Predicting The Energy Output Of Wind Turbine Based On Weather Condition Using IBM Cloud

1 INTRODUCTION

1.1 Overview

Wind energy forecasting plays an important role in wind energy utilization, especially wind speed forecasting, which is a vital component of wind energy management. Over the last decade there has been rapid growth in wind generation of electricity, with the installed wind power capacity worldwide has increased almost fourfold from circa 24.3 GW (Giga Watts) to an expected 203.5 GW(Giga Watts) .In power systems, balance is maintained by continuously adjusting generation capacity and by controlling demand.

1.2 Purpose

Our aim is to map weather data to energy production. We want 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 wind farms. We are, in particular, interested in the correlation of different components that characterize the weather conditions such as wind speed, pressure, and temperature.

A good overview on the different methods that were recently applied in forecasting of wind power generation can be found Statistical approaches use historical data to predict the wind speed on an hourly basis or to predict energy output directly. On the other hand, short term prediction is often done based on meteorological data and learning approaches are applied. Kusiak, Zheng, and Song [8] have shown how wind speed data may be used to predict the power output of a wind farm based on times series prediction modelling. Neural networks are a very popular learning approach for wind power forecasting based on given time series. They provide an implicit model of the function that maps the given weather data to an energy output.

2 LITERATURE SURVEY

2.1 Existing problem

There exist a number of technological, environmental and political challenges linked to supplementing existing electricity generation capacities with wind energy. Here, mathematicians and statisticians could make a substantial contribution at the interface of meteorology and decision-making, in connection with the generation of forecasts tailored to the various operational decision problems involved. Indeed, while wind energy may be seen

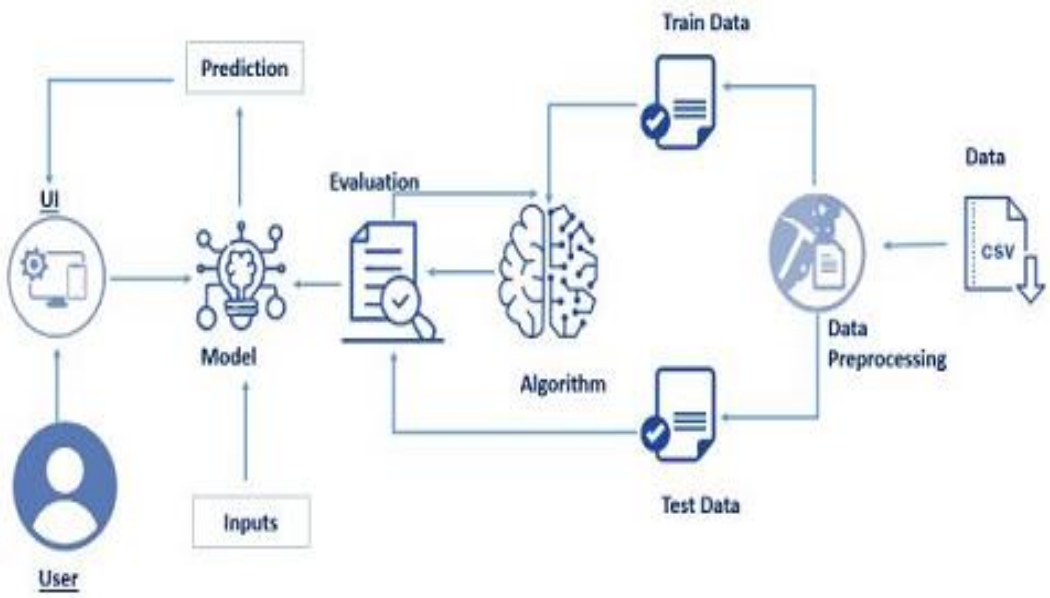
as an environmentally friendly source of energy, full benefits from its usage can only be obtained if one is able to accommodate its variability and limited predictability. Based on a short presentation of its physical basics, the importance of considering wind power generation as a stochastic process is motivated. The conventional moving-average statistical models were proven to be less efficient in forecasting the wind energy, as the wind speed is inherently variable quantity.

2.2 Proposed solution

To overcome the disadvantages of conventional models, advanced deep learning models such as LSTM (Long Short-Term Memory), can be used to map the inherently variable attribute to a complex function. Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model.

3 THEORITICAL ANALYSIS

3.1 Block diagram



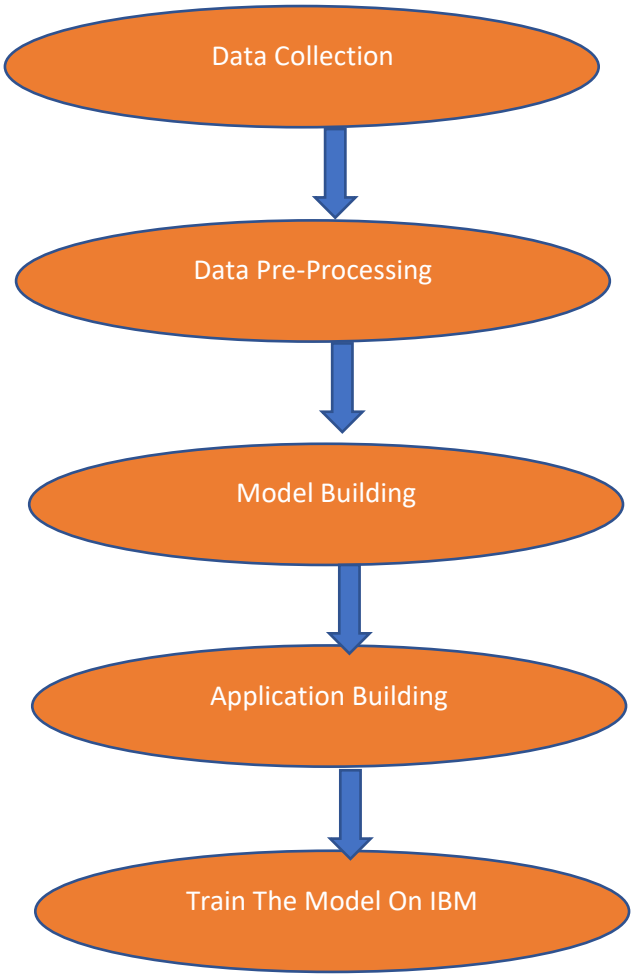
3.2 Hardware / Software designing

The product designing involves training a model and deploying it using flask. The flask app is then deployed in IBM Cloud. The flask app is designed as a RESTful API, where the User Interface curls the result from server and displays it to the user. The model is based on Random Forest Regression. We deployed a REST Api using flask and made the UI to curl the required information. By this way, the end UI application becomes light weight and uses less computational time and space. For the parameters like wind speed and wind direction, we used Open Weather API.

4 EXPERIMENTAL INVESTIGATIONS

The main goal of this project is to check feasibility of wind energy prediction by using a industrial-strength off-the-shelf non-linear modelling and feature selection tool. In our study, we investigate and predict the energy production of the wind farm Wool north in Tasmania, Australia based on publicly available data. The energy production data is made publicly available by the Australian Energy Market Operator (AEMO) in real time to assist in maintaining the security of the power system. For the creation of our models and the prediction, we associate the wind farm with the Australian weather station ID091245, located at Cape Grim, Tasmania. Its data is available for free for a running observation time window of 72 hours

5 FLOWCHART



6 RESULT

The input data set is extracted from open weather map API using API calls. For every call, hourly forecast of weather is retrieved. The data received contains 48 entries,i.e., for every 1.25 min the weather conditions are recorded and retrieved. The model is trained and validated using this dataset.

Mean Absolute Error: 160.55

r^2 score: 0.9065

7 ADVANTAGES & DISADVANTAGES

ADVANTAGES :

- Intuitive User Interface
- Faster Predictions on just a click of a button
- Accurate predictions up to 1 hour to 72 hours

- 48 predictions per hour for more accurate Power Output visualization
- The Power grids can make use of this app to efficiently monitor the wind turbine power output.
- An additional feature: Weather Now, to display the current weather at the location is added to the application.
- The model is also trained to predict direction of the wind and can be used for the alignment of the turbines.

DISADVANTAGES:

- On device Deep Learning model cannot be used. The predictions must be made through REST API call to the model.
- The time taken for predictions is thus comparatively little less.
- There is scope for generalizing the model to suit for various other regions.

8 APPLICATIONS

- The same model mentioned above can also be applied to other data such as Solar energy, tidal energy etc.
- The model can be transfer-learned for making predictions for other locations, instead of training from scratch. This reduces the time taken for training.
- The application can also be used as a standalone REST API calls

9 CONCLUSIONS

In this study we showed that wind energy output can be predicted from publicly available weather data with accuracy at best 80% R2on the training range and at best 85,5% on the unseen test data. We identified the smallest space of input variables (windGust2 and dewpoint), where reported accuracy can be achieved, and provided clear trade-offs of prediction accuracy for decreasing the input space to the windGust2 variable. We demonstrated that an off-the-shelf data modelling and variable selection tool can be used with mostly default settings to run the symbolic regression experiments as well as variable importance, variable contribution analysis, ensemble selection and validation.

10 FUTURE SCOPE

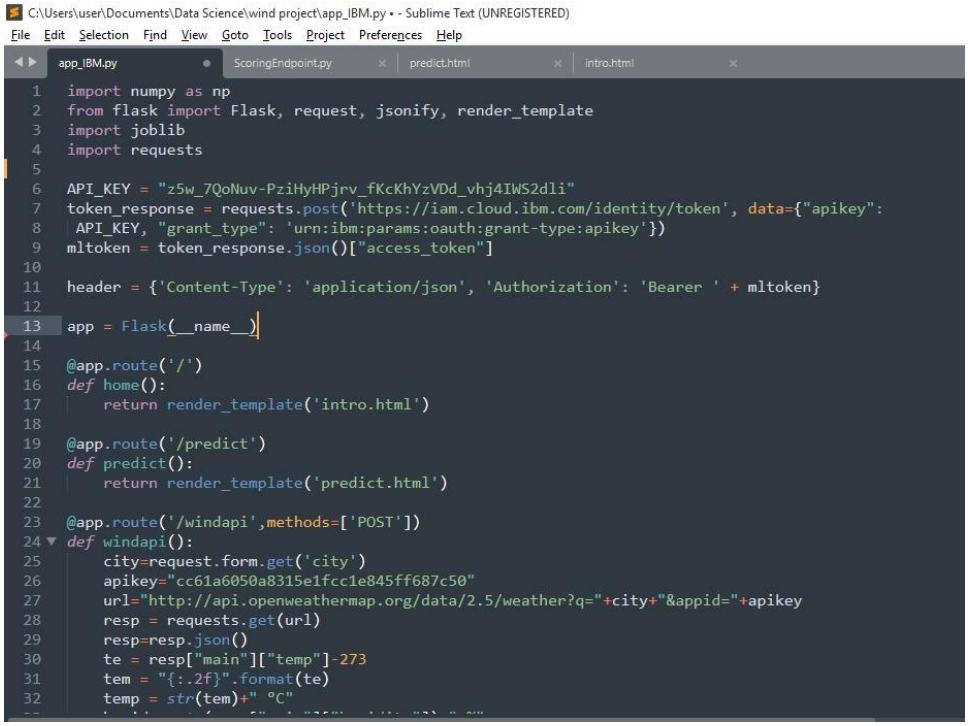
- A more generalized model can be developed to suit forecasting for various locations.
- The model can be developed to make predictions on other sources of data such as solar power, tidal power etc.
- On-device model can be developed to make much more faster predictions.

11 BIBILOGRAPHY

- https://home.openweathermap.org/
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
- https://cloud.ibm.com/login?redirect=%2Faccount
- https://joblib.readthedocs.io/en/latest/
- https://smartinternz.com/

APPENDIX.

A. Source Code for the solution built.



C:\Users\user\Documents\Data Science\wind project\ScoringEndpoint.py - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

app_IBM.py | ScoringEndpoint.py | predict.html | intro.html

```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "z5w_7QoNuv-PziHyHPjrv_fKcKhYzVDd_vhj4IWS2dli"
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}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"fields": [["f0","f1","f2"]], "values":[[123,5,320]] }]}

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/689614b2-8ded-4e79-aaac-
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
# print(response_scoring.json())
pred= response_scoring.json()
output=pred['predictions'][0]['values'][0][0]
print(output)
```

C:\Users\user\Documents\Data Science\wind project\templates\predict.html - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

app_IBM.py | ScoringEndpoint.py | predict.html | intro.html

```
            <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>
          <tr>
            <td>Wind Direction</td><td>{{deg}}</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="/y_predict" method="post" id="myForm">
          <input type="text" name="wind" placeholder="Wind Speed in m/s" value = {{speed}}  />
          <input type="text" name="theo" placeholder="Theoretical Power in KWh" required="required" />
          <input type="text" name="dir" placeholder="Wind Direction in degrees" value = {{deg}}  /><br><br>
          <button type="submit" class="myButton" >Predict</button>

        </form>

         <br>
         <br>
```

kite: Initializing, INSERT MODE, Line 146, Column 13

Run | Code

```
[0.77955763]])

In [25]: from sklearn.model_selection import train_test_split

In [26]: X_trainscaled, X_testscaled, y_trainscaled, y_testscaled = train_test_split(xscaled, yscaled, test_size=0.20, random_state=42)
         X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_state=42)

In [27]: from sklearn.ensemble import RandomForestRegressor

In [28]: forest_model = RandomForestRegressor(max_leaf_nodes =500, random_state=1)

In [29]: forest_model.fit(X_train, y_train)

         C:\Users\user\AppData\Local\Temp\ipykernel_6996\2068033108.py:1: DataConversionWarning: A column-vector y was passed when a 1d
         array was expected. Please change the shape of y to (n_samples,), for example using ravel().
           forest_model.fit(X_train, y_train)

Out[29]: RandomForestRegressor(max_leaf_nodes=500, random_state=1)

In [30]: power_preds = forest_model.predict(X_test)
```