# ABSTRACT

Wind energy plays an increasing role in the supply of energy worldwide. The energy output of a wind farm is highly dependent on the weather conditions present at its site. If the output can be predicted more accurately, energy suppliers can coordinate the collaborative production of different energy sources more efficiently to avoid costly overproduction. In this paper, we take a computer science perspective on energy prediction based on weather data and analyze the important parameters as well as their correlation on the energy output. To deal with the interaction of the different parameters, we use random forest regression of machine learning algorithms. Our studies are carried out on publicly available weather and energy data for a wind farm. We report on the correlation of the different variables for the energy output. The model obtained for energy prediction gives a very reliable prediction of the energy output for supplied weather data.

|  |  |  |
| --- | --- | --- |
|  | **TABLE OF CONTENTS** |  |

**CHAPTER NO TITLE PAGE NO**

# ABSTRACT

# LIST OF FIGURES 9

**1** **INTRODUCTION** 1

1.1 Motivation 2

1.2 Problem Definition 2

**2 AIM AND SCOPEOF THE**

**PRESENT INVESTIGATION**  3

**3 EXPERIMENTAL OR**

**MATERIALS AND METHODS**

**ALGORITHMSUSED**

### 3.1 Prerequisites 5

### 3.2 Project Flow 5

### 3.3 Data Collection 5

3.4 Data Pre-Processing 6

3.5 Model Building 6

**4 RESULTS AND DISCUSSION,**

**PERFORMANCE ANALYSIS**

### 4.1 Project Structure 11

4.2 Application Building 11

**5 SUMMARY AND CONCLUSIONS**

5.1 SUMMARY 15

5.2 CONCLUSION 15

5.3 APPENDIX 16

A. SCREEN SHOTS 16

B. SOURCE CODE 22

5.3.1 PYTHON CODE USED

IN

JUPYTER NOTEBOOK 22

5.3.2 PYTHON CODE USED

FOR

APP BUILDING 27

* + 1. HTML CODES USED 29
* Intro.html 29
* Predict.html 32

**6 REFERENCES** 39

**LIST OF FIGURES**

**FIGURE NO FIGURE NAME PAGE NO**

3.4.1 Imported libraries 7

3.4.2 Datasets are imported as data frames 7

3.4.3 Correlation between the columns 8

3.4.4 Splitting Data Intro Independent And 8

Dependent Variables

3.5.1 Considering the Random Forest Regressor 9

### 3.5.2 Check the Metrics Of The Model 9

3.5.3 Saving the model 9

4.1.1 Project folder 11

4.2.1 Import libraries 12

4.2.2 Initialize flask app 12

4.2.3 Configure app.py for api requests 12

4.2.4 Configure the file with predictions 12

4.2.5 Main page 13

4.2.6 Enter commands & Run the app 13

4.2.7 Open localhost:5000 page and test it. 13

4.2.8 Weather and wind prediction page 14

5.3.1 Juypter code 16

5.3.2 Save the model 17

5.3.3 Intro.html code 17

5.3.4 Predict.html code 18

5.3.5 WindApp.py 19

5.3.6 Main page 20

5.3.7 Weather prediction and

Wind Energy calculation page 21

5.5.1 URL Generated page 29

**CHAPTER 1**

**INTRODUCTION**

The world’s demand for energy is increasingly growing day by day. Renewable energy has attracted attention globally because resources of renewable energy such as solar, wind, geothermal, biomass energy and hydropower are clean, green, free of costs, low carbon and naturally exist in a wide geographical area. Furthermore, usage of these renewable resources decreases environmental pollution by protecting the ecological environment and they can be recycled in nature.

The wind is one of the most useful and important resources of renewable energy. Wind energy has the opportunity to produce power for every hour and it is a clean and popular way for electricity generating owing to its wide availability. Therefore, wind turbines have a crucial role in the electricity generation portfolio worldwide.

However, wind power forecasting is hard to predict because the wind speed is a weather depended parameter and it is highly unstable, random and volatile. It shows strong randomness in a short period time due to its unstable nature and uncontrollability of the wind flows. Random wind power generation causes an imbalance between power generation and consumption, so people who use this energy are affected by the increase of unstable costs due to low predictability

Wind speed/power has received increasing attention around the earth due to its renewable nature as well as environmental friendliness. With the global installed wind power capacity rapidly increasing, the wind industry is growing into a large-scale business. Reliable short-term wind speed forecasts play a practical and crucial role in wind energy conversion systems, such as the dynamic control of wind turbines and power system scheduling. A precise forecast needs to overcome problems of variable energy production caused by fluctuating weather conditions. Power generated by wind is highly dependent on the wind speed. Though it is highly non-linear, wind speed follows a certain pattern over a certain period of time. We exploit this time series pattern to gain useful information and use it for power prediction.

.

**1.1 Motivation**:

Weather-Based Prediction Of Wind Turbine Energy Output: A Next-Generation Approach To Renewable Energy Management

* 1. **Problem Definition**
* Research objectives are prepared after correlating the various works done by contemporary researchers. Most of the researchers have developed methods for wind speed based forecasting.
* Apart from wind speed forecasting, many other parameters required to assess the wind energy potential are studied.
* Meteorological information coupled with topographical data has to be utilized
* for wind power estimation at a particular place. After this, wind turbine power curves have to be mapped against the wind parameters.
* This is to establish the number of energy units that can begenerated at wind plant level, for a month

**CHAPTER 2**

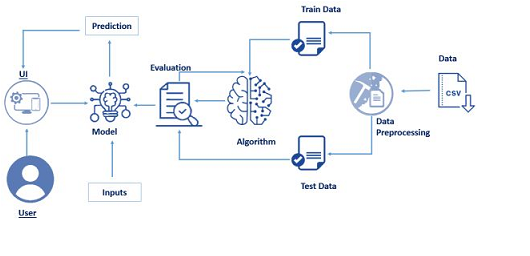
**AIM AND SCOPE OF THE PRESENT INVESTIGATION**

**Aim :**

Wind energy is among the most relevant types of renewable energy and plays a vital role in the projected European energy mix for 2020. The aim of this paper is to comprehensively present current risks and risk management solutions of renewable energy projects and to identify critical gaps in risk transfer, thereby differentiating between onshore and offshore wind parks with focus on the European market. Our study shows that apart from insurance, diversification, in particular, is one of the most important tools for risk management and it is used in various dimensions, which also results from a lack of alternative coverage. Furthermore, policy and regulatory risks appear to represent a major barrier for renewable energy investments, while at the same time, insurance coverage or alternative risk mitigation is strongly limited. This emphasizes the need for new risk transfer solutions to ensure a sustainable growth of renewable energy.

**Scope:**

Proper wind mapping is required before any wind farm can be built. During the assessment of urban wind resources, high investment costs, data availability, and uncertainty all pose challenges . The optimal design conditions are determined by the fluctuation of wind speed with time constants and the reaction time of the wind turbine. For proper utilization of wind energy, different methods are applied depending on the size of utilization systems, as in many large energy systems LOEE is used . In contrast, the probabilistic techniques LLU (loss of the largest unit) are used in small isolated systems to determine capacity requirements

** Fig: Technical Architecture**

**CHAPTER 3**

**EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMSUSED**

### 3.1 Prerequisites:

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

* **Anaconda Navigator**
* **Packages**

**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.

### 3.2 Project Flow:

You will go through all the steps mentioned below to complete the project.

* 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.

### 3.3 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.

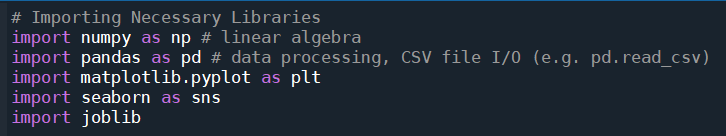
### 3.4Data Pre-Processing

**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.

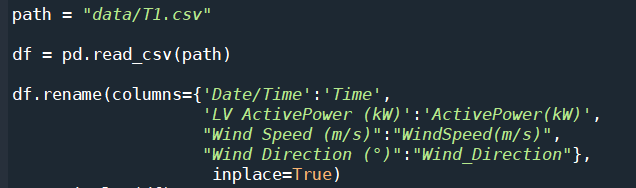


**Fig 3.4.1 imported libraries**

* **Analyze The Datasets:**

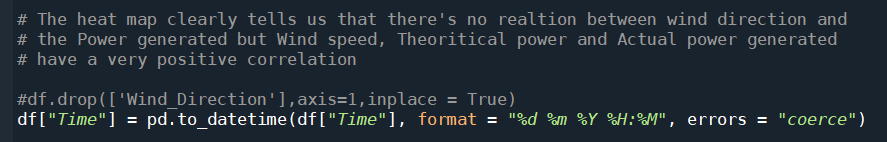
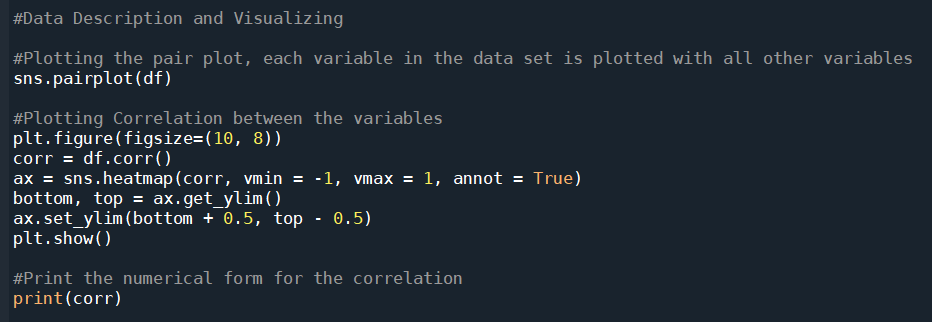
**Step 1**: The datasets are imported as data frames using the pandas library Rename the columns with suitable column names for better understanding

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



**Fig 3.4.2 datasets are imported as data frames**

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

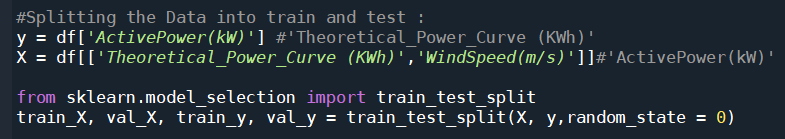


**Fig 3.4.3 correlation between the columns**

### 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.

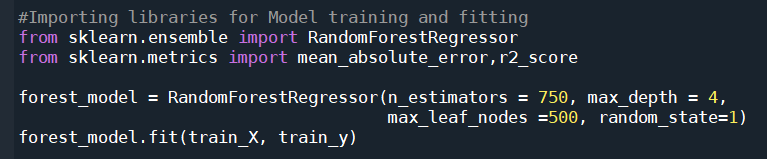
**Fig 3.4.4 Splitting Data Into Independent And Dependent Variables**

### 3.5 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

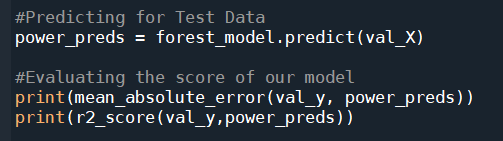
### Choose The Appropriate Model

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

**Fig 3.5.1 considering the Random Forest Regressor**

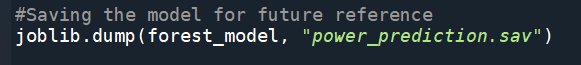
### Check The Metrics Of The Model:

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.



### Fig 3.5.2 Check The Metrics Of The Model

### 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.

**Fig 3.5.3 saving the model**

### API Integration

**Step 1**: Signup for OpenWeather API for current weather forecasting.To signup [click here](https://home.openweathermap.org/users/sign_up).

**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.

**CHAPTER 4**

**RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS**

### 4.1 Project Structure:

### Create a Project folder which contains files as shown below

**Fig 4.1.1 Project folder**

* 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.

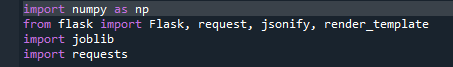
### 4.2 Application Building:

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



**Fig 4.2.1 Import libraries**

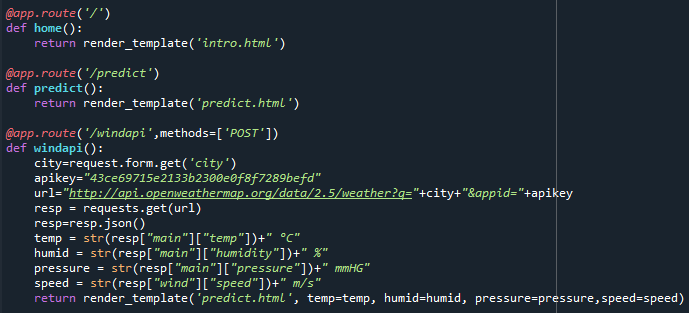
**Step 2:** Load the model and initialise flask app



**Fig 4.2.2 initialise flask app**

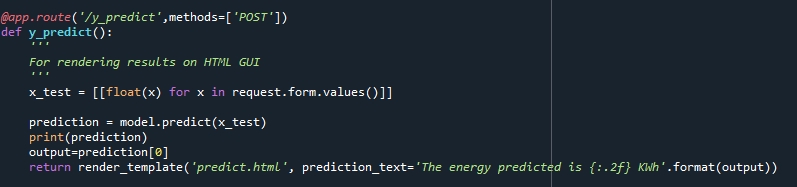
**Step 3:** Configure app.py for api requests

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

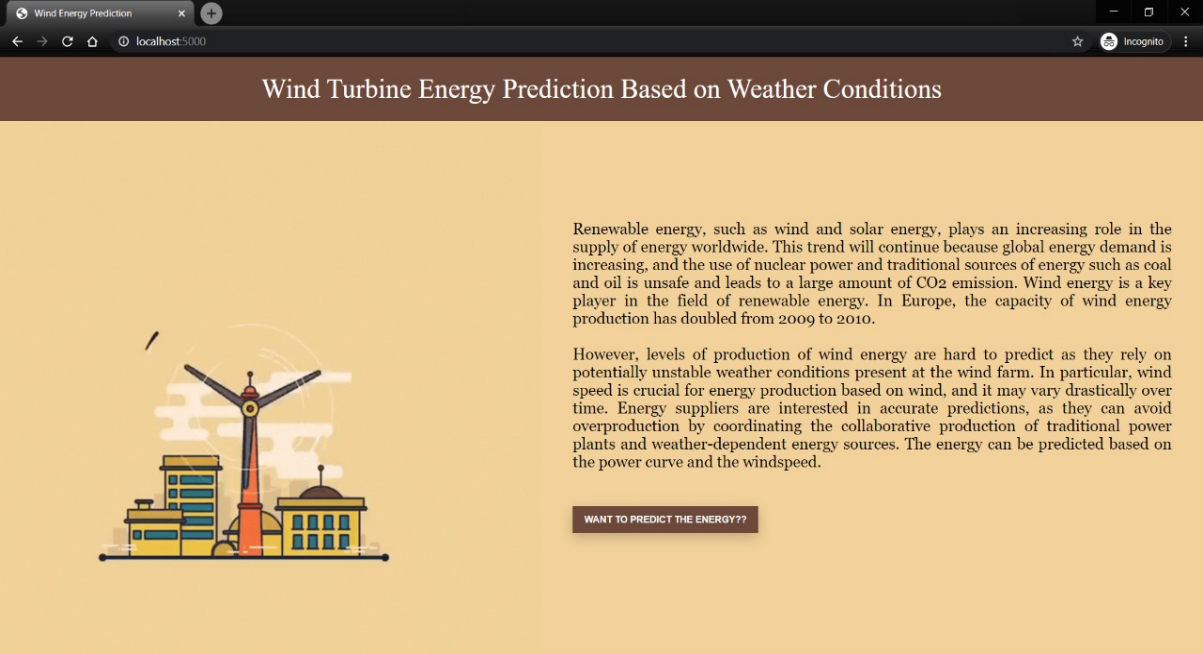


**Fig 4.2.3 Configure app.py for api requests**

**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.

**Fig 4.2.4 Configure the file with predictions**

****

**Fig 4.2.5 Main page**

**Step 5:** Run the app

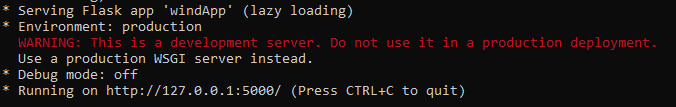
Enter commands as shown below

**Fig 4.2.6 Enter commands & Run the app**

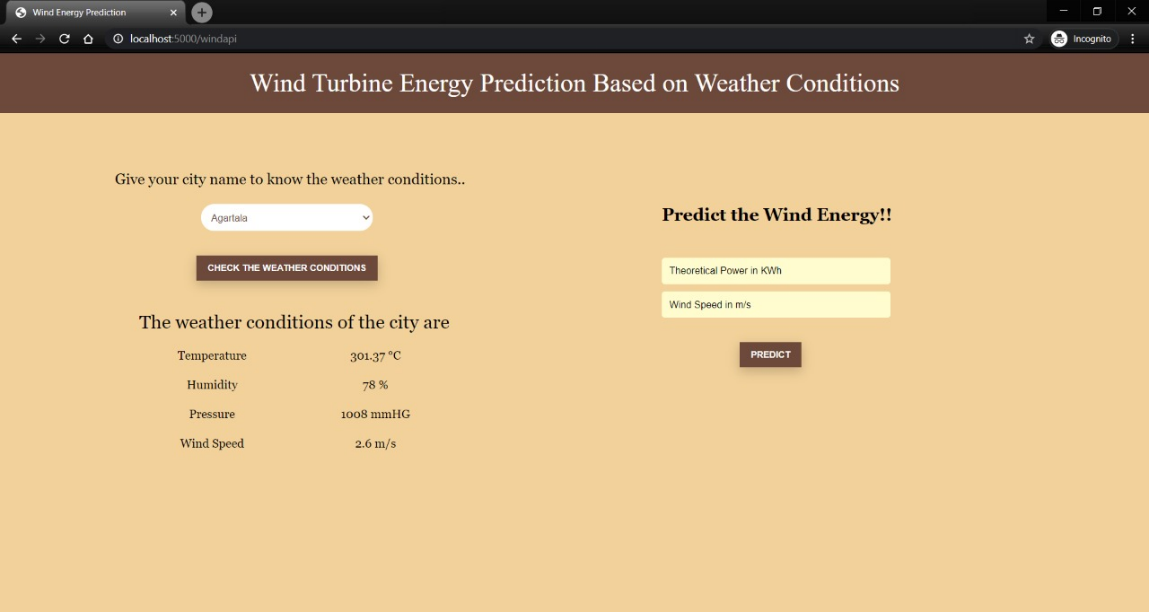
### Build an HTML Page

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.



**Fig 4.2.7 open localhost:5000 page and test it.**



**Fig 4.2.8 weather and wind prediction page**

**CHAPTER 5**

**SUMMARY AND CONCLUSIONS**

**5.1 SUMMARY:**

Electric power generation is mainly depends on the wind energy. The wind speed prediction has an important place in wind energy systems and to drive turbines that further helpful for generating electricity. Wind power forecasting is primarily depends on the forecasting of wind speed.

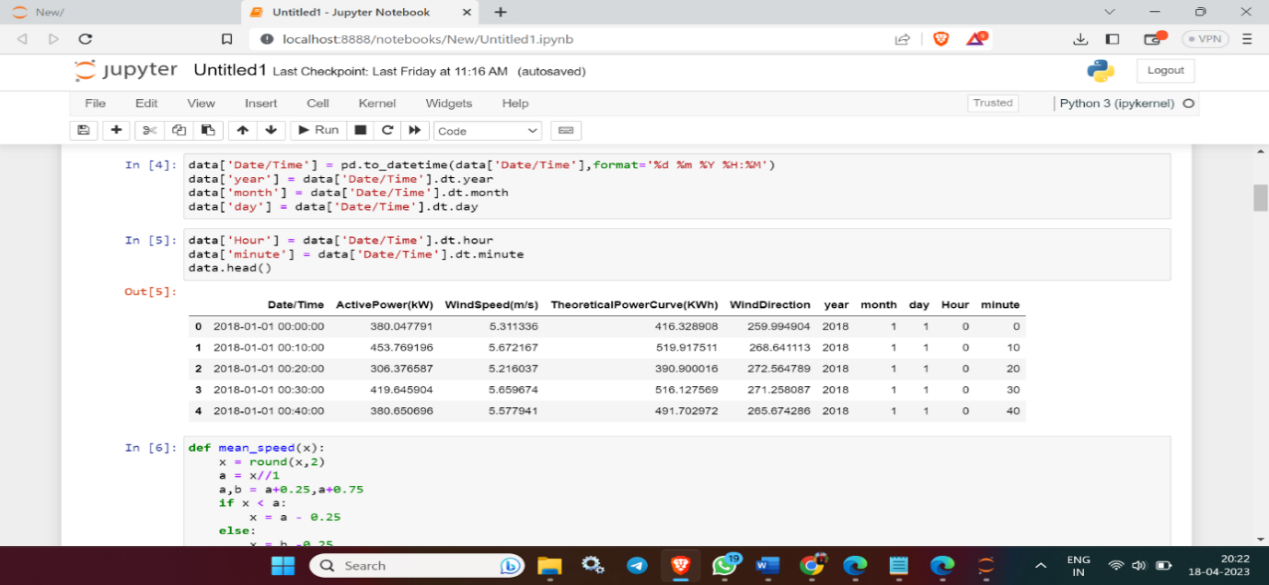
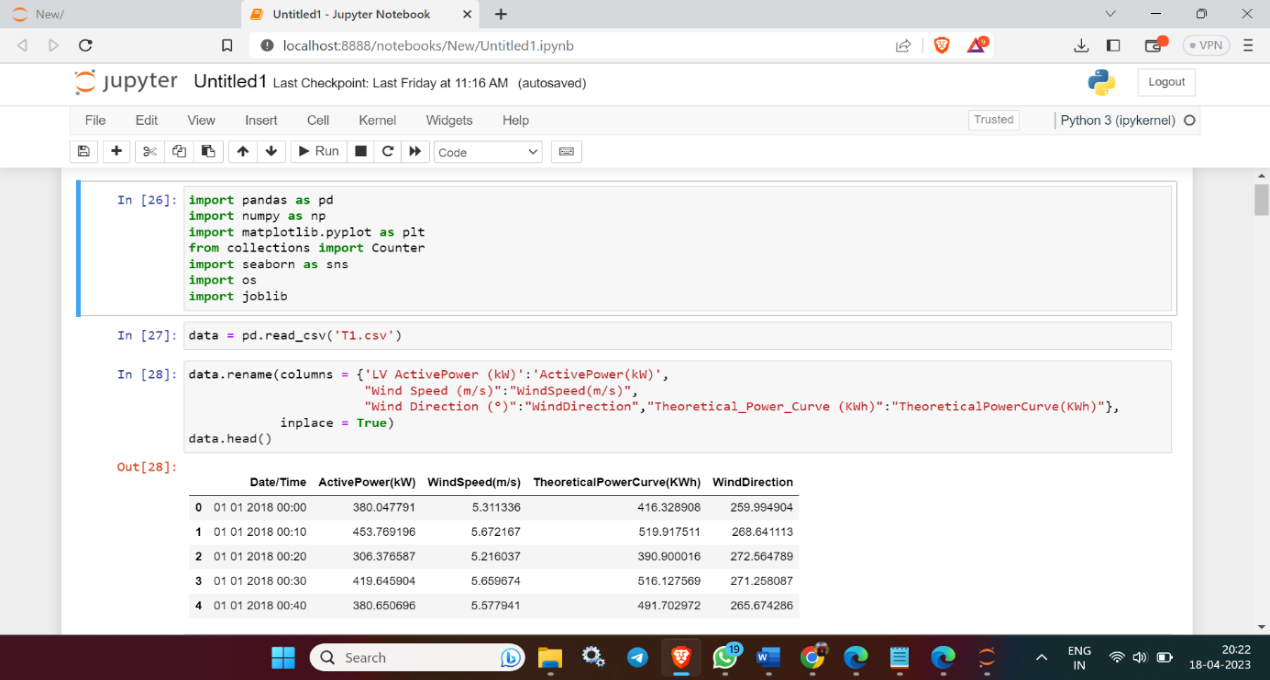
Accurate forecasting results have significant influence on the economy. Recently, academia and industry have paid more attention to wind speed forecasting. More accurate forecasting could reduce costs and risks, improve the security of power systems.

**5.2 Conclusion:**

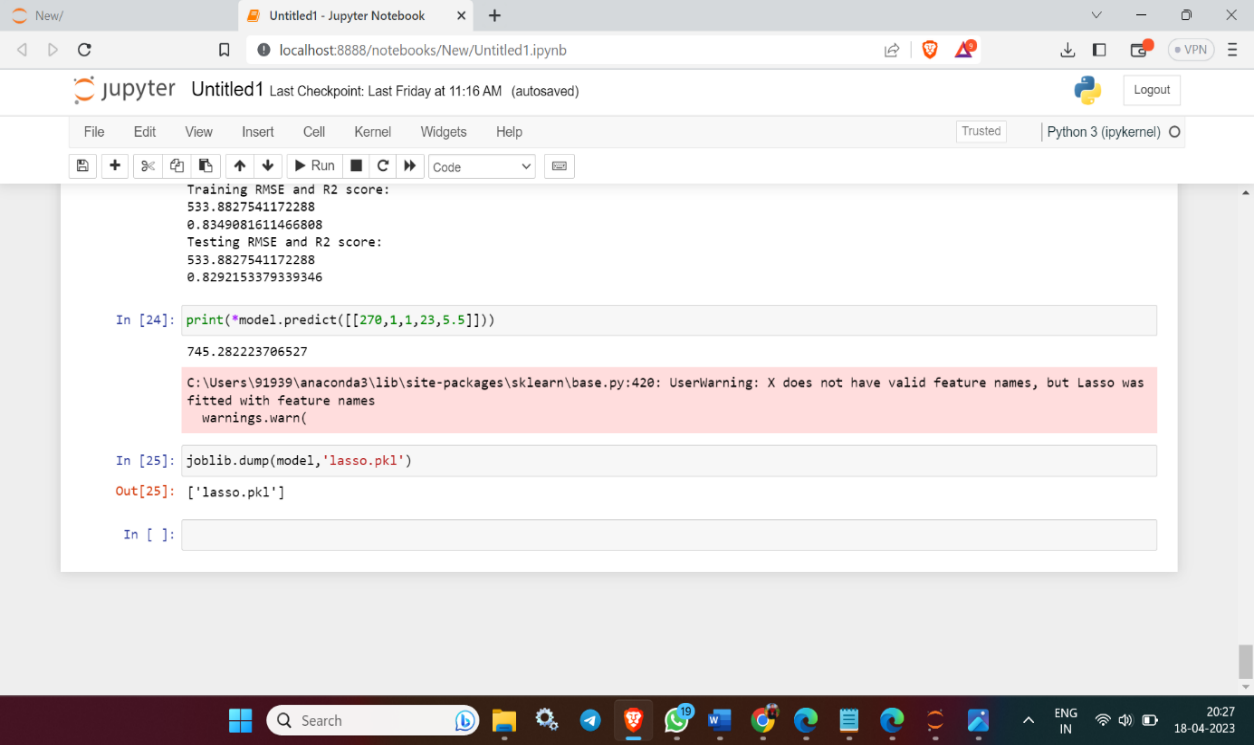
We started with the aim of improving the predictions of power generated using wind energy and we have achieved that using LSTM as machine learning model andperforming model optimization on it. We have also observed that if the wind speed is less than 4 m/s the power generated by the system is zero. LSTM is not able to learn this pattern as this is not the part which it can understand in time series analysis. So, if a hybrid new model is created which can work as the combination of Decision Tree/Random Forest and LSTM we can improve upon these results as well.

**5.3 APPENDIX:**

**A.SCREENSHOTS:**

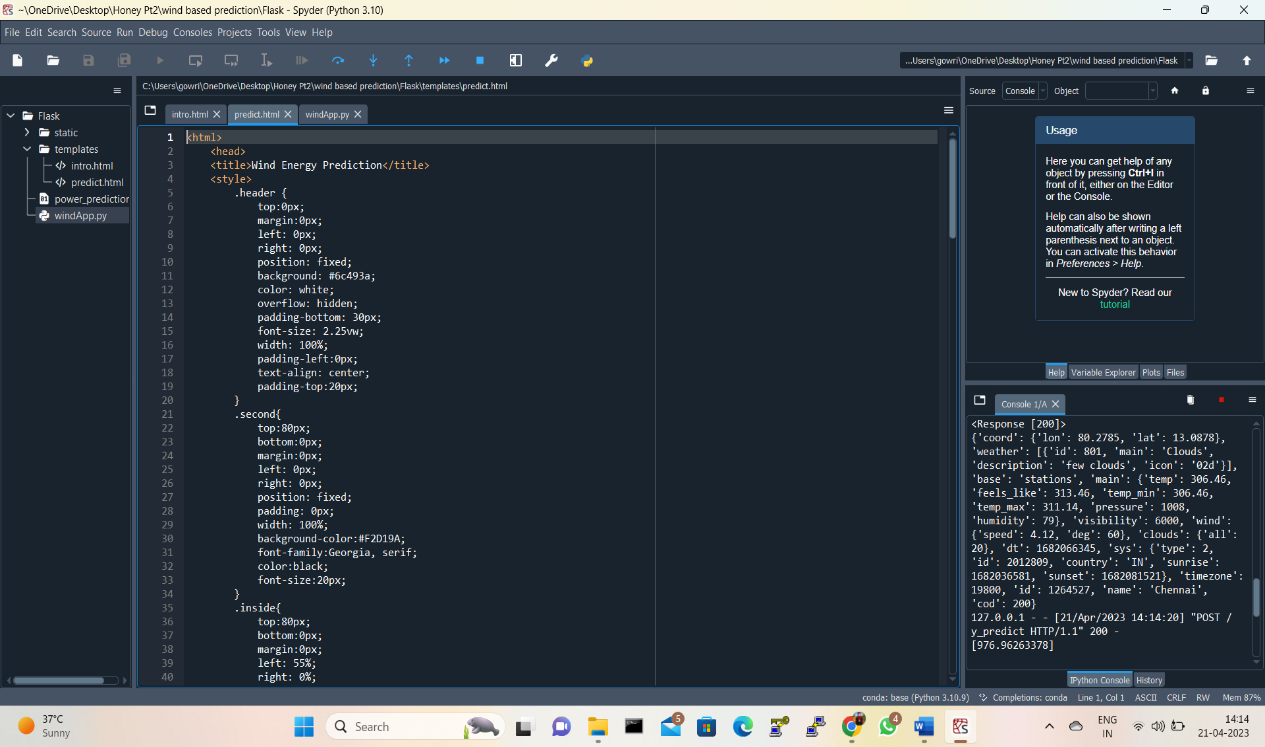


**5.3.1 juypter code**

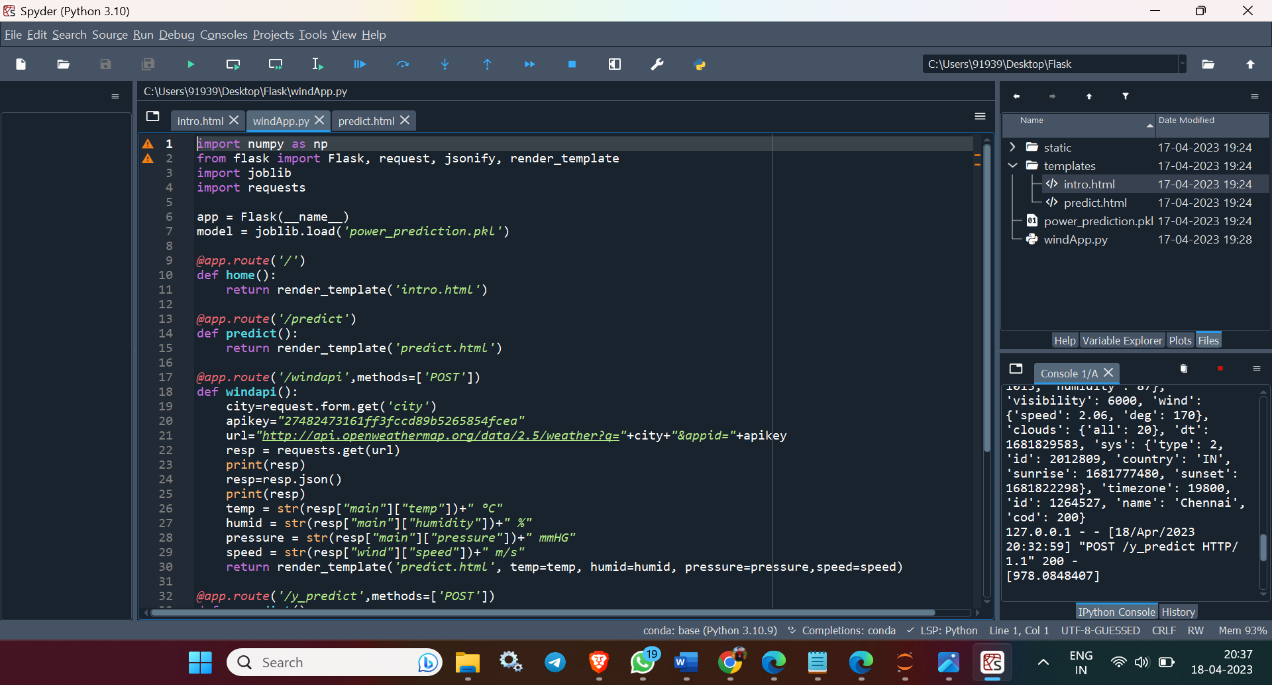
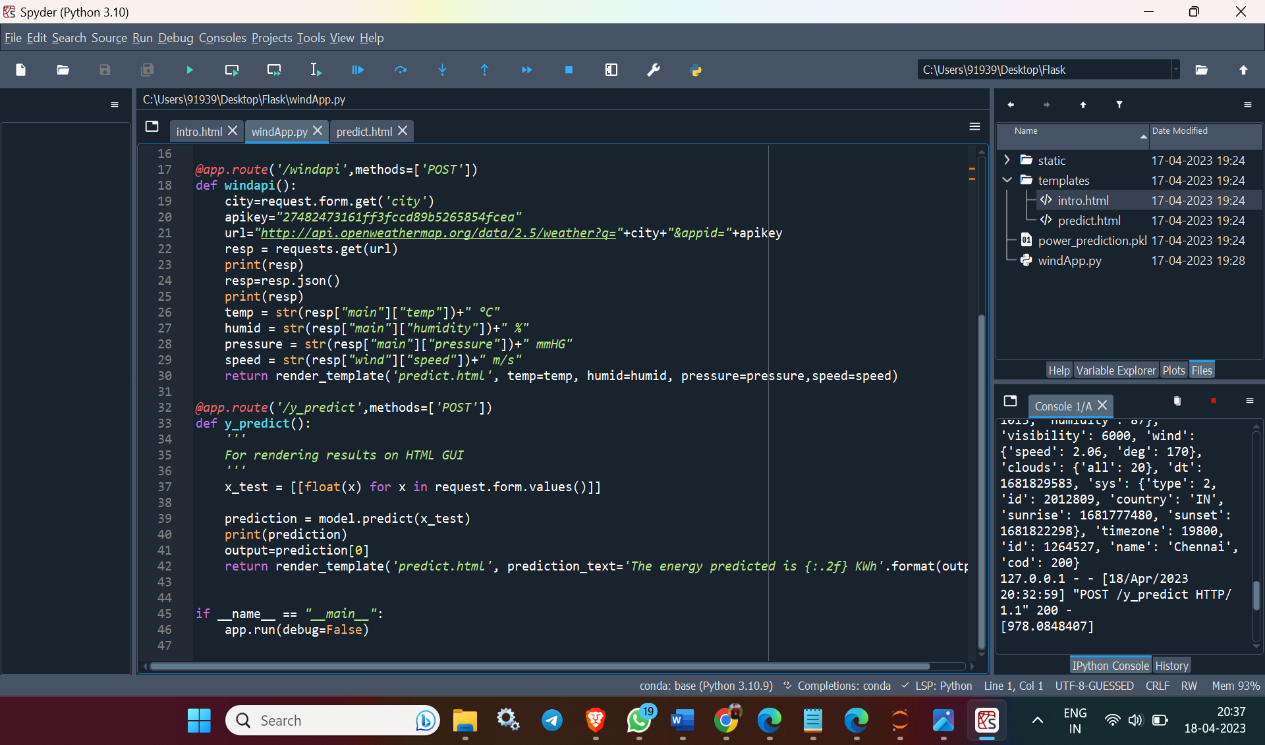
  **Fig 5.3.2 Save the model**



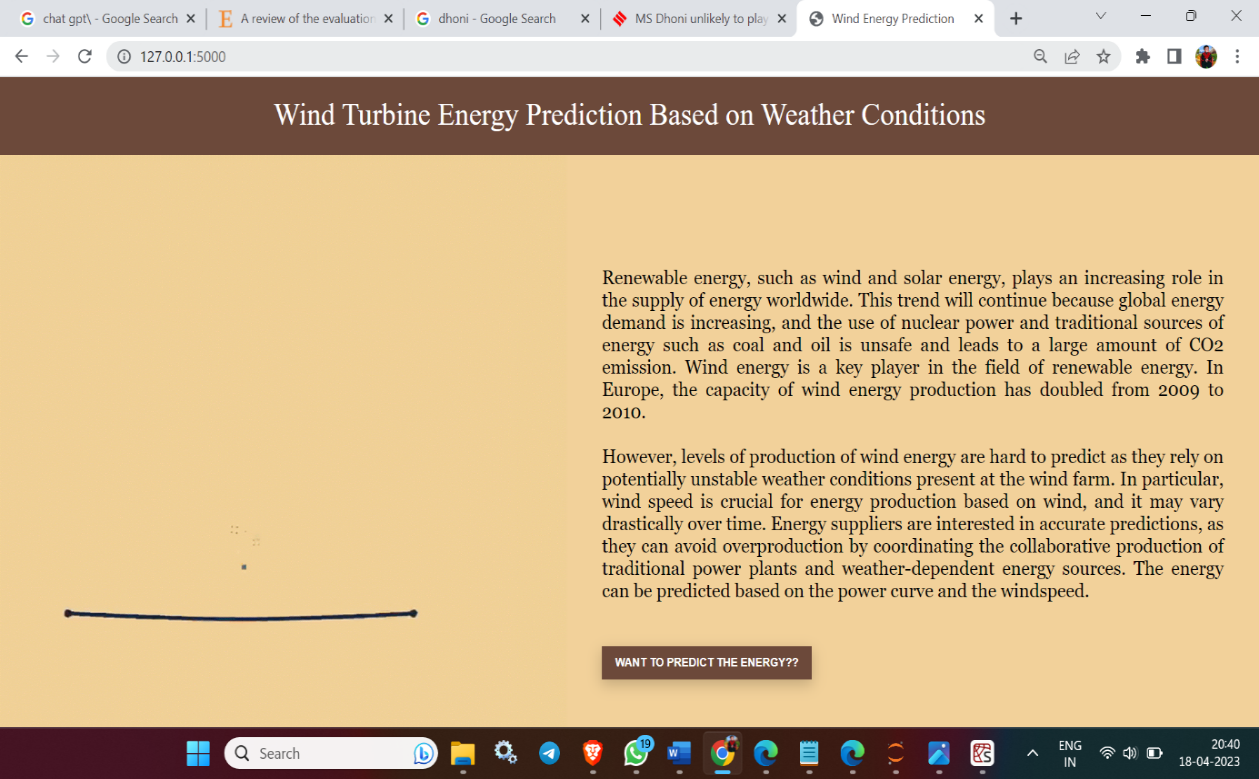
**Fig 5.3.3 intro.html code**



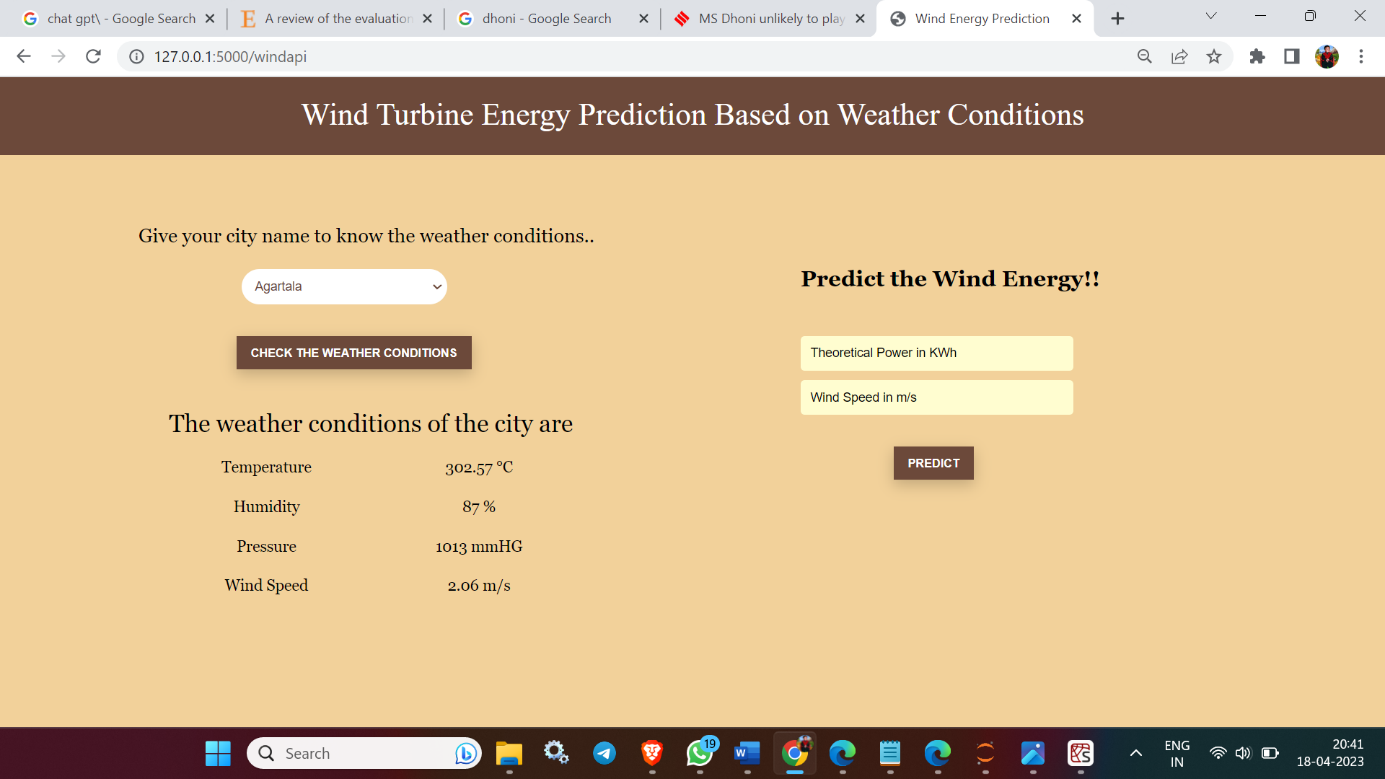
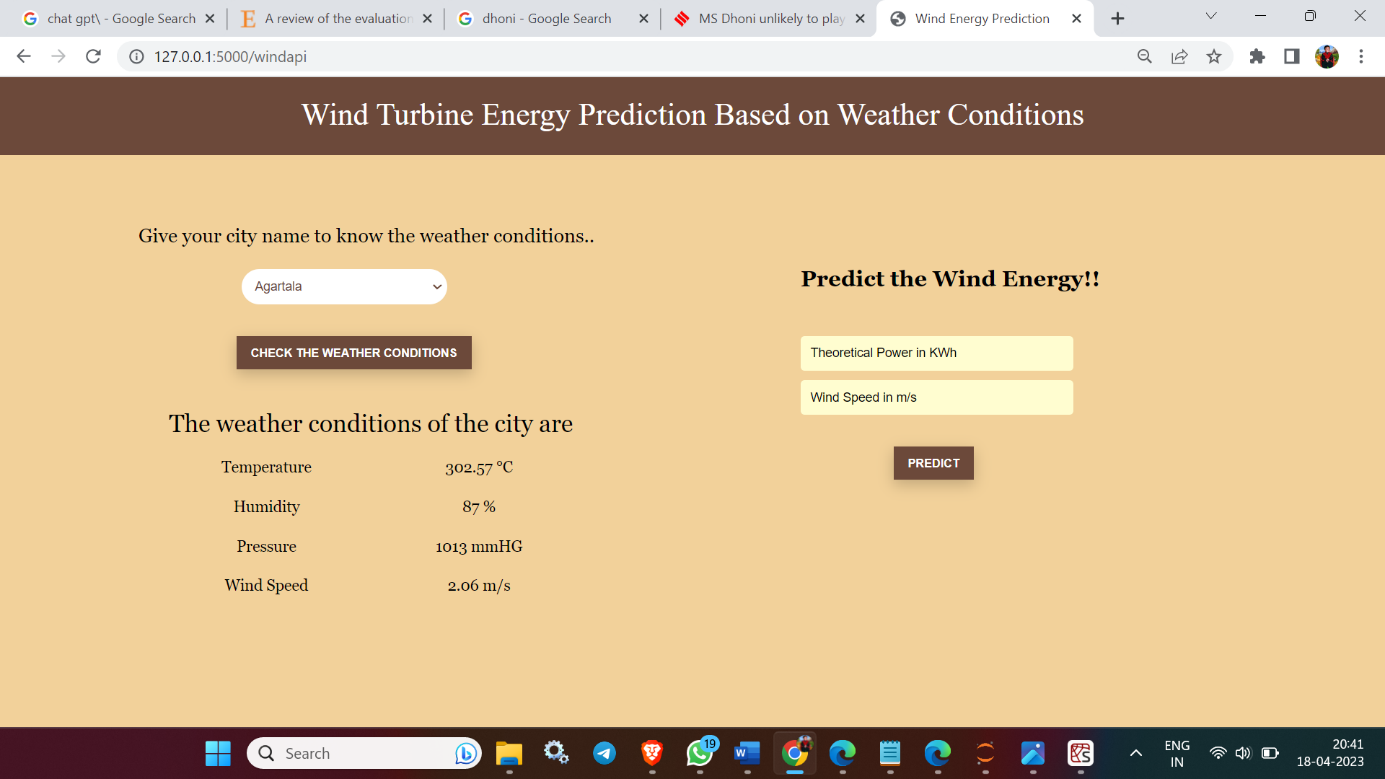
**5.3.4 Predict.html code**



**5.3.5 WindApp.py**



**5.3.6 main page**



**5.3.7 weather prediction and Wind Energy calculation page**

**B. SOURCE CODE:**

**5.4 PYTHON CODE USED IN JUPYTER NOTEBOOK**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from collections import Counter

import seaborn as sns

import os

import joblib

data = pd.read\_csv('T1.csv')

data.rename(columns = {'LV ActivePower (kW)':'ActivePower(kW)',

"Wind Speed (m/s)":"WindSpeed(m/s)",

"Wind Direction (°)":"WindDirection","Theoretical\_Power\_Curve (KWh)":"TheoreticalPowerCurve(KWh)"},

inplace = True)

data.head()

data['Date/Time'] = pd.to\_datetime(data['Date/Time'],format='%d %m %Y %H:%M')

data['year'] = data['Date/Time'].dt.year

data['month'] = data['Date/Time'].dt.month

data['day'] = data['Date/Time'].dt.day

data['Hour'] = data['Date/Time'].dt.hour

data['minute'] = data['Date/Time'].dt.minute

data.head()

def mean\_speed(x):

x = round(x,2)

a = x//1

a,b = a+0.25,a+0.75

if x < a:

x = a - 0.25

else:

x = b -0.25

return x

data['meanSpeed'] = data['WindSpeed(m/s)'].apply(mean\_speed)

data.head(100)s

def mean\_direction(x):

list=[]

i=15

while i<=375:

list.append(i)

i+=30

for i in list:

if x < i:

x=i-15

if x==360:

return 0

else:

return x

data["meanDirection"]=data["WindDirection"].apply(mean\_direction)

data.head(100)

directiondict = {0:"N", 30:"NNE", 60:"NEE", 90:"E", 120:"SEE", 150:"SSE", 180:"S", 210:"SSW", 240:"SWW", 270:"W", 300:"NWW", 330:"NNW"}

def wind\_direction(x):

for x in directiondict:

return directiondict[x]

data['windCDirection'] = data['meanDirection'].apply(wind\_direction)

data.head(10)

list\_data=[]

list\_yon=["N","NNE","NEE","E","SEE","SSE","S","SSW","SWW","W","NWW","NNW"]

for i in range(0,12):

data1T\_A=data[data["windCDirection"] == list\_yon[i]]

DepGroup\_A = data1T\_A.groupby("meanSpeed")

data\_T\_A = DepGroup\_A.mean()

data\_T\_A.drop(columns = {"WindSpeed(m/s)",

"WindDirection",

"meanDirection"},

inplace = True)

listTA\_WS = data\_T\_A.index.copy()

data\_T\_A["WindSpeed(m/s)"] = listTA\_WS

data\_T\_A = data\_T\_A[["WindSpeed(m/s)",

"ActivePower(kW)",

"TheoreticalPowerCurve(KWh)"]]

data\_T\_A["Index"] = list(range(1,len(data\_T\_A.index)+1))

data\_T\_A.set\_index("Index", inplace = True)

data\_T\_A = data\_T\_A.round({'ActivePower(kW)': 2,

'TheoreticalPowerCurve(KWh)': 2})

data\_T\_A["count"] = [len(data1T\_A["meanSpeed"][data1T\_A["meanSpeed"] == x]) for x in data\_T\_A["WindSpeed(m/s)"]]

list\_data.append(data\_T\_A)

list\_table=[data\_T\_N,data\_T\_NNE,data\_T\_NEE,data\_T\_E,data\_T\_SEE,data\_T\_SSE,data\_T\_S,

data\_T\_SSW,data\_T\_SWW,data\_T\_W,data\_T\_NWW,data\_T\_NNW]

list\_tableName=["N","NNE","NEE","E","SEE","SSE","S","SSW","SWW","W","NWW","NNW"]

def graph\_T(i):

fig = plt.figure(figsize=(20,10))

plt.plot(list\_table[i]["WindSpeed(m/s)"],

list\_table[i]["TheoreticalPowerCurve(KWh)"],

label = "Theoretical Power Curve",

marker = "o", markersize = 10, linewidth = 5)

plt.plot(list\_table[i]["WindSpeed(m/s)"],

list\_table[i]["ActivePower(kW)"],

label = "Actual Power Curve",

marker = "o", markersize = 10, linewidth = 5)

plt.xlabel("Wind Speed (m/s)")

plt.ylabel("Power (kW)")

plt.title("Direction towards {}".format(list\_tableName[i]))

plt.legend()

plt.show()

fig.savefig("{}\_Powercurve.jpeg".format(list\_tableName[i]))

plt.close(fig)

for i in range(0,12):

graph\_T(i)

X = data[[ 'WindDirection', 'month', 'day', 'Hour', 'meanSpeed']]

X

y = data['ActivePower(kW)']

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.20, random\_state = 42)

from sklearn.linear\_model import Lasso

from sklearn import metrics

from sklearn.metrics import r2\_score

lasso = Lasso(alpha = 0.01)

model = lasso.fit(X\_train, y\_train)

pred\_train\_lasso= lasso.predict(X\_train)

a=metrics.mean\_squared\_error(y\_train,pred\_train\_lasso)

print("Training RMSE and R2 score:")

print(np.sqrt(a))

print(r2\_score(y\_train, pred\_train\_lasso))

pred\_test\_lasso= lasso.predict(X\_test)

print("Testing RMSE and R2 score:")

print(np.sqrt(a))

print(r2\_score(y\_test, pred\_test\_lasso))

print(\*model.predict([[270,1,1,23,5.5]]))

joblib.dump(model,'lasso.pkl')

**5.5 PYTHON CODE USED FOR APP BUILDING:**

import numpy as np

from flask import Flask, request, jsonify, render\_template

import joblib

import requests

app = Flask(\_\_name\_\_)

model = joblib.load('power\_prediction.pkl')

@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="27482473161ff3fccd89b5265854fcea"

url="http://api.openweathermap.org/data/2.5/weather?q="+city+"&appid="+apikey

resp = requests.get(url)

print(resp)

resp=resp.json()

print(resp)

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)

@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)

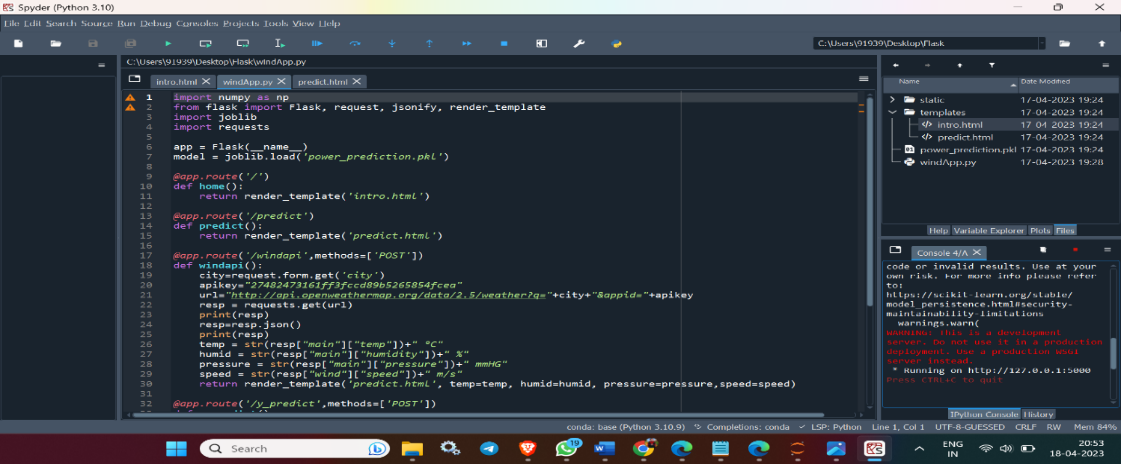
print(prediction)

output=prediction[0]

return render\_template('predict.html', prediction\_text='The energy predicted is {:.2f} KWh'.format(output))

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=False)



**Fig 5.5.1 URL Generated page**

**5.6 HTML CODES USED:**

**Intro.html**

<html>

<head>

<title>Wind Energy Prediction</title>

<style>

.header {

top:0px;

margin:0px;

left: 0px;

right: 0px;

position: fixed;

background: #6c493a;

color: white;

overflow: hidden;

padding-bottom: 30px;

font-size: 2.25vw;

width: 100%;

padding-left:0px;

text-align: center;

padding-top:20px;

}

.second{

top:80px;

bottom:0px;

margin:0px;

left: 0px;

right: 0px;

position: fixed;

padding: 0px;

width: 100%;

background-image: url(static/images/m123.gif);

background-repeat:no-repeat;

background-size: contain;

}

.inside{

top:80px;

bottom:0px;

margin:0px;

left: 45%;

right: 0%;

position: fixed;

padding-left: 40px;

padding-top:8%;

padding-right:40px;

background-color:#F2D19A;

font-family:Georgia, serif;

color:black;

font-size:20px;

text-align:justify;

}

.myButton{

border: none;

text-align: center;

cursor: pointer;

text-transform: uppercase;

outline: none;

overflow: hidden;

color: #fff;

font-weight: 700;

font-size: 12px;

background-color: #6c493a;

padding: 10px 15px;

margin: 0 auto;

box-shadow: 0 5px 15px rgba(0,0,0,0.20);

}

</style>

</head>

<body>

<div class="header">Wind Turbine Energy Prediction Based on Weather Conditions</div>

<div class="second">

<div class="inside">

<br><br><br>

<a href="{{url\_for('predict')}}"><button type="button" class="myButton" >Want to predict the energy??</button></a>

</div>

</div>

</body>

</html>

**Predict.html**

<html>

<head>

<title>Wind Energy Prediction</title>

<style>

.header {

top:0px;

margin:0px;

left: 0px;

right: 0px;

position: fixed;

background: #6c493a;

color: white;

overflow: hidden;

padding-bottom: 30px;

font-size: 2.25vw;

width: 100%;

padding-left:0px;

text-align: center;

padding-top:20px;

}

.second{

top:80px;

bottom:0px;

margin:0px;

left: 0px;

right: 0px;

position: fixed;

padding: 0px;

width: 100%;

background-color:#F2D19A;

font-family:Georgia, serif;

color:black;

font-size:20px;

}

.inside{

top:80px;

bottom:0px;

margin:0px;

left: 55%;

right: 0%;

position: fixed;

padding-left: 40px;

padding-top:8%;

padding-right:40px;

background-color:#F2D19A;

font-family:Georgia, serif;

color:black;

font-size:20px;

text-align:justify;

}

.myButton{

border: none;

text-align: center;

cursor: pointer;

text-transform: uppercase;

outline: none;

overflow: hidden;

color: #fff;

font-weight: 700;

font-size: 12px;

background-color: #6c493a;

padding: 10px 15px;

margin: 0 auto;

box-shadow: 0 5px 15px rgba(0,0,0,0.20);

margin-left:17%;

}

input {

width:50%;

margin-bottom: 10px;

background: #FFFDD0;

border: none;

outline: none;

padding: 10px;

font-size: 13px;

color: #6c493a;

text-shadow: white;

border: #6c493a;

border-radius: 4px;

box-shadow: white;

}

::placeholder {

color: black;

opacity: 1;

}

.left{

top:80px;

bottom:0px;

margin:0px;

left: 0%;

right: 50%;

position: fixed;

padding-left: 10%;

padding-top:5%;

padding-right:40px;

background-color:#F2D19A;

font-family:Georgia, serif;

color:black;

font-size:20px;

align:center;

}

select {

width:50%;

margin-bottom: 10px;

background: white;

border: none;

outline: none;

padding: 10px;

font-size: 13px;

color: #6c493a;

text-shadow: white;

border: #6c493a;

border-radius: 40px;

box-shadow: white;

}

input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2); }

</style>

</head>

<body>

<div class="header">Wind Turbine Energy Prediction Based on Weather Conditions</div>

<div class="second">

<div class="left">

Give your city name to know the weather conditions..<br><br>

<div style="margin-left:20%">

<form action="{{ url\_for('windapi')}}"method="post">

<select name="city" required>

<option value ="Agartala" > Agartala </option>

<option value ="Aizawl" > Aizawl </option>

<option value ="Amravati" > Amravati </option>

<option value ="Bengaluru" > Bengaluru </option>

<option value ="Bhopal" > Bhopal </option>

<option value ="Bhubaneswar" > Bhubaneswar </option>

<option value ="Chandigarh" > Chandigarh </option>

<option value ="Chennai" > Chennai </option>

<option value ="Daman" > Daman </option>

<option value ="Dehradun" > Dehradun </option>

<option value ="Delhi" > Delhi </option>

<option value ="Dispur" > Dispur </option>

<option value ="Gandhinagar" > Gandhinagar </option>

<option value ="Gangtok" > Gangtok </option>

<option value ="Hyderabad" > Hyderabad </option>

<option value ="Imphal" > Imphal </option>

<option value ="Itanagar" > Itanagar </option>

<option value ="Jaipur" > Jaipur </option>

<option value ="Kavaratti" > Kavaratti </option>

<option value ="Kohima" > Kohima </option>

<option value ="Kolkata" > Kolkata </option>

<option value ="Lucknow" > Lucknow </option>

<option value ="Mumbai" > Mumbai </option>

<option value ="Panaji" > Panaji </option>

<option value ="Patna" > Patna </option>

<option value ="Pondicherry" > Pondicherry </option>

<option value ="Port Blair" > Port Blair </option>

<option value ="Raipur" > Raipur </option>

<option value ="Ranchi" > Ranchi </option>

<option value ="Shillong" > Shillong </option>

<option value ="Shimla" > Shimla </option>

<option value ="Silvassa" > Silvassa </option>

<option value ="Srinagar" > Srinagar </option>

<option value ="Thiruvananthapuram" > Thiruvananthapuram </option>

</select><br><br>

<div style="margin-left:-22%"><button type="submit" class="myButton" >Check the Weather Conditions</button></div>

</form>

</div>

<div>

<table style="margin-left:2%; text-align:center; border-spacing:20px;">

<tr>

<td colspan="2" style="font-size:25px;">The weather conditions of the city are</td>

</tr>

<tr>

<td>Temperature</td><td>{{temp}}</td>

</tr>

<tr>

<td>Humidity</td><td>{{humid}}</td>

</tr>

<tr>

<td>Pressure</td><td>{{pressure}}</td>

</tr>

<tr>

<td>Wind Speed</td><td>{{speed}}</td>

</tr>

</table>

</div>

</div>

<div class="inside">

<div style="font-size:23px;font-weight:bold;">Predict the Wind Energy!!</div>

<br><br>

<form action="{{ url\_for('y\_predict')}}" method="post">

<input type="text" name="theo" placeholder="Theoretical Power in KWh" required="required" />

<input type="text" name="wind" placeholder="Wind Speed in m/s" required="required" /><br><br>

<button type="submit" class="myButton" >Predict</button>

</form>

<br>

<br>

{{ prediction\_text }}

</div>

</div>

</body>

</html>

**CHAPTER 6**

**REFERENCES:**

* See <http://anemos.cma.fr/>
* Moreno, P., L. Benito, R. Borén and M. Cabré: Short-Term Wind Forecast. Results of First Year Planning Maintenance at a Wind Farm. Poster presented on the European Wind Energy Conference and Exhibition, Madrid (ES), 16-20 June, 2003
* Still, D., B. Grainger: Demanding Seas – The UK’s First Offshore Wind Farm. Proceedings of the European Wind Energy Conference, Copenhagen, Denmark, 2-6 June 2001, pp. 169-171, ISBN 3-936338-09-4. Note: This statement was done during the talk.
* Landberg, L., G. Giebel, H.Aa. Nielsen, T.S. Nielsen, H. Madsen: Short-term Prediction – An Overview. Wind Energy 6(3), pp. 273-280, June 2003. DOI 10.1002/we.96  
  Landberg, L.: A Mathematical Look at a Physical Power Prediction Model. Wind Energy 1, pp. 23-28 (1998). DOI: 10.1002/(SICI)1099-1824(199809)1:13.0.CO;2-9
* Bossanyi, E.A.: Short-Term Wind Prediction Using Kalman Filters. Wind Engineering 9(1), pp. 1-8 (1985)
* Informationen aus dem Forschungsschwerpunkt Energieversorgung mit dezentralen Kleinkraftwerken in leistungsbegrenzten Versorgungsnetzen. Fachhochschule Wilhelmshaven, Fachbereich Elektrotechnik, Oktober 1999
* Tantareanu, C.: Wind Prediction in Short Term: A first step for a better wind turbine control. Nordvestjysk Folkecenter for Vedvarende Energi, October 1992, ISBN 87-7778-005-1
* Dutton, A.G., G. Kariniotakis, J.A. Halliday, and E. Nogaret: Load and Wind Power Forecasting Methods for the Optimal Management of Isolated Power Systems with High Wind Penetration. Wind Engineering 23(2), pp. 69-87 (1999)
* Kariniotakis, G., E. Nogaret, and G. Stavrakis: Advanced Short-Term Forecasting of Wind Power Production. Proceedings of the European Wind Energy Conference held in Dublin, Ireland, October 1997, pp. 751-754, ISBN 0 9533922 0 1
* Kariniotakis, G.N., E. Nogaret, A.G. Dutton, J.A. Halliday, and A. Androutsos: Evaluation of Advanced Wind Power and Load Forecasting Meghods for the Optimal Management of Isolated Power Systems. Proceedings of the European Wind Energy Conference, Nice, France, 1-5 March 1999, pp. 1082-1085, ISBN 1 902916 00 X