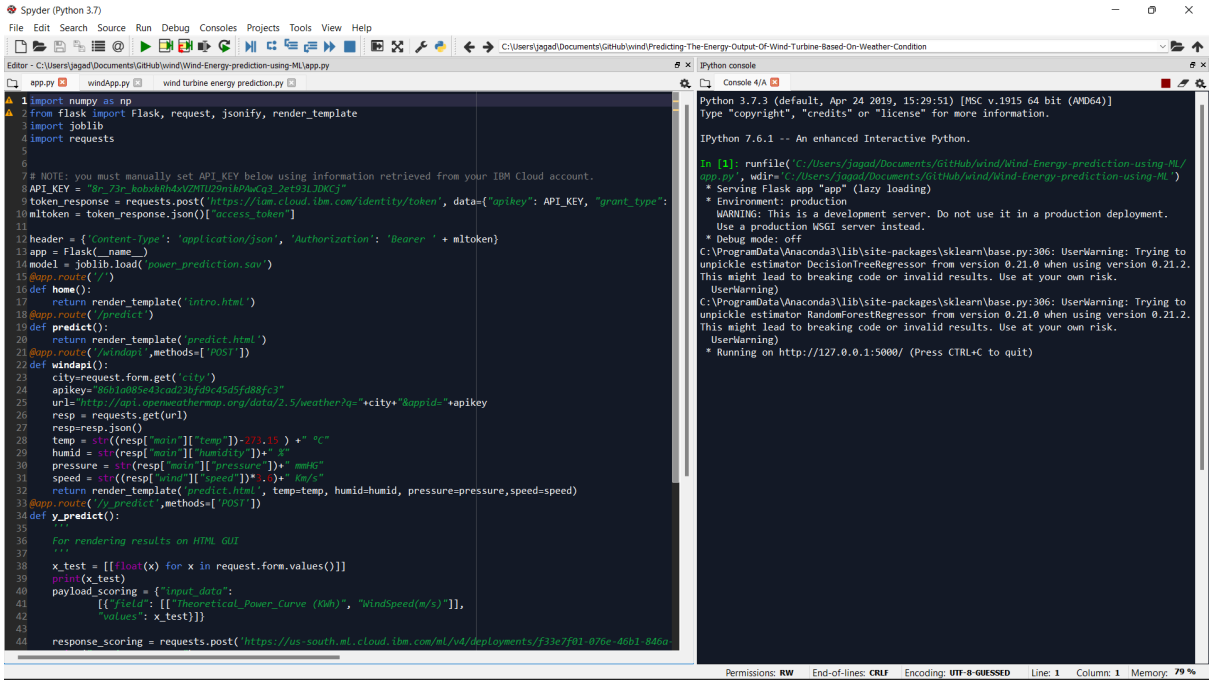
```
    '''
    x_test = [[float(x) for x in request.form.values()]]

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



App.py

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


# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud
account.
API_KEY = "8r_73r_kobxkRh4xVZMTU29nikPAwCq3_2et93LJDKCj"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
app = Flask(__name__)
model = joblib.load('power_prediction.sav')
@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="86b1a085e43cad23bfd9c45d5fd88fc3"
    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"])-273.15) +" °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)
@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()]]
    print(x_test)
    payload_scoring = {"input_data":
          [{"field": [["Theoretical_Power_Curve (KWh)", "WindSpeed(m/s)"]],
          "values": x_test}]}

    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/f33e7f01-076e-46b1-846a-
7a3a74afa11b/predictions?version=2021-10-28', json=payload_scoring, headers={'Authorization':
'Bearer ' + mltoken})
    print("Scoring response")
    predictions =response_scoring.json()
    print(predictions)
    print('Final Prediction Result',predictions['predictions'][0]['values'][0][0])


    pred =response_scoring.json()
    print(pred)
    #print('Final Prediction Result',predictions['predictions'][0]['values'][0][0])

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


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


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\jagad\Documents\GitHub\wind\Predicting-The-Energy-Output-Of-Wind-Turbine-Based-On-Weather-Condition
Editor - C:\Users\jagad\Documents\GitHub\wind\Wind-Energy-prediction-using-ML\app.py
IPython console
app.py
windApp.py
wind turbine energy prediction.py
Console 4/A
import numpy as np
from flask import Flask, request, jsonify, render_template
import joblib
import requests
# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "8r_73r_kobxkRh4xVZMIU29nikPAwCq3_2et93LJDKCj"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey": API_KEY, "grant_type":
mltoken = token_response.json()["access_token"]
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
app = Flask(__name__)
model = joblib.load('power_prediction.sav')
@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="8bb1a085e43ca23bfd9c45d5fd88fc3"
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"])-273.15) +" °C"
humid = str(resp["main"]["humidity"])+" %"
pressure = str(resp["main"]["pressure"])+" mmHG"
speed = str((resp["wind"]["speed"])*3.6)+" Km/s"
return render_template('predict.html', temp=temp, humid=humid, pressure=pressure,speed=speed)
@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()]]
print(x_test)
payload_scoring = {"input_data":
[{"field": [["Theoretical_Power_Curve (KWh)", "WindSpeed(m/s)"]],
"values": x_test}]}
response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/f33e7f01-076e-46b1-846a-
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 7.6.1 -- An enhanced Interactive Python.
In [1]: runfile('C:/Users/jagad/Documents/GitHub/wind/Wind-Energy-prediction-using-ML/app.py', wdir='C:/Users/jagad/Documents/GitHub/wind/Wind-Energy-prediction-using-ML')
* Serving Flask app "app" (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
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:306: UserWarning: Trying to unpickle estimator DecisionTreeRegressor from version 0.21.0 when using version 0.21.2. This might lead to breaking code or invalid results. Use at your own risk.
UserWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:306: UserWarning: Trying to unpickle estimator RandomForestRegressor from version 0.21.0 when using version 0.21.2. This might lead to breaking code or invalid results. Use at your own risk.
UserWarning)
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
Permissions: RW
End-of-lines: CRLF
Encoding: UTF-8-GUESSED
Line: 1
Column: 1
Memory: 79 %

b. UI Output Screenshot:

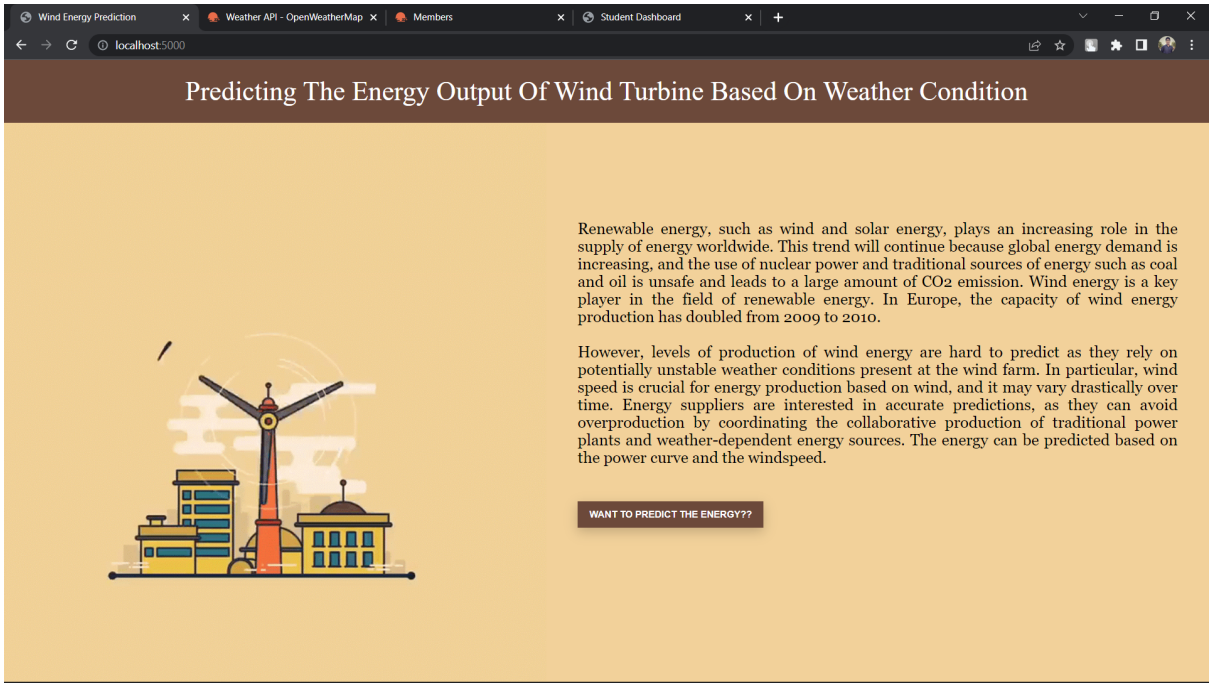


FIG:1 HOME PAGE

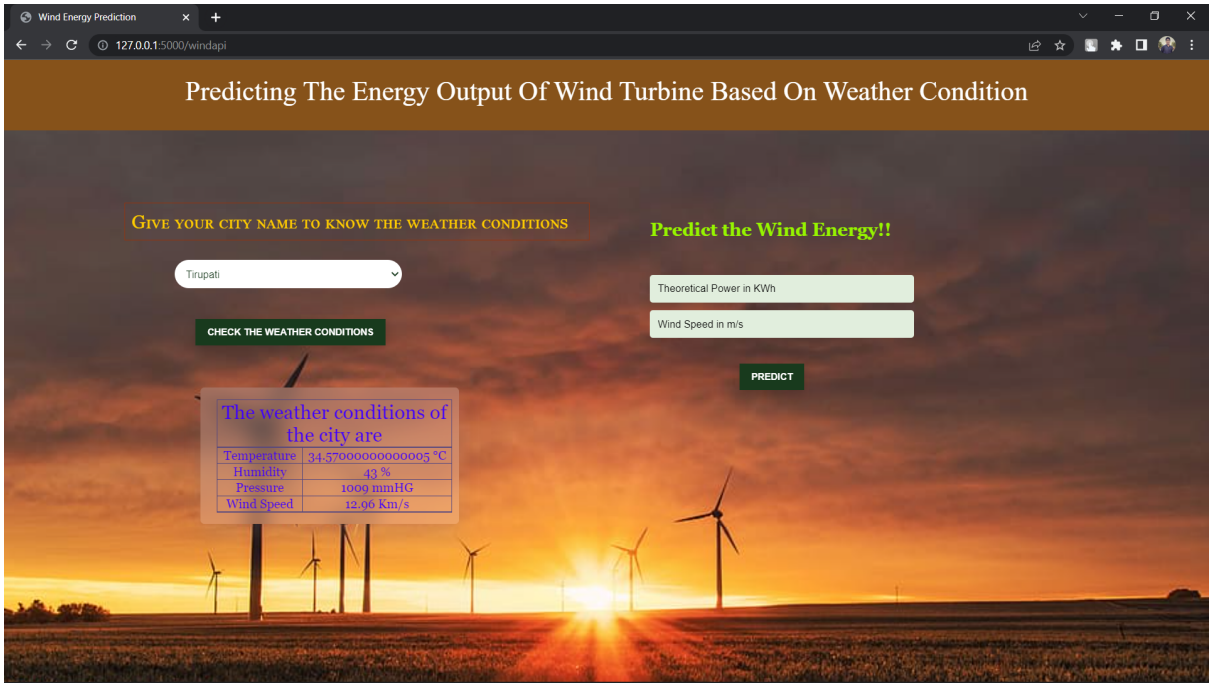


FIG:2 Weather Analysis Dashboard

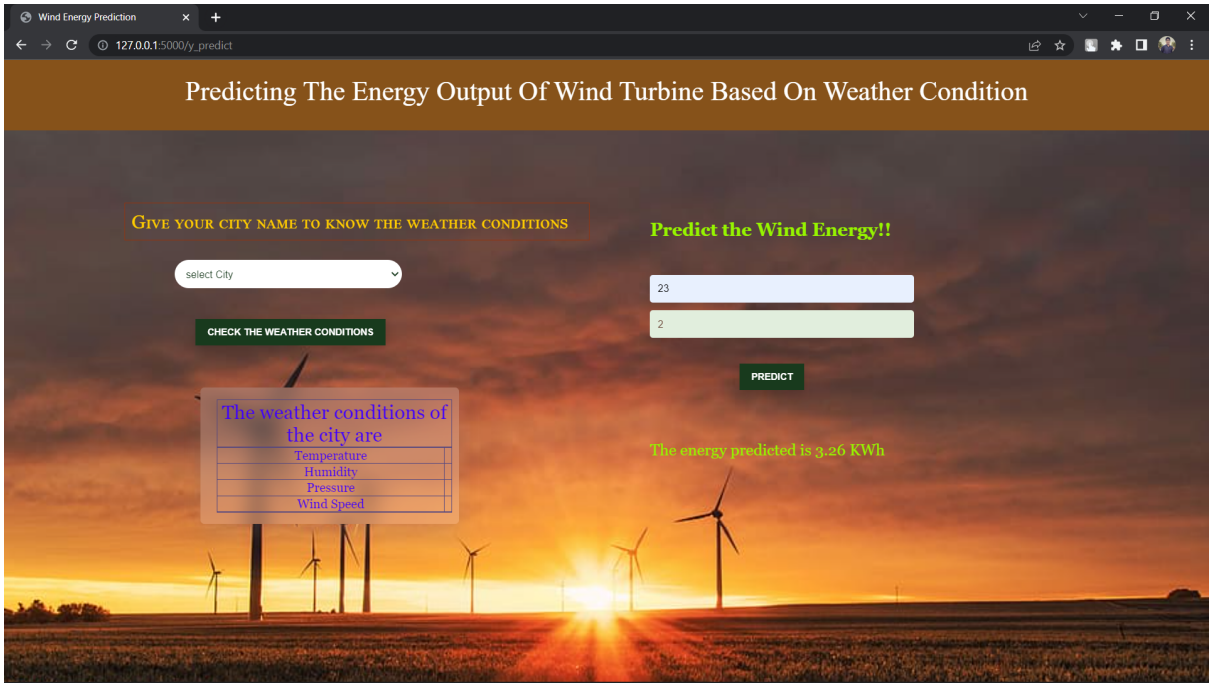


Fig.3: Wind Energy Analysis Dashboard

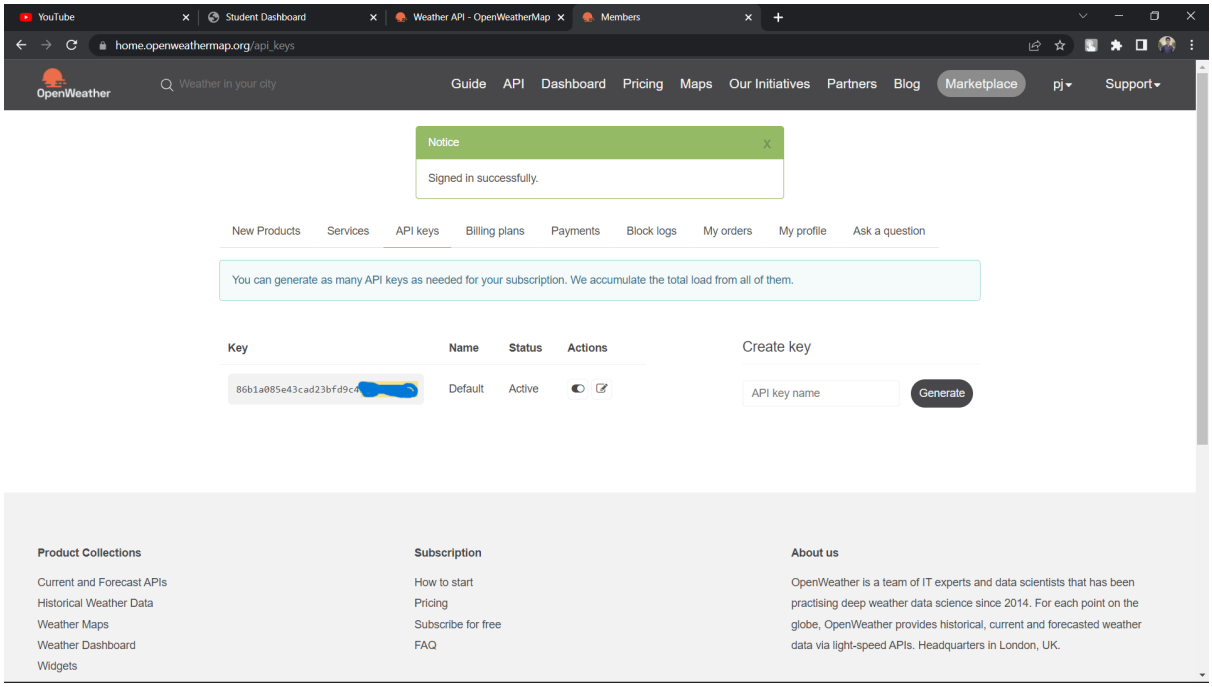


FIG: 4 Openweathermap API KEY

THANK YOU

Submitted By

P. JAGADEESH

39110394