

Predicting the Energy Output of Wind Turbine Based on Weather Condition

**IBM NALAIYA THIRAN PROJECT BASED LEARNING
ON
PROFESSIONAL READLINESS FOR INNOVATION,
EMPLOYABILITY AND ENTREPRENEURSHIP**

Submitted by

- **ISHWARYA(Team Lead)-921319104066**
- **JEEVADHARSHA G S-921319104073**
- **JOAN ALOSIA MARY S-921319104082**
- **MADHUVANTHI M-921319104107**

Team ID	PNT2022TMID04956
Project Name	PREDICTING THE ENERGY OUTPUT OF WIND TURBINE BASED ON WEATHER CONDITION

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING**

PSNA COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution,Affiliated to Anna University, Chennai)

DINDIGUL-624622

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF FIGURES	ii
	LIST OF TABLES	iii
	LIST OF ABBREVIATIONS	iv
1	INTRODUCTION	1
	1.1 Overview	1
	1.2 Deep learning	1
	1.3 Bi-LSTM	3
2	LITERATURE REVIEW	4
	2.1 Related Works	4
	2.2 Comparative Study	7
3	SOFTWARE DESCRIPTION	9
	3.1 Software Requirements	9
4	PROJECT DESCRIPTION	10
	4.1 System Architecture	11
	4.1.1 Data Collection	11
	4.1.2 Data Pre-Processing	14
	4.1.3 Training with deep learning models	15
	4.1.4 Graphical User Interface	15
	4.2 Module Description	16
	4.2.1 Prediction Module	16
	4.2.2 Forecasting Module	16
5	SYSTEM IMPLEMENTATION	17
6	ADVANTAGES OF THE PROPOSED SYSTEM	19
7	RESULTS	20
8	CONCLUSION AND FUTURE SCOPE	22
9	REFERENCES	24
	APPENDIX	26

ABSTRACT

Extracting electricity from renewable resources has been widely investigated in the past decades to decrease the worldwide crisis in the electrical energy and environmental pollution. For a wind farm which converts the wind power to electrical energy, a big challenge is to accurately predict the wind power in spite of the fluctuations. The energy output of a wind farm is highly dependent on the weather conditions present at its site. For the wind-farm operator, this poses difficulty in the system scheduling and energy dispatching, as the schedule of the wind-power availability is not known in advance. A precise forecast needs to overcome problems of variable energy production caused by fluctuating weather conditions. 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. The objective of our project is to develop an end-to-end web application to predict & forecast the energy output of the wind turbine based on weather conditions. In this project, a prediction system is developed using a special kind of RNN (Recurrent Neural Network), Bidirectional Long Short Term Memory (Bi-LSTM) deep learning model which has a prominent performance in capturing the long-term dependencies along the time steps, and thus very applicable for wind power prediction.

LIST OF FIGURES

FIGURE NO	NAME OF THE FIGURE	PAGE NO
1.1	Recurrent Neural Network	2
1.2	Bi-LSTM	3
3.1	Technology Stack	9
4.1	System Architecture	11
4.2	Power versus Wind Speed	12
4.3	Power versus Air Temperature	12
4.4	Power versus Air Pressure	13
4.5	Power versus Wind Direction	14
4.6	Accuracy observed with different models	15
4.7	Prediction Module Architecture	16
4.8	Forecasting Module Architecture	17
5.1	Block Diagram	18
6.1	Preferences Tab	19
7.1	User Interface	20
7.2	Forecasting Results	21

LIST OF TABLES

TABLE NO	NAME OF THE TABLE	PAGE NUMBER
2.1	Comparative Study	8

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
ANN	Artificial Neural Network
Bi-LSTM	Bidirectional Long Short Term Memory
CNN	Convolutional Neural Network
DBN	Deep Belief Network
GA-LSTM	Genetic Algorithm-Optimized Long Short-Term Memory
GRU	Gated Recurrent Units
LSTM	Long Short Term Memory
NARX	Nonlinear Autoregressive Exogenous Model
NWP	Numerical Weather Prediction
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Network
SVM	Support Vector Machine

CHAPTER 1

INTRODUCTION

1.1 Project Overview

In today's power grid, the penetration of **wind energy** has been booming. The growing integration of wind turbines into the power grid can only be balanced with **precise forecasts of upcoming energy productions**. Knowing the wind power beforehand helps us in many ways by minimizing the losses. The technique incorporated in our project is **deep learning**. Bi-LSTM (Bidirectional Long Short Term Memory), a special kind of Recurrent Neural Network (RNN) is a deep learning model which makes use of multiple hidden layers to train the model and provide accurate results. The trained models are rendered and deployed in a web server with a simple user interface.

1.2 Purpose

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.

CHAPTER 2

LITERATURE REVIEW

2.1 Existing Problem

Renewable energy sources, especially wind energy, are to play a larger role in providing electricity to industrial and domestic consumers. This is already the case today for a number of European countries, closely followed by the US and high growth countries, for example, Brazil, India and China. 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. After describing representative operational decision-making problems for both market participants and system operators, it is underlined that forecasts should be issued in a probabilistic framework. Even though, eventually, the forecaster may only communicate single-valued predictions. The existing approaches to wind power forecasting are subsequently described, with focus on single-valued predictions, predictive marginal densities and space-time trajectories. Upcoming challenges related to generating improved and new types of forecasts, as well as their verification and value to forecast users, are finally discussed.

2.2 References

- [1] Yubo Tan, Hongkunchen Chuang Qiu, “Wind Power Prediction and Pattern Feature Based on Deep Learning Method”, <http://pipad.org/tmp/WindPower.pdf> , 2014.
- [2] Justin Philipp Heinermann, “Wind Power Prediction with MachineLearning Ensembles”,<https://d-nb.info/1122481861/34>,2016.
- [3]T Brahimi, “Using Artificial Intelligence to Predict Wind Speed for Energy Application in Saudi Arabia”, <https://www.mdpi.com/1996-1073/12/24/4669/pdf>,2019.
- [4] Qu Xiaoyun, Kang Xiaoning, Zhang Chao, Jiang Shuai, Ma Xiuda, “Short-Term Prediction of Wind Power Based on Deep Long Short-TermMemory”, <https://sci-hub.do/10.1109/APPEEC.2016.7779672>, 2016.
- [5] Sowmya SevoorVaitheeswaran , Varun Raj Ventrapragada , “Wind Power Pattern Predictionintimeseriesmeasurementdataforwindenergypredictionmodellingsusing LSTM-GA networks”, <https://sci-hub.do/https://doi.org/10.1109/ICCCNT45670.2019.8944827>, 2019
- [6] YiweiF, Wei HU, “Multi-step Ahead Wind Power Forecasting Based on Recurrent Neural Networks”, <https://scihub.do/https://doi.org/10.1109/APPEEC.2018.8566471>,2018.
- [7] Zhichao Shi, Hao Liang, Venkata Dinavah, “Direct Interval Forecast of Uncertain Wind Power Based on Recurrent NeuralNetworks”, <https://sci-hub.do/https://doi.org/10.1109/TSTE.2017.2774195>, 2018.
- [8] Jaseela V Rasheed, “A Survey on Hybrid model to Forecast Wind Power using LSTM”,http://www.ijirset.com/upload/2017/july/138_A_Survey.pdf,2017.
- [9] ZheRen,ChengshuaiHuang,MengLi,“ResearchonWindPowerPrediction”, ,2019. <https://sci-hub.ee/10.1109/EI247390.2019.9061851>
- [10] AnwenZhu,XiaohuiLi,ZhiyongMo,HuarenWu,“WindPowerPredictionBased on a Convolutional Neural Network”,<https://sci-hub.ee/10.1109/ICCDs.2017.8120465> ,2017.
- [11] AmilaT.Peiris,JeevaniJayasinghe,UpakaRathnayake,“ForecastingWindPower Generation Using Artificial Neural Network: “Pawan Danawi”—A Case Study from Sri Lanka”, <https://doi.org/10.1155/2021/5577547> ,2021.
- [12] SumantaPasari, Aditya Shah, Utkarsh Sirpurkar, “Wind Energy PredictionUsing Artificial Neural Networks”, https://link.springer.com/chapter/10.1007/978-3-030-44248-4_10 ,2020.
- [13] Z. Liu, W. Gao,Y-H. Wan, E. Muljadi, “Wind Power Plant Prediction by Using Neural Networks”,<https://www.nrel.gov/docs/fy12osti/55871.pdf> 2012.

2.3 Problem Statement Definition

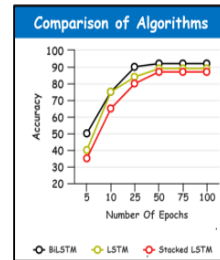
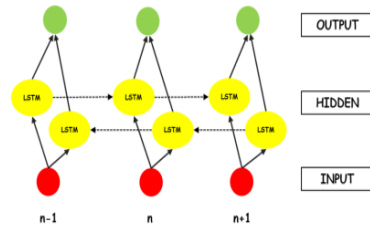
Wind power is the process of harnessing energy from the movement of the wind and converting it to useful forms of mechanical power and electricity. Today, most wind energy comes from turbines – essentially giant windmills. The wind turns two or three of the turbine's propeller-like blades around the turbine's rotor. The rotor is connected to a main shaft, which spins a generator to create electricity. Here the production of the energy is highly dependent on the speed of the wind. The speed of the wind can be predicted from the weather forecasting and this enables us to calculate the production of energy from the turbine. In order to predict the energy production of the wind farm it is necessary to undertake the following tasks:

- Predict the variation in the long-term wind speed over the site at the hub height of the machines, based on the long-term speeds at the mast locations;
- Predict the wake losses that arise as a result of one turbine operating behind another- in other words in its wake and Calculate or estimate the other.

2.4 Ideation & Proposed Solution

Wind power is calculated based on weather conditions (wind speed, wind direction, pressure, temperature, dew point, relative humidity) Our aim is to develop an end to end web application to predict the energy output of the wind turbine based on weather conditions. The technique incorporated in our project is deep learning. Rather than installing more devices on the turbine, this idea can maximize yields and efficiency while having small effects on the climate. At the same time, it boosts the performance and competitiveness of market players, making this business more attractive.

- 🧩 Time series problems are mostly solved using RNN. These models have memory, i.e., the model can remember the information throughout the time.



- A special kind of RNN – BiLSTM Network (Bidirectional Long Short Term Memory) is implemented which has a prominent performance in capturing the long-term dependencies along the time steps, and thus very applicable for wind power prediction.
- The proposed algorithm gave us more accurate results when compared with other models.

2.5 Empathy Map Canvas



2.6 Ideation & Brainstorming



2.7 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The manufacturer needs to find a way to analyze the weather conditions of a region so they can choose regions that produce high quality and quantities of wind energy. Overproduction and cost of production needs to be reduced. Wind energy should be utilized in a way to provide a steady supply of electricity.
2.	Idea / Solution Description	We examine the impact of different weather conditions on the energy output of wind farms. By accurately forecasting the wind-power, we reduce the need for additional balancing energy and reserve power to integrate wind power. A prediction system is developed with a method of combining statistical models and physical models. In this model, the inlet condition of the wind farm is forecasted by the auto regressive model.
3.	Novelty / Uniqueness	Currently, wind energy is not a primary source of electricity. Implementing our solution makes it possible to maximize energy output. This solution would make renewable energy sources more widely used. The user can upload their own data in real-time for forecasting.
4.	Social Impact / Customer Satisfaction	Local employment, better health, consumer choice, improvement of life standard, social bonds creation, income development, demographic impacts, and community development can be achieved by the proper usage of renewable energy system. Renewable energy improves human well-being and overall welfare well beyond GDP. Switching to clean sources of energy, thus helps address not only climate change but also air pollution and health.

2.8 Proposed Solution Fit

CUSTOMER SEGMENT:

A parent who is afraid of their child's safety whenever they are not with the child.

PROBLEMS/PAINS:

Sometimes the children can prone to dangers. This device can protect them from those situations.

TRIGGERS TO ACT:

After noticing that child entering the danger zone the device triggers the alarm.

CUSTOMER LIMITATIONS:

- Should be affordable.
- Device should be weight less
- Easy to operate.

PROBLEM ROOT/CAUSE:

- Balancing office works and children safety is tough.
- Sometimes we miss watching out children

AVAILABLE SOLUTIONS:

Ensuring the children's protection by using the IOT technology and smart sensors.

BEHAVIOUR:

Child's moment is always notified to parents through their smart phones.

EMOTIONS:

- Worry
- Anxiety
- Frustration
- Angry

OUR SOLUTION:

A gadget needs to be invented which monitors the child and activates buzzer when need.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Functional Requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User requirements	Collecting informations like date of travel, departing & arrival destination, flight number or booking number, etc for providing the status of the flight.
FR-4	User friendliness	This system is easy to learn and understand.

3.2 Non-Functional Requirement

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	How easy is it for a customer to use the system?
NFR-2	Security	Security's part will be protected against malware attacks or unauthorized access. But there's a catch. The lion's share of security non-functional requirements can be translated into concrete functional counterparts. If you want to protect the admin panel from unauthorized access, you would define the login flow and different user roles as system behavior or user actions.

NFR-3	Reliability	<p>Reliability specifies how likely the system or its element would run without a failure for a given period of time under predefined conditions. Traditionally, this probability is expressed in percentages. For instance, if the system has 85 percent reliability for a month, this means that during this month, under normal usage conditions, there's an 85 percent chance that the system won't experience critical failure.</p>
NFR-4	Performance	<p>Performance defines how fast a software system or a particular piece of it responds to certain users' actions under a certain workload. In most cases, this metric explains how long a user must wait before the target operation happens (the page renders, a transaction is processed, etc.) given the overall number of users at the moment. But it's not always like that. Performance requirements may describe background processes invisible to users, e.g. backup. But let's focus on user-centric performance.</p>

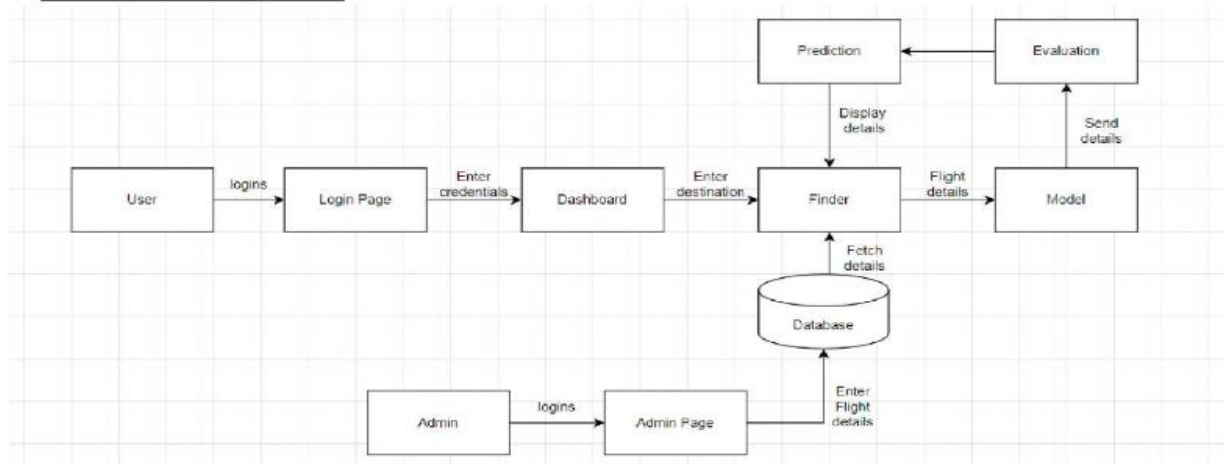
NFR-5	Availability	<p>Availability describes how likely the system is accessible to a user at a given point in time. While it can be expressed as an expected percentage of successful requests, you may also define it as a percentage of time the system is accessible for operation during some time period. For instance, the system may be available 98 percent of the time during a month. Availability is perhaps the most business-critical requirement, but to define it, you also must have estimations for reliability and maintainability.</p>
-------	---------------------	---

CHAPTER 5

PROJECT DESIGN

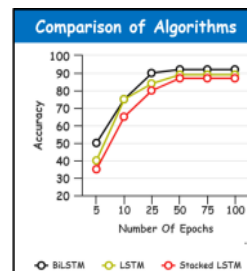
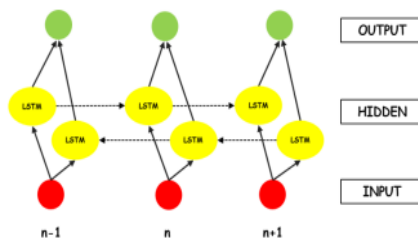
5.1 DATA FLOW DIAGRAM

Data Flow Diagrams:



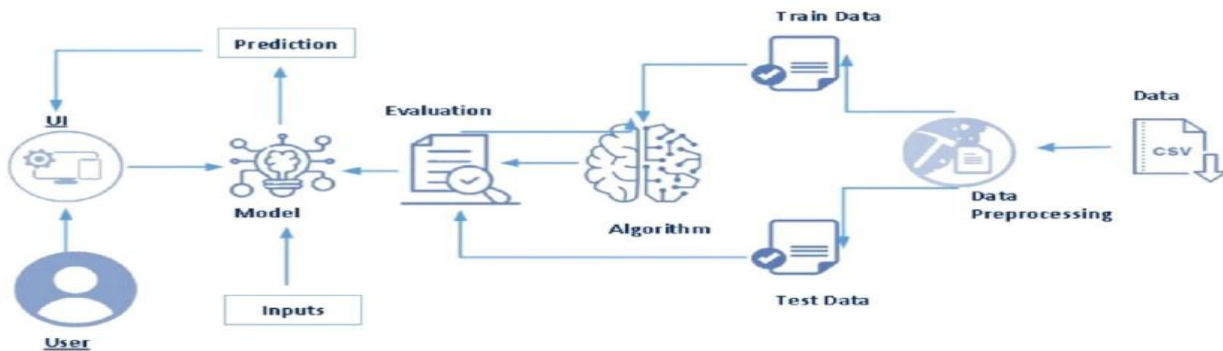
5.2 SOLUTION AND TECHNICAL ARCHITECTURE

- Wind power is calculated based on : weather conditions (wind speed, wind direction, pressure, temperature, dewpoint, relative humidity)
- Our aim is to develop an end to end web application to predict the energy output of the wind turbine based on weather conditions.
- The technique incorporated in our project is deep learning
- Time series problems are mostly solved using RNN. These models have memory, i.e., the model can remember the information throughout the time.



- A special kind of RNN – BiLSTM Network (Bidirectional Long Short Term Memory) is implemented which has a prominent performance in capturing the long-term dependencies along the time steps, and thus very applicable for wind power prediction.
- The proposed algorithm gave us more accurate results when compared with other models.

TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Type	Functional Requirement(Epic)	User Story Number	User story/ Task	Acceptance Criteria	Priority	Release
Customer(Web user)	Registration	USN - 1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation	I can receive confirmation email & click	High	Sprint-1

			email once I have registered for the application	confirm		
		USN-3	As a user, I can register for the Application through Facebook	I can register & access the Dashboard with Facebook Login	Low	Sprint-2

		USN-4	As a user, I	I can register and access by entering the OTP	Medium	Sprint-1
			can register			
			for the			
			application			
			through			
			mobile			
	Login	USN-5	As a user, I	I can access the dashboard	High	Sprint-1
			can log into the app by entering			
			email and			
	Dashboard	USN-6	password	I can various pages	High	Sprint-2
			As a user, I can navigate through			
			different			
			pages using			
			the			
			dashboard			

	Search	USN-7	As a user, I	I can receive information on various flights.	High	Sprint-2
			can search			
			for flights			
			with			
			destination			
	View	USN-8	location.	I get the information such as flight no, departure and arrival	High	Sprint-3
			As a user, I can view the details of the flights.			

CHAPTER 6

PROJECT PLANNING PHRASE

6.1 Mile Stone and Activity List

Sprint	Milestone
Sprint 1	<ol style="list-style-type: none">1. User Registers into the application through entering Email Id Password and Re enter Password for confirmation.2. User Receives a confirmation mail for their registered Email.3. User can also register to the application through Mobile number.4. User logs in into the website using Email Id password or through Gmail
Sprint 2	<ol style="list-style-type: none">1. User can access the dashboard2. User can enter the required details on weather conditions and get the desired turbine power output based on model's prediction
Sprint 3	<ol style="list-style-type: none">1. Application should store the predictions, and these predictions can be used for future analysis.2. The data stored should be secure.
Sprint 4	<ol style="list-style-type: none">1. Administrator should properly maintain the website and update it when required.

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	03 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	10 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	17 Nov 2022

6.2 Sprint Delivery plan

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	High	Ishwarya M
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	5	High	Jeevadarsha G S
Sprint-1		USN-3	As a user, I can register for the application through Phone number	2	Low	Joan Alosia Mary S
Sprint-1		USN-4	As a user, I can register for the application through Gmail	3	Medium	Madhuvanthi M

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Login(User)	USN-5	As a user, I can log into the application by entering email & password.	5	High	Ishwarya M
Sprint-2	Dashboard	USN-6	Once I have logged in, I can see my dashboard.	6	Medium	Madhuvanthi M
Sprint-2	Web access	USN-7	As a customer I can access the website to predict the turbine power.	7	High	Jeevadarsha G S
Sprint-2	Prediction	USN=8	As a customer when I enter the weather details the website should predict the approximate turbine power.	7	High	Jeevadarsha G S
Sprint-3	Analysis	USN-9	As a customer, I wish to store my predictions and make analysis.	10	Medium	Joan Alosia Mary S
Sprint-3	Security	USN-10	As a customer I expect my data to be secured.	10	Medium	Ishwarya M

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-4	Database Access	USN-11	A Administrator, I should maintain the website. And update the website regularly.	20	Low	Joan Alosia Mary S

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	03 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	10 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	17 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.





CHAPTER 4

PROJECT DESCRIPTION

4.1 System Architecture

The models operating on the production server would work with the real-life data and provide predictions to the users. The below mentioned framework represents the most basic way data scientists handle deep learning. Fig.4.1 represents our System Architecture.

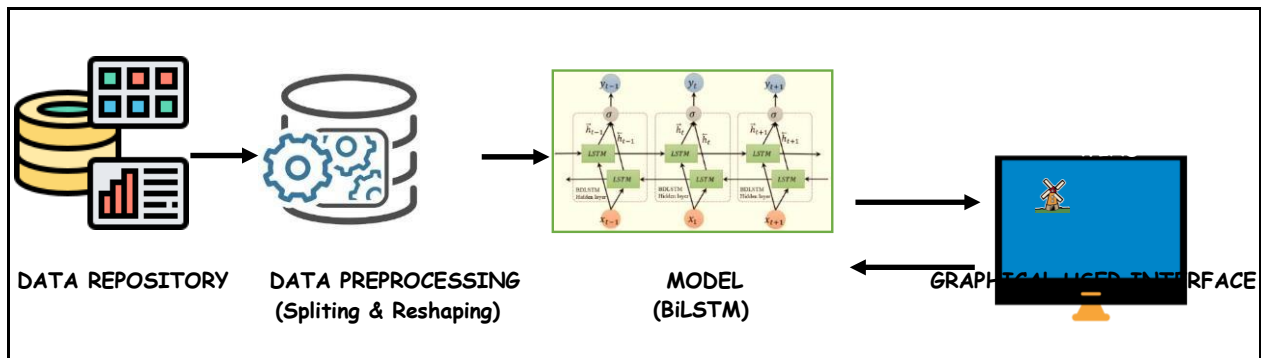


Figure 4.1: System Architecture

4.1.1. Data Collection:

Collecting the required data is the beginning of the whole process. Data repository has the repository of all the data related to weather conditions and power generated. The data which is used to train the model is obtained from National Renewable Energy Laboratory (NREL) to do this analysis. The dataset contains the details about timestamp, air temperature ($^{\circ}\text{C}$), pressure (atm), wind direction (deg), wind speed (m/s) and Power generated by the system (kW). We have hourly data for about 6 years (i.e) almost 52000 entries.

Input Parameters:

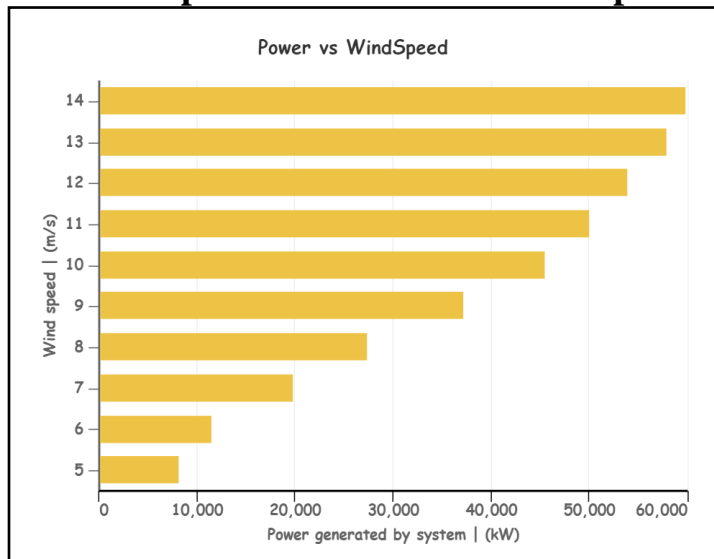
- **Wind Speed (m/s):**

Higher wind speeds generate more power because stronger winds allow the blades to rotate faster.

Faster rotation translates to more mechanical power and more electrical power from the generator.

Fig.4.2 shows the variation of power output with respect to wind speed.

Figure 4.2: Graph of Power versus Wind Speed



- **Air Temperature($^{\circ}\text{C}$):**

Wind speeds increase with a greater temperature difference. If the temperature is too high, the air density will be low, which will lessen the energy output. If the temperature is too low, the blades and other parts might be frozen, and the wind turbine will stop working.

Fig.4.3 shows the variation of power output with respect to air temperature.

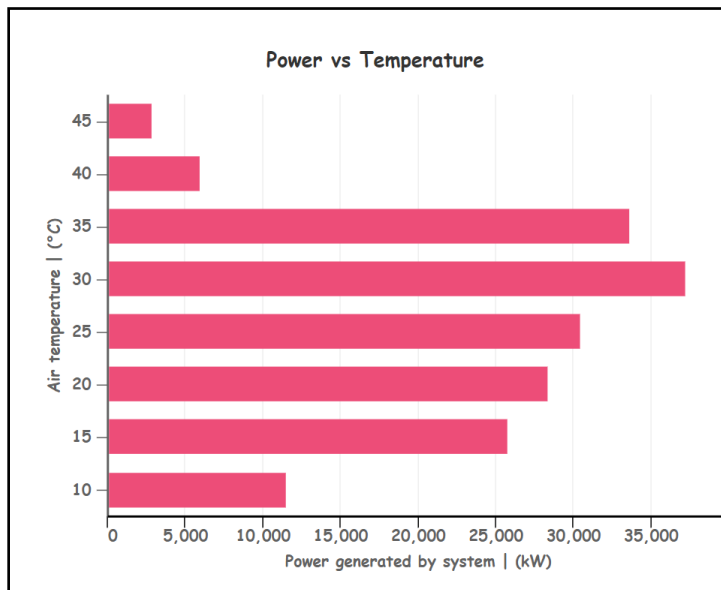


Figure 4.3: Graph of Power versus Air Temperature

- **Air Pressure(atm):**

When air slows down, its pressure increases. This means that higher wind speeds will show lower air pressure readings. Fig.4.4 shows the variation of power output with respect to air pressure. We can conclude that wind power increases with decrease in air pressure.

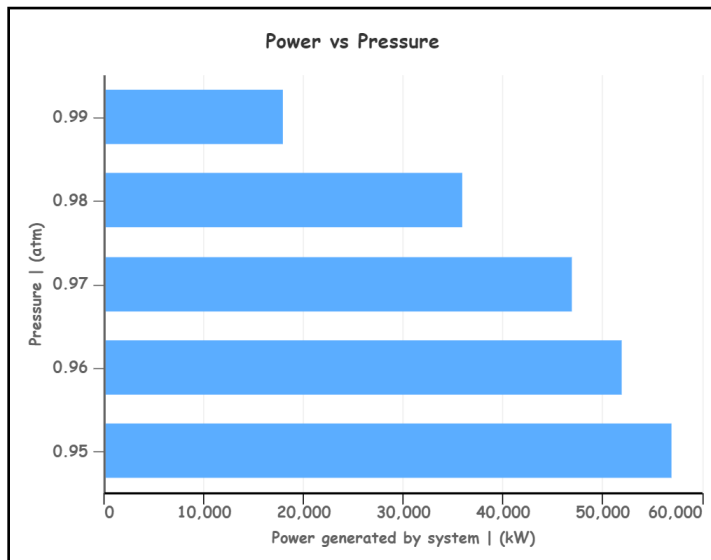
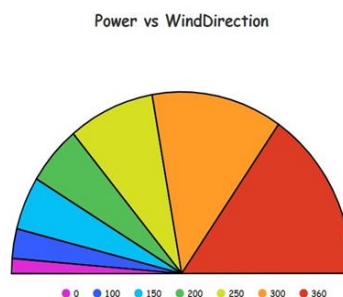


Figure 4.4: Graph of Power versus Air Pressure

- **Wind Direction(deg):**

Wind flow direction affects the turbines, reducing the wind speed and increasing turbulence for the wind turbines. A weather vane is a instrument which shows the direction the wind is blowing. Getting more nearer to 360° (north), wind speed increases, so severe winds blow from north generating more power.

Fig.4.5 shows the variation of power output with respect to wind direction.



From all the above graphs we can able to learn the pattern in the data very well.

Wind

speed available at the wind farm is a crucial parameter. Other parameters that influence the energy output are for example temperature, pressure, wind direction. Our goal is to make use of the correlation of these parameters with respect to the energy output.

$$\text{Power (kW)} = \frac{(\text{wind speed})^3 * \text{wind direction} * 10}{\text{pressure} * \text{temperature (in } ^\circ\text{F)}}$$

4.1.2. Data Pre-processing:

The model is completely dependent on the data, so it needs to be consistent and precise. Any missing value is handled, redundant values and columns are dropped out in this step. EDA (Exploratory Data Analysis) helps in visualizing the data. This process helps to decide which attributes are relevant for the model. With the help of pre-processing we get the data that is suitable to train the data.

4.1.3. Training with Deep Learning Models:

Train the deep learning model with different sets of processed data. Finally select the model that gives the maximum accuracy of all. Fig.4.6 represents the accuracy observed with different models.

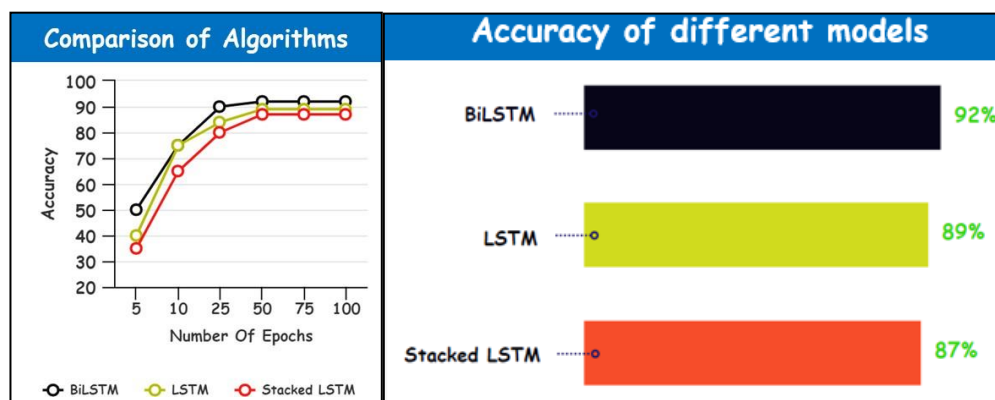


Figure 4.6: Accuracy observed with different models

4.1.4. Graphical User Interface (GUI):

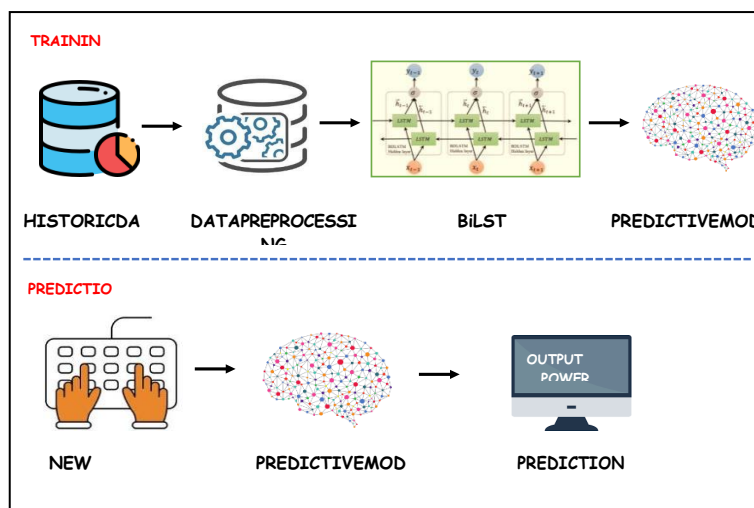
A simple, easy and understandable user interface is designed so that any layman can use it to know the future wind power status. For Predicting the power output for user defined values, user just needs to input the weather parameters for which the power is to be predicted. For Forecasting the power output of wind turbine from several hours upto 24 hours ahead, user can upload their own real time dataset (csv or xlsx format) for forecasting. A simple visualization tool is also presented in our user interface, where the data can be visualized using the graphs.

4.2. Module Description

4.2.1. Prediction Module


- 🌐 Predicting the power output for user defined values.
- 🌐 User just needs to input the weather parameters for which the power is to be predicted. Hence, the GUI is user friendly, easy, simple in nature.
- 🌐 Fig.4.7 represents the architecture of prediction module. First, we train our model with the historic data. Later, we can feed a new weather data to the model and predict the power output.


Figure 4.7: Prediction Module Architecture




4.2.2. ForecastingModule

 Forecasting the power output of wind turbine from several hours upto 24 hours ahead.

 Our Deep Wind is individually different from other wind power forecasting websites where the user can upload their own real time dataset (csv or xlsx format) for forecasting.

 Fig.4.8 represents the architecture of forecasting module. The dataset uploaded by the user is divided into train and test sets and given to our model for forecasting the power output.

 Accurate Results with minimum load time.

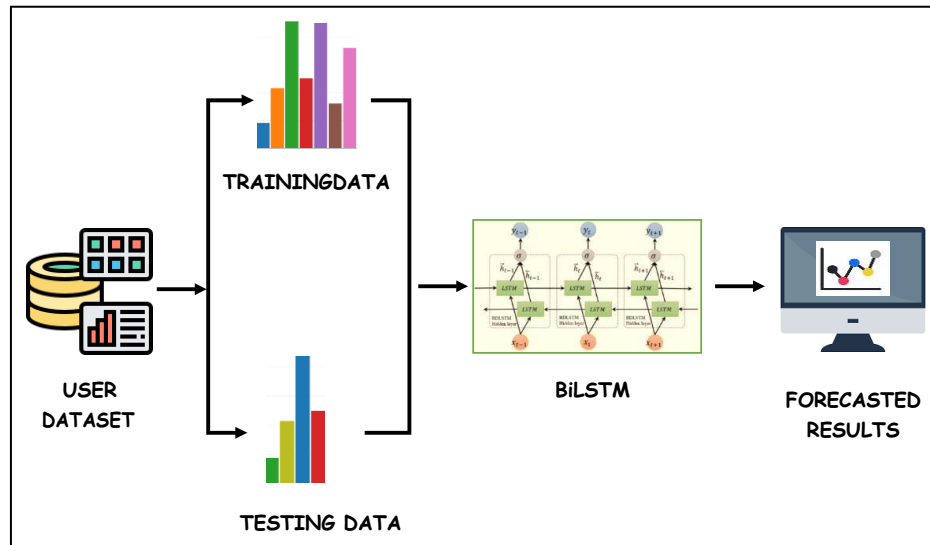


Figure 4.8: Forecasting Module Architecture

CHAPTER 8

SYSTEM IMPLEMENTATION

Our project is deep learning based. This is implemented by making use of python language. The models are developed using many libraries available in python language. The deep learning based model is implemented with the help of numpy, pandas, tensorflow and keras libraries. The trained models are deployed on a web server which is implemented using streamlit library.

Fig.5.1 represents the block diagram. The flow of the application is as follows:

- ✚ The starting part of the application is the User Interface (UI). As the web application is visited by the user, a simple get request is sent from the streamlit frontend to the backend. This request signals the backend server to run the model file.
- ✚ Our Bi-LSTM requires some input dataset for the model to predict the future values. This data is then converted into the required format and sent as the input for the model to get the required predictions for 24 hours ahead.
- ✚ Once this data is provided to the model it starts the calculation and provides the necessary output.
- ✚ When the models provide the output, the server sends these outputs as the response to the request and these values are displayed in the User Interface.

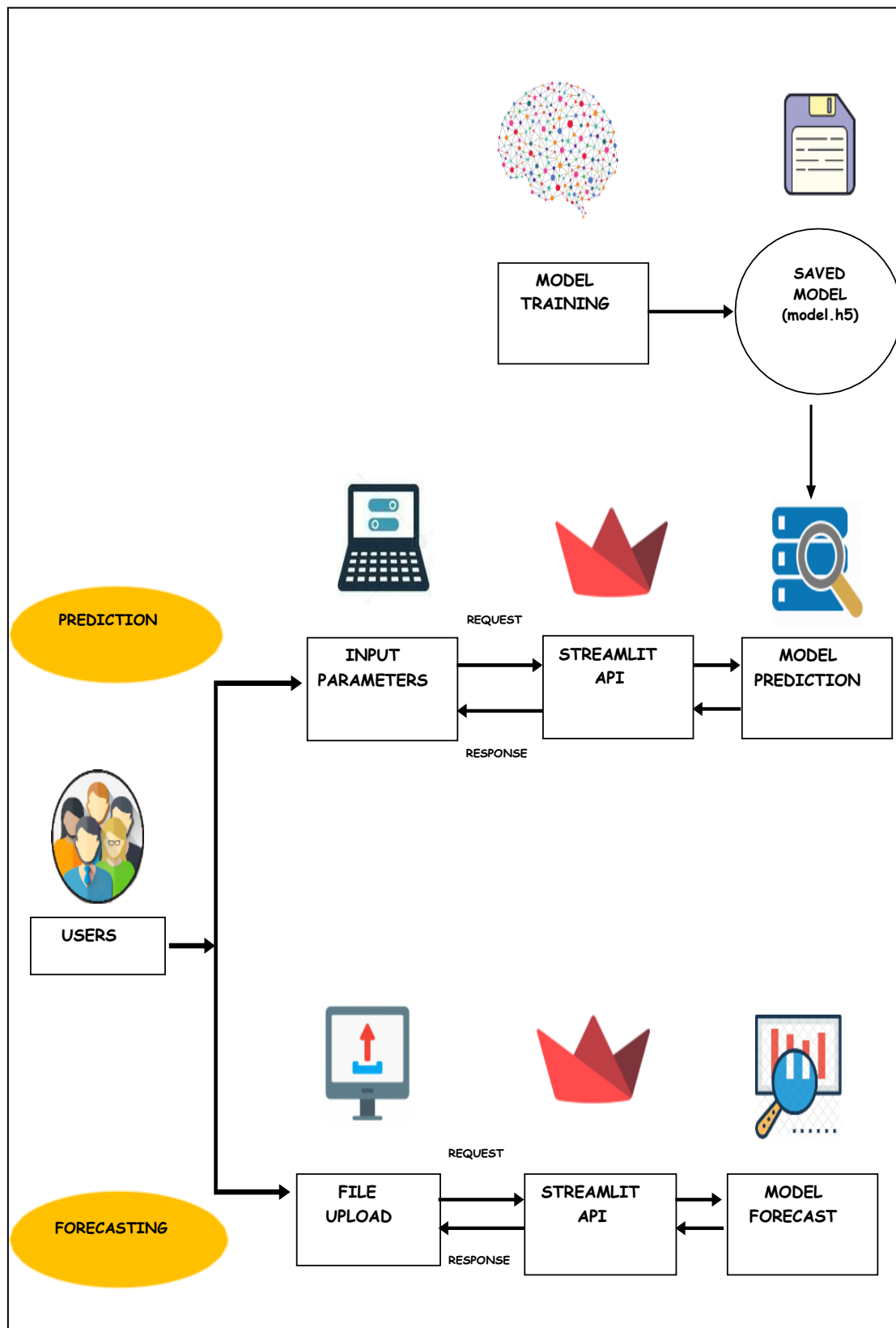
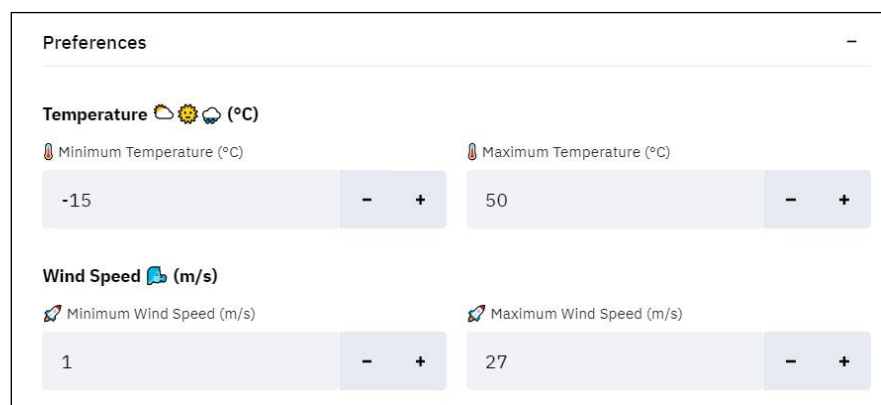


Figure 8.1: Block Diagram (flow of the application)

CHAPTER 9

ADVANTAGES OF THE PROPOSED SYSTEM

- ✚ The RNN can process any amount of data which is very useful in power forecasting because it is based on previous data.
- ✚ The neural network model presumes inputs as independent of each other and for every input the neural network layers and the output are also calculated independently, but in case of RNN as we use Bi-LSTM (Bidirectional Long Short-Term Memory) the computed output of the previous layer becomes the input of the next layer thereby forming a sequence of tasks which is useful in prediction and forecasting.
- ✚ The other main advantage of RNN is the model size does not depend on size of input. Accurate results with minimum loads.
- ✚ User can also set their preferable cut in (minimum) and cut out (maximum) values for wind speed and temperature with the help of preference tab.



The image shows a 'Preferences' window with a title bar and a close button. It contains two sections: 'Temperature' and 'Wind Speed'. Each section has a title with a weather icon and a unit, followed by two input fields for minimum and maximum values, each with minus and plus adjustment buttons.

Temperature ☁️🌞🌧️ (°C)			
🔧 Minimum Temperature (°C)	-15	-	+
🔧 Maximum Temperature (°C)	50	-	+

Wind Speed 🌬️ (m/s)			
🔧 Minimum Wind Speed (m/s)	1	-	+
🔧 Maximum Wind Speed (m/s)	27	-	+

Figure 9.1: Preferences Tab

- ✚ Our Deep Wind is individually different from other wind power forecasting websites where the user can upload their own real time dataset (csv or xls format with a minimum of 30 entries) for forecasting.

✚ Accurate Results with minimum load time.



CHAPTER 7

PROJECT DEVELOPMENT PHASE

The goal of an effective user interface is to make the experience of a user very simple and interactive, requiring minimal effort from the user to achieve the maximum desired results. Fig.7.1 represents our user interface consisting of three tabs: home page, user defined prediction, and forecasting tab (Sprint 1&2).

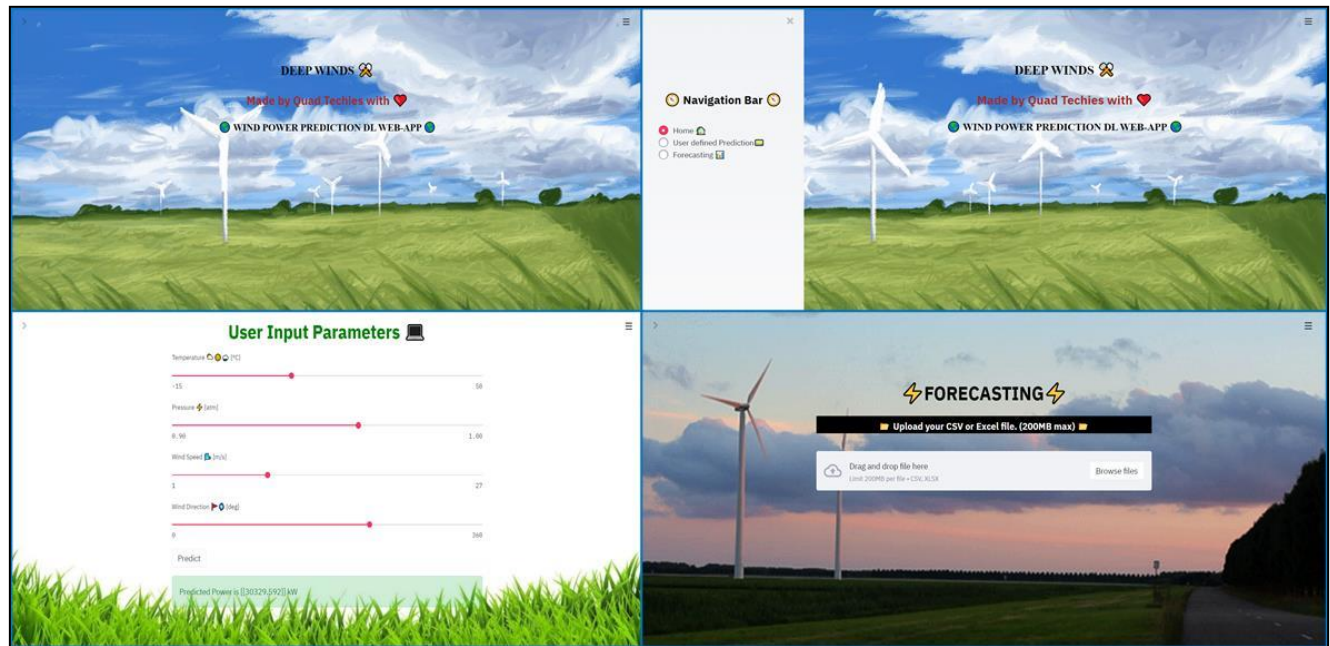


Figure 7.1: Sprint 1&2(User Interface)

Our Deepwind provides an interactive interface with a simple visualization tool which brings the accurate results with minimal load time. Fig.7.2 shows the result obtained from the forecasting (Sprint 3&4).

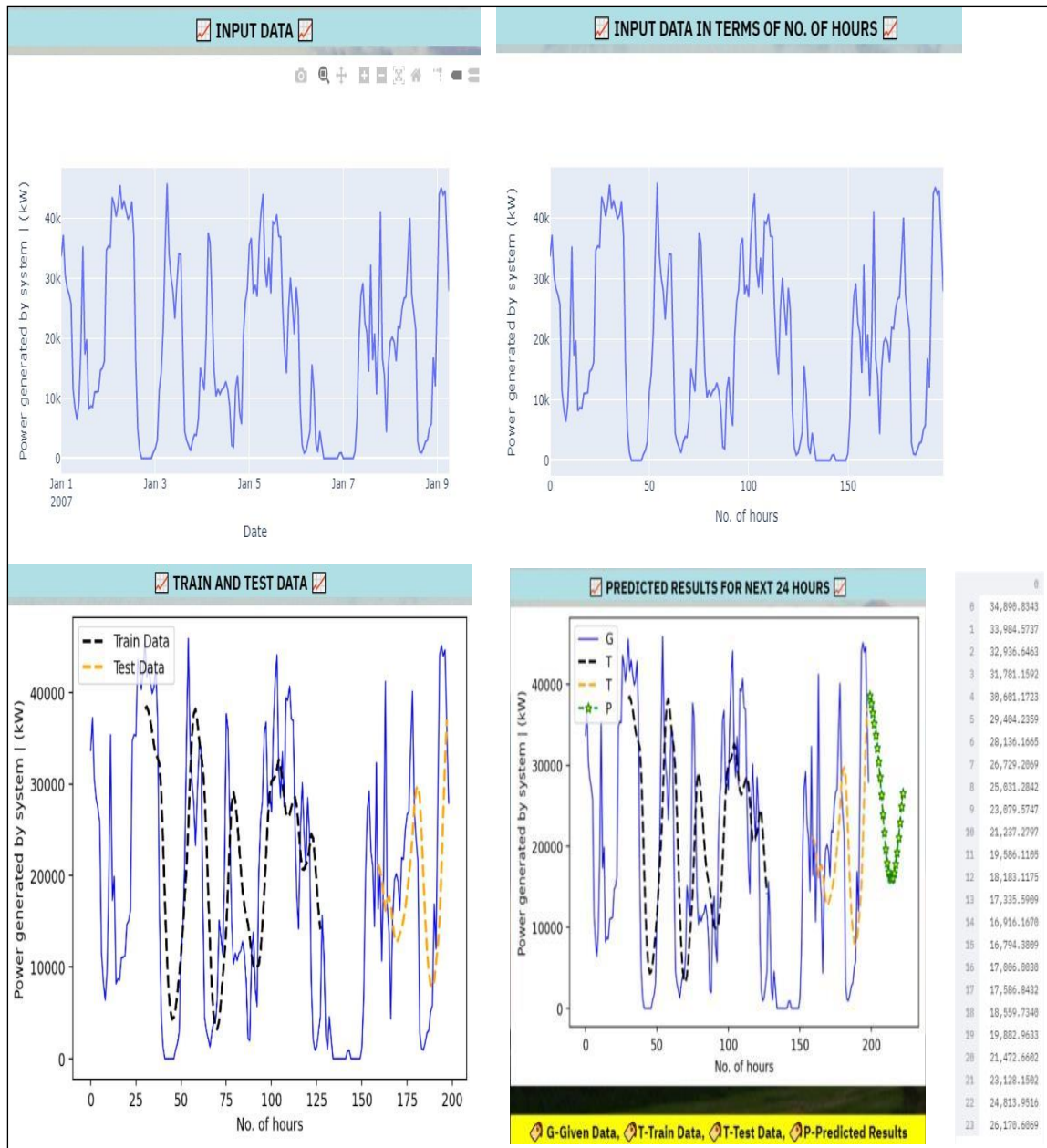
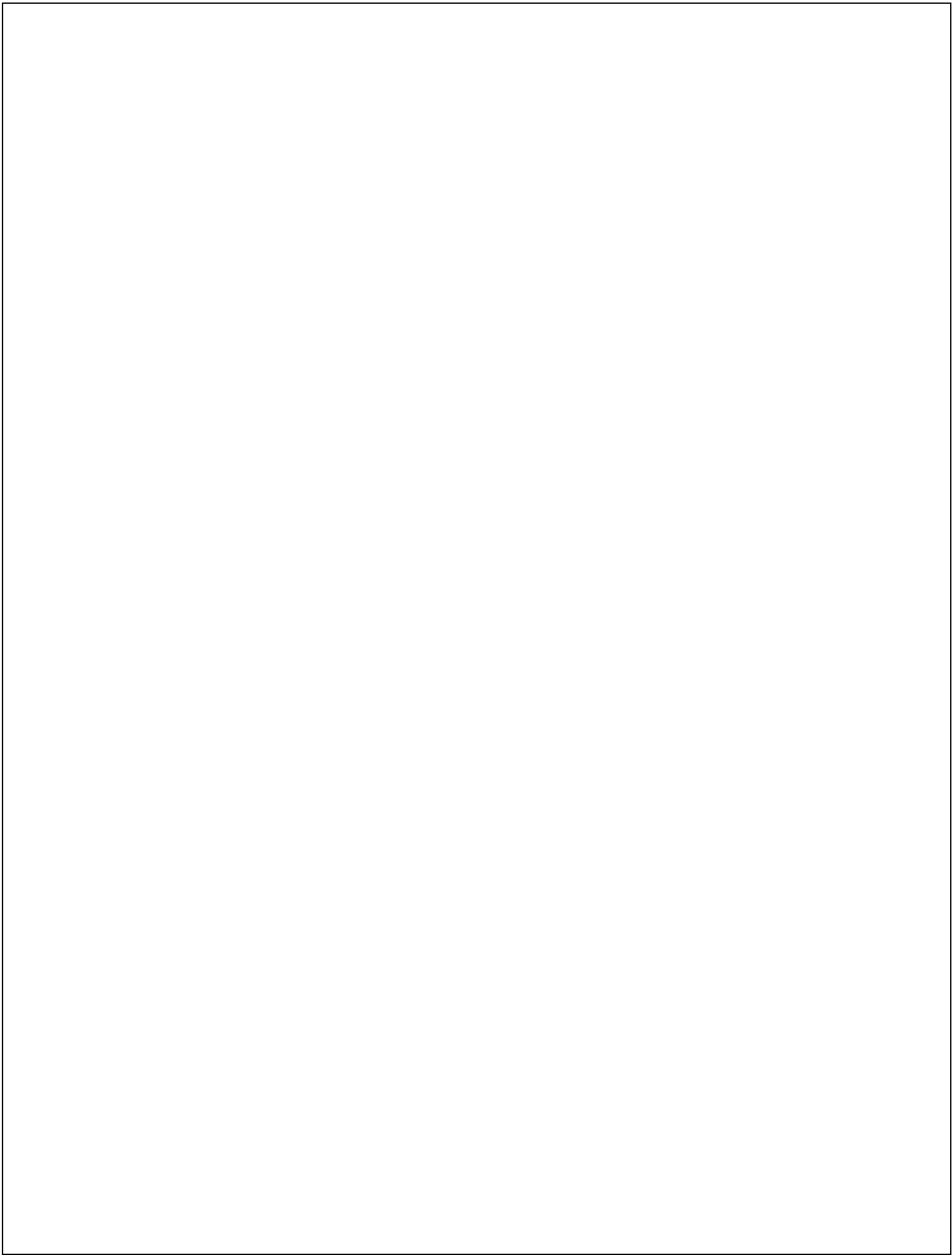


Figure 7.2: Sprint 3 & 4(Forecasting Results)

CHAPTER 10

CONCLUSION AND FUTURE SCOPE

Thus accurate wind power forecasting plays a key role in dealing with the challenges of power system operation under uncertainties in an economical and technical way. This unique approach would surely open up new avenues and make wind farm data more reliable and precise. In our application only weather parameters are considered. Other features can also be added by training the model and integrating it with the current application. Making our website to send alerting message if the forecasted power drops to minimum value. Hopefully, the power of Deep Learning would boost the mass adoption of wind power and turn it into a popular alternative to traditional sources of electricity over the years.



CHAPTER 11

APPENDIX

➤ **app.py**

```
import numpy as np
import streamlit as st
import pandas as pd
import datetime
import plotly.graph_objects as go
import base64
import time
import tensorflow

st.set_page_config(
    page_title=" DEEP WIND ",
    page_icon=""
)
old_models = tensorflow.keras.models.load_model('model.h5')

# set background, use base64 to read local file
def get_base64_of_bin_file(bin_file):
    with open(bin_file, 'rb') as f:
        data = f.read()
    return base64.b64encode(data).decode()

def set_png_as_page_bg(png_file):
    bin_str = get_base64_of_bin_file(png_file)
    page_bg_img = """
<style>
    body {
        background-image: url("data:image/png;base64,%s");
        background-size: cover;
    }
</style>
    """ % bin_str

st.markdown(page_bg_img, unsafe_allow_html=True)
return
set_png_as_page_bg('gr.gif')

def home():
    return "welcome"

def predict(temperature, pressure, wind_speed, wind_direction):
    values = np.array([[temperature, pressure, wind_speed, wind_direction]])
```

```

prediction=old_models.predict(values.reshape(-1,1,4),batch_size=1)
print(prediction)
return prediction

```

```
def main():
```

```

st.sidebar.markdown("<h1 style='text-align: center; color: black;'>Navigation Bar</h1>",
unsafe_allow_html=True)

```

```

    nav = st.sidebar.radio("",["Home ", "User defined Prediction", "Forecasting "]) if nav
    == "Home":

```

```

st.markdown("<h1 style = 'color:black; text_align:center;font-family:times new roman;font-
size:20pt; font-weight: bold;'>DEEP WINDS</h1>", unsafe_allow_html=True)

```

```

st.markdown("<h1 style=' color:brown; text_align:center;font-weight: bold;font-size:19pt;'>Made
by Quad Techies with</h1>", unsafe_allow_html=True)

```

```

st.markdown("<h1 style = 'color:black; text_align:center;font-family:times new roman;font-weight:
bold;font-size:16pt;'>WIND POWER PREDICTION DL WEB-APP</h1>",
unsafe_allow_html=True)

```

```
    if nav == "User defined Prediction":
```

```

set_png_as_page_bg('gra (1).jpg')

```

```

st.markdown("<h1 style='text-align: center; color: green;'>User Input Parameters</h1>",
unsafe_allow_html=True)

```

```

with st.beta_expander("Preferences"):

```

```

st.markdown("<h1 style='text-align: left; font-weight:bold;color:black;background-
color:white;font-size:11pt;'> Temperature(°C) </h1>",unsafe_allow_html=True)

```

```

    col1,col2 = st.beta_columns(2)

```

```

    with col1:

```

```

min_temp=st.number_input('Minimum Temperature (°C)',min_value=-
89,max_value=55,value=-15,step=1)

```

```

    with col2:

```

```

max_temp=st.number_input('Maximum Temperature (°C)',min_value=-
88,max_value=56,value=50,step=1)

```

```

st.markdown("<h1 style='text-align: left; font-weight:bold;color:black;background-
color:white;font-size:11pt;'> Wind Speed(m/s) </h1>",unsafe_allow_html=True)

```

```

    col1,col2 = st.beta_columns(2)

```

```

    with col1:

```

```

min_speed=st.number_input('Minimum Wind Speed
(m/s)',min_value=0,max_value=99,value=1,step=1)

```

```

    with col2:

```

```

max_speed=st.number_input('Maximum Wind Speed
(m/s)',min_value=2,max_value=100,value=27,step=1)

```

```

st.write("")

```

```

    temperature = st.slider('Temperature[°C]', min_value=min_temp, step=1,
max_value=max_temp,value=max_temp)

```

```

    pressure =st.slider('Pressure[atm]', 0.9, 1.0, 1.0)

```



```

wind_speed = st.slider('Wind Speed[m/s]', min_value=min_speed, step=1,
max_value=max_speed,value=max_speed)
wind_direction = st.slider('Wind Direction [deg]', 0, 1, 360) result
=""
if st.button("Predict"):
    result = predict(temperature,pressure,wind_speed,wind_direction)
st.balloons()
st.success('Predicted Power is {} kW'.format(result))

if nav == "Forecasting":
set_png_as_page_bg('04.gif')
st.markdown("<h1 style='text-align: center; color:black ;>FORECASTING</h1>",
unsafe_allow_html=True)
# Setup file upload
st.markdown("<h1 style='text-align:center; color:white;background-color:black;font-
size:14pt>Upload your CSV or Excel file. (200MB max)</h1>", unsafe_allow_html=True)
uploaded_file = st.file_uploader(label="",type=['csv', 'xlsx'])
global df
if uploaded_file is not None:
    print(uploaded_file)
st.markdown("<h1 style='text-align:center; color:black;background-color:lightgreen;font-
size:14pt>File upload successful</h1>", unsafe_allow_html=True)
try:
    df =pd.read_csv(uploaded_file)
st.write(df)

except Exception as e:
    df = pd.read_excel(uploaded_file)
st.write(df)

st.markdown("<h1 style='text-align: center; color:black ;background-color:powderblue;font-
size:14pt>INPUT DATA</h1>",unsafe_allow_html=True)

trace = go.Scatter(x = df['DateTime'], y = df['Power generated by system | (kW)'],mode =
'lines',name = 'Data')
layout = go.Layout(title = "",xaxis = {'title' : "Date"},yaxis = {'title' : "Power generated by
system | (kW)"})
fig = go.Figure(data=[trace], layout=layout)
st.write(fig)

df1=df.reset_index()['Power generated by system | (kW)']
import matplotlib.pyplot as plt
st.write("\n")

```

```

st.markdown("<h1 style='text-align: center; color:black ;background-color:powderblue;font-size:14pt'>INPUT DATA IN TERMS OF NO. OF HOURS</h1>",
unsafe_allow_html=True)
    trace = go.Scatter(x = df1.index,y = df1['Power generated by system | (kW)'],mode = 'lines',
name = 'Data' )
    layout = go.Layout(title = "",xaxis = {'title' : "No. of hours"},yaxis = {'title' : "Power
generated by system (kW)"})

    fig = go.Figure(data=[trace], layout=layout)
    #fig.show()
st.write(fig)
    from sklearn.preprocessing import MinMaxScaler
    scaler=MinMaxScaler(feature_range=(0,1))
    df1=scaler.fit_transform(np.array(df1).reshape(-1,1))
    ##splitting dataset into train and test split
training_size=int(len(df1)*0.65)
test_size=len(df1)-training_size
    train_data,test_data=df1[0:training_size:],df1[training_size:len(df1),:1]

    import numpy
    # convert an array of values into a dataset matrix
    # convert an array of values into a dataset matrix
    def create_dataset(dataset, time_step=1):
        dataX, dataY = [], []
        for i in range(len(dataset)-time_step-1):
            a = dataset[i:(i+time_step), 0] ###i=0, 0,1,2,3-----99 100
            dataX.append(a)
            dataY.append(dataset[i + time_step, 0])
        return numpy.array(dataX), numpy.array(dataY)
    # reshape into X=t,t+1,t+2,t+3 and Y=t+4
time_step = 30
X_train, y_train = create_dataset(train_data, time_step)
X_test, ytest = create_dataset(test_data, time_step)
    # reshape input to be [samples, time steps, features] which is required for LSTM
X_train=X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)
    # Create the BILSTM model
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.layers import Dense
    from tensorflow.keras.layers import LSTM
    from tensorflow.keras.layers import Bidirectional
    model = Sequential()
model.add(Bidirectional(LSTM(250, input_shape=(1, 30))))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam')

```

```

model.fit(X_train,y_train,validation_data=(X_test,ytest),epochs=10,batch_size=64,verbose=1)
import tensorflow as tf
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
#Transformback to original form
train_predict=scaler.inverse_transform(train_predict)
test_predict=scaler.inverse_transform(test_predict)
#### Calculate RMSE performance metrics
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(y_train,train_predict))
#### Test Data RMSEmath.sqrt(mean_squared_error(ytest,test_predict))
#### Plotting
# shift train predictions for plotting
look_back=30
trainPredictPlot = numpy.empty_like(df1)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
# shift test predictions for plotting
testPredictPlot = numpy.empty_like(df1)
testPredictPlot[:, :] = numpy.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(df1)-1, :] = test_predict
# plot baseline and predictions
st.markdown("<h1 style='text-align: center; color:black ;background-color:powderblue;font-size:14pt'>TRAIN AND TEST DATA</h1>",unsafe_allow_html=True)

#plt.plot(scaler.inverse_transform(df1))
plt.plot(scaler.inverse_transform(df1), color="blue", linewidth=1, linestyle="-")
plt.xlabel('No. of hours')
# Set the y axis label of the current axis.
plt.ylabel('Power generated by system | (kW)')
plt.plot(trainPredictPlot,label='Train Data',color="black",linewidth=2, linestyle="--")
plt.plot(testPredictPlot,label='Test Data',color="orange",linewidth=2, linestyle="--")
plt.legend(loc="upper left")
#plt.show()
st.pyplot(plt)

x_input=test_data[len(test_data)-30:].reshape(1,-1)
temp_input=list(x_input)
temp_input=temp_input[0].tolist()
# demonstrate prediction for next 24 hours
from numpy import array
lst_output=[]
n_steps=30
i=0

```

```

        while(i<24):
            if(len(temp_input)>30):
                #print(temp_input)
            x_input=np.array(temp_input[1:])
            x_input=x_input.reshape(1,-1)
            x_input = x_input.reshape((1, n_steps, 1))
            yhat = model.predict(x_input, verbose=0)
            temp_input.extend(yhat[0].tolist())
            temp_input=temp_input[1:]
            lst_output.extend(yhat.tolist())
            i=i+1
        else:
            x_input = x_input.reshape((1, n_steps,1))
            yhat = model.predict(x_input, verbose=0)
            print(yhat[0])
            temp_input.extend(yhat[0].tolist())
            print(len(temp_input))
            lst_output.extend(yhat.tolist())
            i=i+1

    print(lst_output)
    day_new=np.arange(1,31)
    day_pred=np.arange(len(df1),len(df1)+24)
    import matplotlib.pyplot as plt
    print(len(df1))
    progress=st.progress(0)
    for i in range(100):
        time.sleep(0.1)
        progress.progress(i+1)
        st.balloons()
        st.markdown("<h1 style='text-align: center; color:black ;background-color:powderblue;font-size:14pt'>PREDICTED RESULTS FOR NEXT 24 HOURS</h1>",
        unsafe_allow_html=True)
        plt.plot(day_pred,scaler.inverse_transform(lst_output),color="green",linewidth=1.5, linestyle="--",marker='*',markerfacecolor='yellow', markersize=7)
        plt.legend('GTTP',loc="upper left")

    plt.xlabel('No. of hours')
    # Set the y axis label of the current axis.
    plt.ylabel('Power generated by system | (kW)')

    st.pyplot(plt)
    st.markdown("<h1 style='text-align: center; color:black ;background-color:yellow;font-size:14pt'>G-Given Data, \nT-Train Data, \nT-Test Data, \nP-Predicted Results</h1>",
    unsafe_allow_html=True)

```

```
st.write(scaler.inverse_transform(lst_output))
```

```
if __name__ == "__main__":  
    main()
```

➤ **model.py**

```
import pandas as pd  
import datetime  
import numpy as np  
from keras.models import Sequential  
from keras.layers import Dense  
from keras.layers import LSTM  
from keras.layers import Bidirectional  
import pandas as pd  
import keras  
''' Loading data '''  
df = pd.read_excel('Dataset.csv')  
df=df.drop(columns=['DateTime'])  
''' Cleaning Data '''  
#dataframe.drop['Date'].values  
df['Power generated by system | (kW)'].replace(0, np.nan, inplace=True)  
df['Power generated by system | (kW)'].fillna(method='ffill', inplace=True)  
  
X = df.drop(columns=['Power generated by system | (kW)'])  
Y = df[['Power generated by system | (kW)']]  
X=np.array(X).reshape(-1,1,4)  
Y=np.array(Y).reshape(-1,1,1)  
  
model =Sequential()  
model.add(Bidirectional(LSTM(100, activation='relu',input_shape=(-1,1,4))))  
model.add(Dense(1))  
model.compile(loss='mae', optimizer='adam',metrics=['accuracy'])  
model.fit(X, Y,epochs=100,callbacks=[keras.callbacks.EarlyStopping(patience=3)])  
  
test_data = np.array([[ -4.858,0.989741,6.651,273]])  
o=model.predict(test_data.reshape(-1,1,4), batch_size=1)  
print(o)  
  
# Saving model to disk mod
```

Demo link:

https://drive.google.com/file/d/1BdO91TsJlvGS1crCzrb4PMk-iCUZdbO-/view?usp=share_link

Github link:

<https://github.com/IBM-EPBL/IBM-Project-8578-1658925086>







