Abstract

The extraction of energy from renewable sources is rapidly growing. The current pace of technological development makes it commercially viable to harness energy from sun, wind, geothermal and many other renewable sources. Because of the negative effects on the environment and the economy, conventional energy sources like natural gas, crude oil and coal are coming under political and economic pressure. Thus, they require a better mix of energy sources with a higher percentage of renewable energy sources. Harnessing energy from renewable sources range from small scale (e.g., a single household) to large scale (e.g., power plants producing several MWs to a few GWs providing energy to an entire city). An inherent characteristic common to all renewable power plants is that power generation is dependent on environmental parameters and thus cannot be fully controlled or planned for in advance. In a power grid, it is necessary to predict the amount of power that will be generated in the future, including those from the renewable sources, as fluctuations in capacity and/or quality can have negative impacts on the physical health of the entire grid as well as the quality of life of its users. In addition, management of the smart grid, in which the renewable energy plants are integrated, is also a challenging problem. In this project, we are going to applied different machine learning techniques used to address the above issues related to renewable energy generation and finalize the best fit model to deploy the project.

Keywords: Renewable Energy, solar, wind, power generation. Smart Grids, Machine Learning.

CHAPTER 1
Introduction

The world is faced with a number of challenges related to energy sustainability and security. If not promptly addressed, these can lead to economic and political instability. The depletion of fossil fuel reserves as well as the environmental impact of burning these fuels have led to increased interest in developing alternative and more sustainable energy sources. Renewable energy resources like solar photovoltaic (PV), solar thermal (a.k.a. concentrated solar power, CSP), geothermal, tidal waves, wind power, and biomass have been growing rapidly in energy market. Many countries and companies are seeking to diversify their energy mix by increasing the share of renewables.

In conventional energy generation process, energy production depends on the energy demand from the users, and the stability of the power grid relies on the equilibrium of energy demand and supply. When the energy demand surpasses the energy supply, it destabilizes the power grid and results in power quality degradation and/or blackouts in some parts of the grid. When the demand is lower than

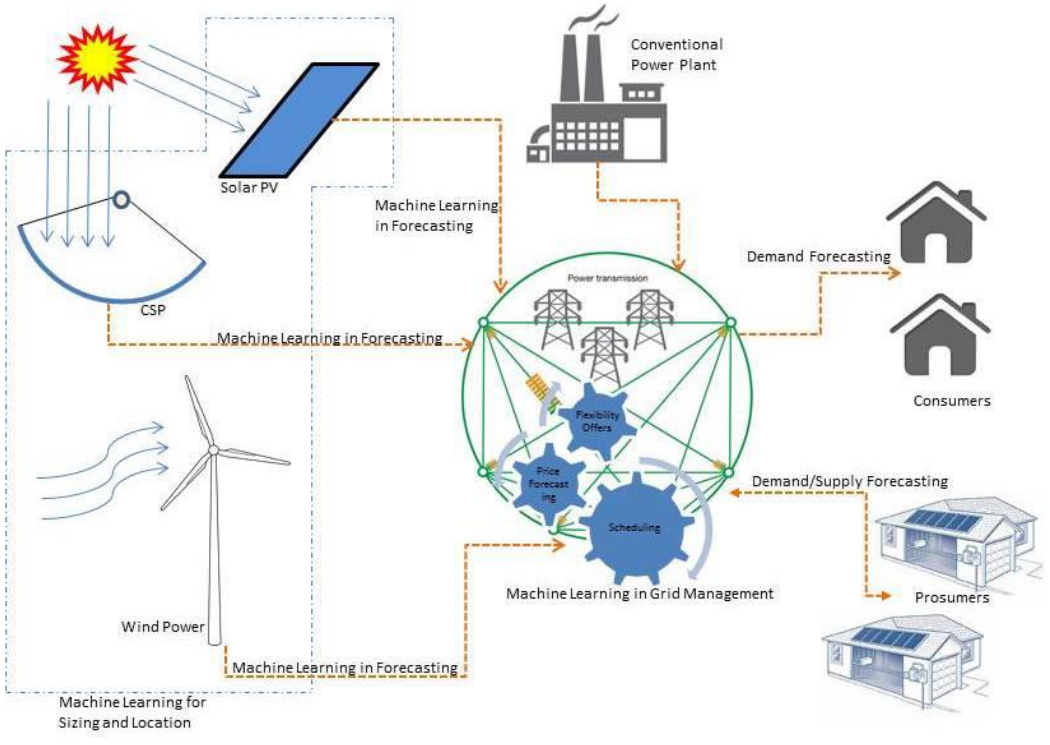


Figure 1 Overview of power grid with integrated renewable sources and its usage

the supply, energy is lost incurring high unnecessary costs due to wastage. Producing the right amount of energy at the right time is crucial both for the smooth running of the grid and for higher economic benefits. To maintain this stability, much research has focused on energy supply and demand forecasting to predict the amount of energy that will be required. This will then ensure that there will be sufficient capacity to meet these requirements, but also that excess capacity and hence energy wasted will be minimized.

Different machine learning techniques are used in different stages of a renewable energy-integrated power grid, depending on the requirements and the characteristics of the problem. For a power grid with renewable energy sources contributing a considerable proportion of energy supply, it is necessary to forecast both short and medium term demand. This would facilitate the formulation of well informed energy policies, for example by helping to determine important parameters such as the appropriate spinning reserve levels and storage requirements. On the other hand, it is also necessary to forecast the energy output from renewable energy power plants themselves, since the energy output from these power plants depends on many environmental factors that cannot be controlled.

Renewable energy resources like solar light, solar heat and wind are highly variable and the resulting fluctuations in the generation capacity can cause instability in the power grid. This is because the energy/power output of these plants is defined by the environmental factors such as wind speed, the intensity of solar radiation, cloud cover and other factors. Another important limitation of renewable energy power plants is that they are subject to marked daily and annual cycles (e.g., solar energy is only available during the day). Thus, it is necessary to generate power when resources are available and store it for later use while using a certain portion of the generated power at the same time. Wind and solar PV energy is expensive to store, thus careful management of energy generation is needed. When the generation capacity of natural resources are insufficient to meet demand, conventional sources such as gas power plants are typically used to cover the electricity shortfall.

The above mentioned challenges have motivated the use of machine learning techniques to support better management of energy generation and consumption. Different machine learning techniques are used in different stages of a renewable energy-integrated power grid, depending on the requirements and the characteristics of the problem. For a power grid with renewable energy sources contributing a considerable proportion of energy supply, it is necessary to forecast both short and medium term demand. This would facilitate the formulation of well informed energy policies, for example by helping to determine important parameters such as the appropriate spinning reserve levels and storage requirements. On the other hand, it is also necessary to forecast the energy output from renewable energy power plants themselves, since the energy output from these power plants depends on many environmental factors that cannot be controlled. This in turn necessitates the prediction of these environmental factors such as wind speed, direction and solar radiation in the region of the power plant. An Machine Learning in Renewable Energy 3 other important use for machine learning techniques in the context of renewable energy is in determining the optimal location, size and configuration of renewable power plants. These parameters are dependent on many factors such as proximity to population centers, local climatic fluctuations, terrain, availability and costs of logistics and other facilities and many others. Yet another area for the application of machine learning methods is in the overall operations and management of the smart grid, i.e. issues such as fault detection, control and so on.

1.1 Problem Statement :

In renewable energy generation process, energy production depends on the weather forecasting. Renewable energy resources like solar, wind are highly variable and the resulting fluctuations in the generation capacity can cause instability in the power grid. This is because the power output of these plants is defined by the environmental factors such as wind speed, the intensity of solar radiation, cloud cover and other factors. Limitation of renewable energy power plants is that they are subject to marked daily and annual cycles (e.g., solar energy is only available during the day). Thus, it is necessary to generate power when resources are available and store it for later use while using a certain portion of the generated power at the same time. So, the above limitation challenges to use the machine learning techniques for supporting better management of energy generation and consumption. So, the machine learning techniques help to predict the generation for future power use and also to preserve the energy in a bad situation.

1.2Purpose:

The goal of this project is to find out the Renewable Energy generation on the basis of Weather forecasting. In this Project, we are going to predict the Solar and Wind Generation by taking the co-related features from the weather data. The main objective is to use different machine learning algorithm (Take the best model) and predict the wind and solar generation.

1.3 Scope of the project:

1.3.1 Initial functional requirement will be:-

- Firstly, we have to collect the data (dataset consist of weather data and generation data).
- Importing the dataset then performing the pre-processing i.e. data cleaning.
- Performing the Exploratory Data Analysis.

- Applying the different Machine Learning Models and select the best fit model.

- Analysis of result and making changes in algorithm accordingly.

1.4.2 Initial non-functional requirement will be:-

- Using Flask to deploy the model on web and Pickle to serialize and deserialized the object.

1.5 Machine Learning Life Cycle Model:

Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below

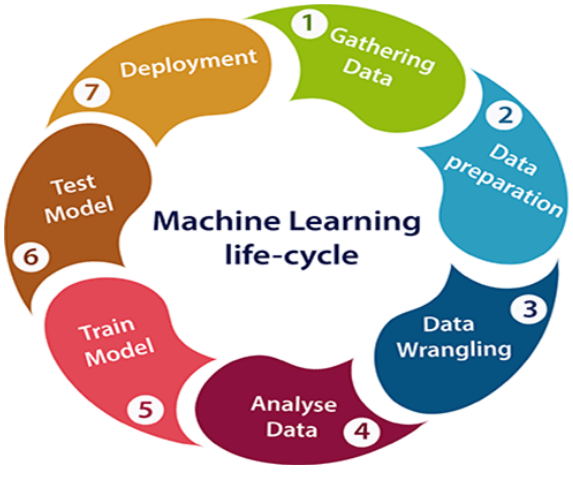


Figure 2 Machine Learning life cycle model

- Gathering Data
- Data preparation
- Data Wrangling
- Analyze Data
- Train the model
- Test the model
- Deployment

1.5.1 Gathering Data:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as files, database, internet, or mobile devices. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

- Identify various data sources
- Collect data
- Integrate the data obtained from different sources

By performing the above task, we get a coherent set of data, also called as a dataset

1.5.2 Data preparation:

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.
In this step, first, we put all data together, and then randomize the ordering of data.
This step can be further divided into two processes:

- **Data exploration:**
  It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.
  A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.
- **Data pre-processing:**
  Now the next step is preprocessing of data for its analysis.

1.5.3 Data Wrangling:

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

- Missing Values
- Duplicate data
- Invalid data
- Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

1.5.4 Data Analysis:

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

- Selection of analytical techniques
- Building models
- Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as Classification, Regression, Cluster analysis, Association, etc. then build the model using prepared data, and evaluate the model.

Hence, in this step, we take the data and use machine learning algorithms to build the model.

1.5.5 Train Model:

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

1.5.6 Test Model:

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

1.5.7 Deployment:

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

CHAPTER 2
Overall Description

This project aims to predict the renewable energy generation by applying different machine learning algorithms and taking the best fit model of them.

- **Functional requirement specification:**

Use case: Prediction of the renewable energy generation like solar and wind power generation.

Initial Step by step description:

- Firstly, we have to collect the data (dataset consist of weather data and generation data).
- Importing the dataset then performing the pre-processing i.e. data cleaning.
- Performing the Exploratory Data Analysis.
- Applying the different Machine Learning Models and select the best fit model.
- Analysis of result and making changes in algorithm accordingly.

- **Dataset :**

The data used in this analysis come from Open Power System Data, a free-of-charge platform with data on installed generation capacity by country/technology, individual power plants (conventional and renewable), and time series data. The platform contains data for 37 European countries, but in this project we are going to focus on data for Germany in 2016.

There are two .csv files use in this project:

Generation data of wind and solar, prices in hourly resolution contains 8784 rows and 2 columns. Weather data with wind speed, radiation, temperature and other measurements. Given the huge amount of data, the Open Power System Data platform provides a CSV file for the German dataset for 2016 contains 2.2 million rows and 14 columns.

1. The Weather data in the weather.csv file contains the following:

- **Wind parameters:**

  - v1: velocity [m/s] @ height h1 (2 meters above displacement height)

  - v2: velocity [m/s] @ height h2 (10 meters above displacement height)

  - v_50m: velocity [m/s] @ 50 meters above ground

  - h1: height above ground [m] (h1 = displacement height +2m)

  - h2: height above ground [m] (h2 = displacement height +10m)

  - z0: roughness length [m]

- **Solar parameters:**

  - SWTDN: total top-of-the-atmosphere horizontal radiation [W/m²]

  - SWGDN: total ground horizontal radiation [W/m²]

- **Temperature data:**

  - T: Temperature [K] @ 2 meters above displacement height (see h1)

- **Air data:**

  - Rho: air density [kg/m³] @ surface

  - p: air pressure [Pa] @ surface

2. The Generation data in the generation.csv file contains the following:

  - DE_solar_generation_actual - Actual solar generation in Germany in MW.

  - DE_wind_generation_actual - Actual wind generation in Germany in MW.

CHAPTER 3
Specification Requirement

3.1.1 Hardware Requirement:

Processor: Intel Dual Core

RAM: Minimum 4GB

OS: Windows, Linux

3.1.2 Software Requirement:

Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, macOS and Linux. To get Navigator, get the Navigator cheat sheet and install Anaconda.

The Navigator Getting started with Navigator section shows how to start Navigator from the shortcuts or from a terminal window.

Following libraries of python should be install:

Numpy – pip install numpy

Pandas – pip install pandas

Seaborn – pip install seaborn

Matplotlib – pip install matplotlib

CHAPTER 4
System Design

5.1 Flowchart of the System:

The flowchart of the system is represented in Figure.

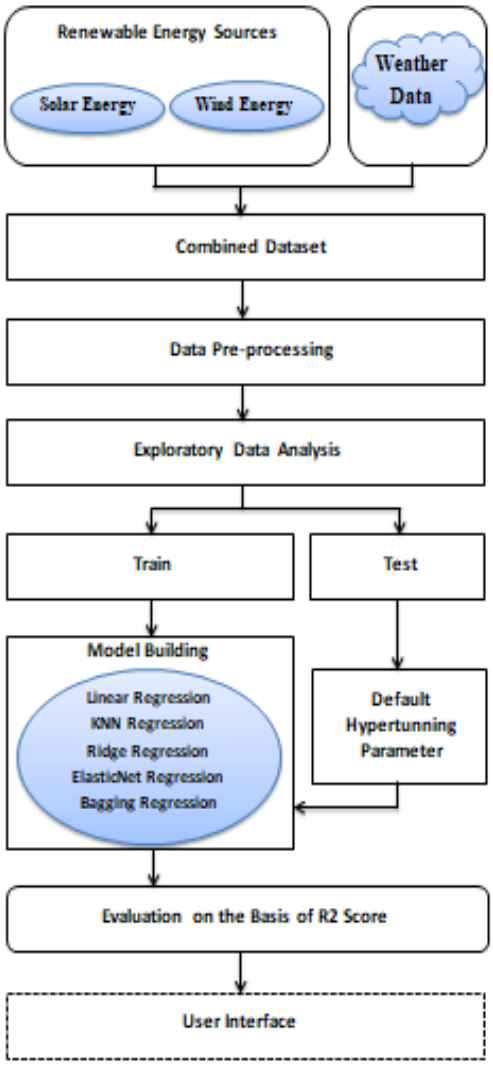


Figure 3 Flowchart of the Project

- **Flowchart Description:**

The above flowchart describes the working flow of the project. We have firstly collect the data from the Open Power System Data Platform. The data is consist of solar and wind generation. We did the Pre-processing on the dataset.

Afterward, the Exploratory Data Analysis is done on the dataset to know the nature of data and correlation between the variables. The data is splitted into train and test split. After on, different models has been tried on the dataset. Then Evaluation has been carried out on the basis of R2 score. The User Interface is created by deploying the model.

CHAPTER 5
Data Preprocessing & Exploratory Data Analysis

- **Data preparation:**

  After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

  In this step, first, we put all data together, and then randomize the ordering of data.

  This step can be further divided into two processes:

  - **Data exploration:**

    It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.

    A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

    Now the next step is preprocessing of data for its analysis.

- **Data pre-processing:**

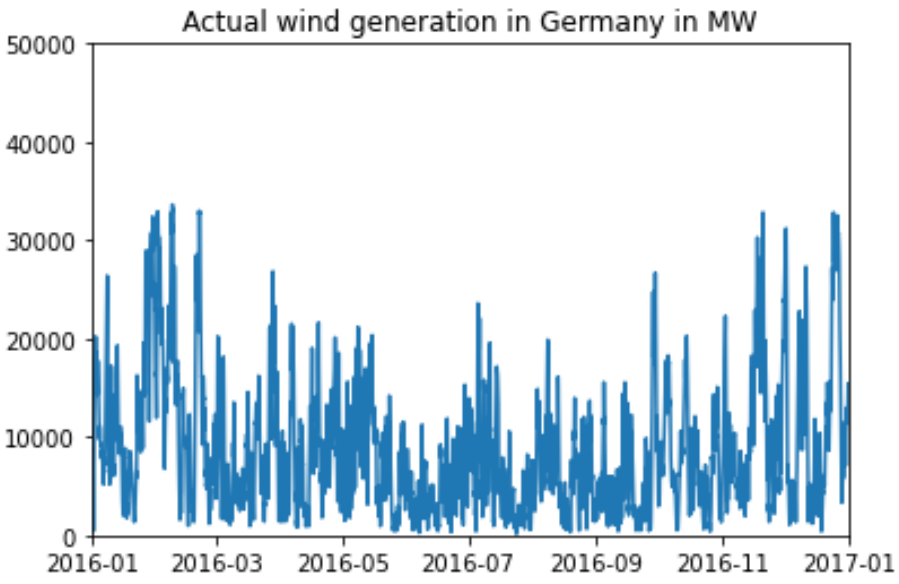
```
# Renewable energy production data
"""

production = pd.read_csv(r"C:\Users\Administrator\Desktop\Renewable_Energy_Production_Project\time_
                         usecols=(lambda s: s.startswith('utc') | s.startswith('DE')),
                         parse_dates=[0], index_col=0)


print(production.shape)
production.head(1)


production = production.loc[production.index.year == 2016, :]

"Drop the NA columns"

production=production.dropna(axis=1, how='all')

print(production.shape)
production.head(1)


production.info()

from sklearn.impute import SimpleImputer
imp = SimpleImputer(strategy='mean')
ProductionImputed = imp.fit_transform(production)

production = pd.DataFrame(ProductionImputed,
                                columns= production.columns,
                                index=production.index)
"There 8784 entries, which correspond to the number of hours in 2016."
```

Figure 4 Data Cleaning

- **Exploratory Data Analysis:**

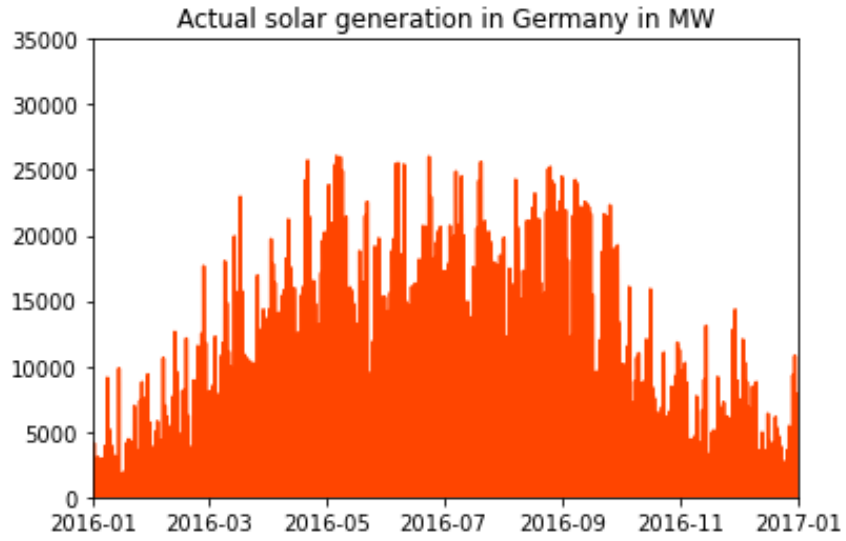
1. Exploratory Data Analysis on Generation data:

1.1 Actual wind generation in Germany in MW:



Above graph shows the time series of Actual Wind Generation in MW. From the graph it has been observed that maximum generation is obtain from Jan 2016 to March 2016.

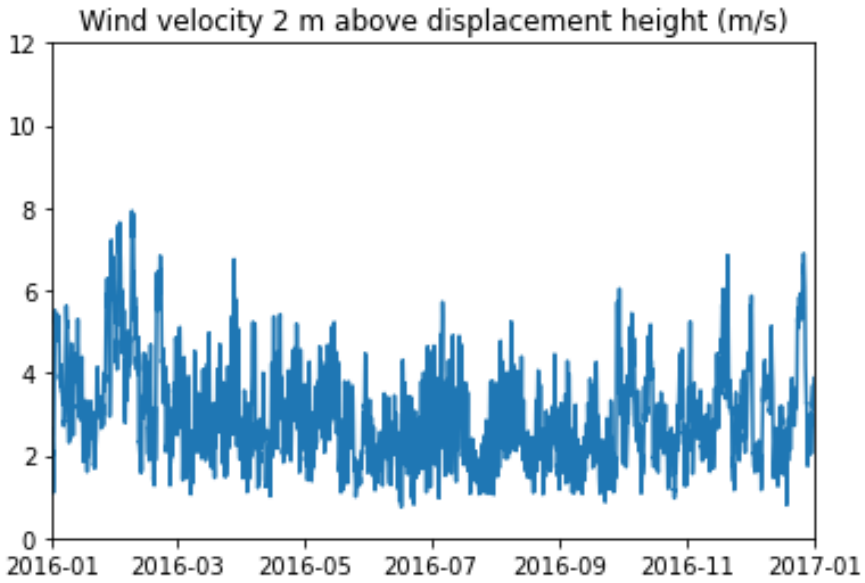
1.2 Actual Solar generation in Germany in MW:



Above graph shows the time series of Actual Solar Generation in MW. From the graph it has been observed that maximum generation is obtain from March 2016 to October 2016.

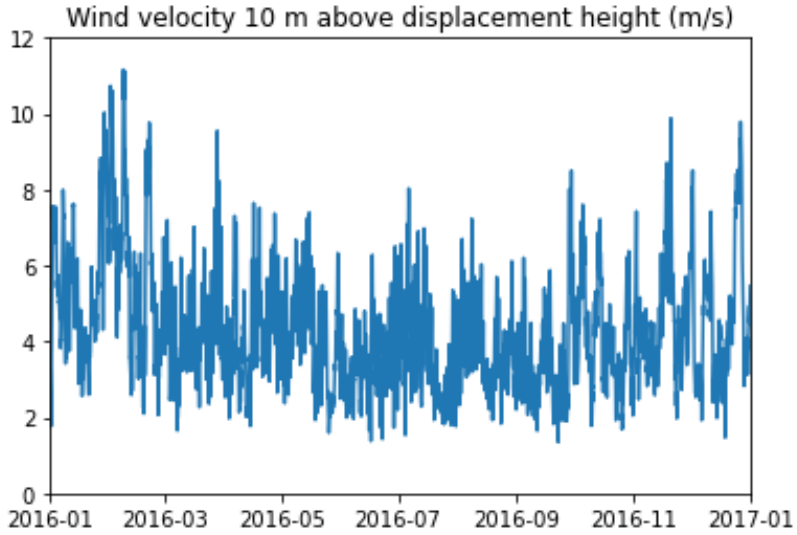
2. Exploratory Data Analysis on Weather data:

2.1 Wind velocity 2m above displacement height (m/s):



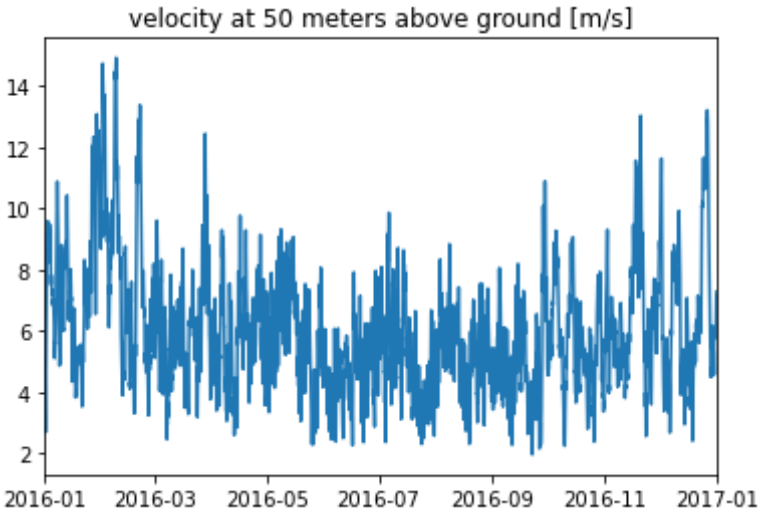
Above graph shows the time series of velocity at 2 meter from surface. The wind velocity above 2 meter in the months of February 2016 to April 2016 and November 2016 to December 2016 is at the peak level and in the mid-year the velocity ranges between 4 to 6 m/s.

2.2 Wind velocity 10m above displacement height (m/s):



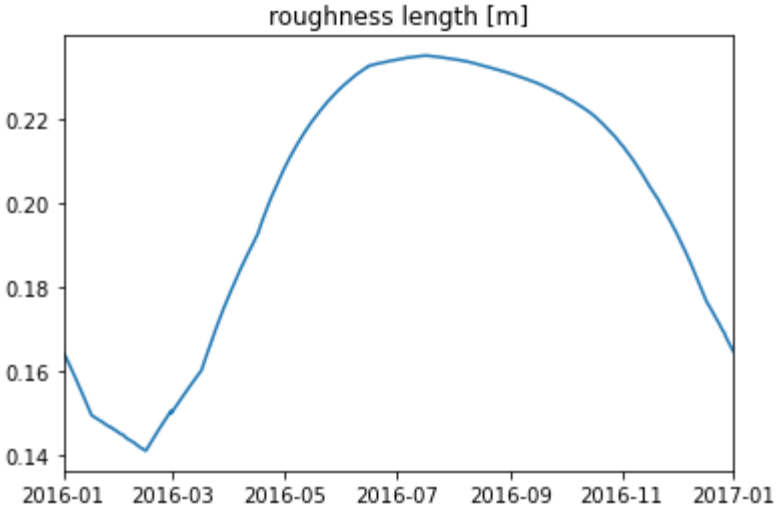
The given graph is for wind velocity above 10 meters from the surface
As the height increases to 10 meters the wind velocity is increased,v2 is at its peak value of 11 m/s in the month of February.

2.3 Wind velocity 50m above displacement height (m/s):



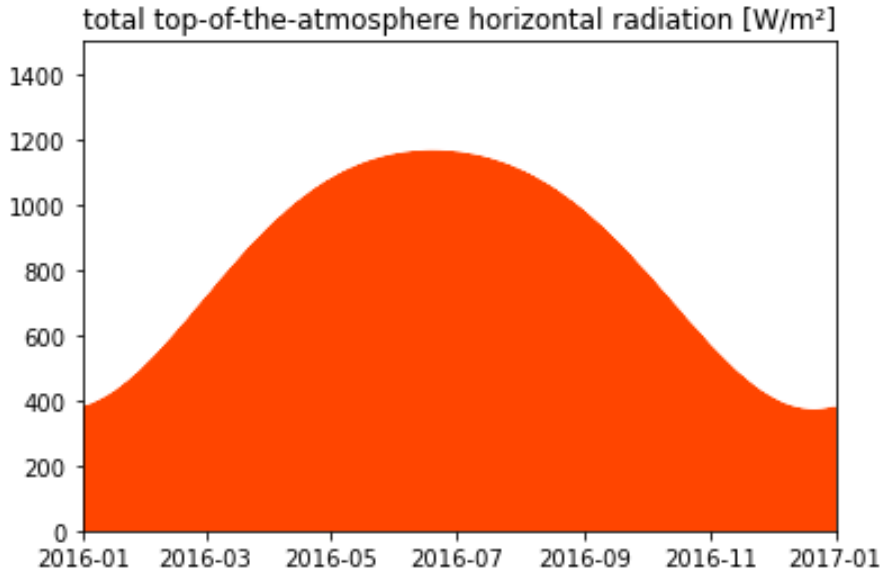
It seems that going upward in the atmosphere results the variation in the wind velocity, as the height goes above 50 meters from the earth there is increase in the velocity. Here also starting and ending months of year results the greater than mid months.

2.4 z0: roughness length [m]:



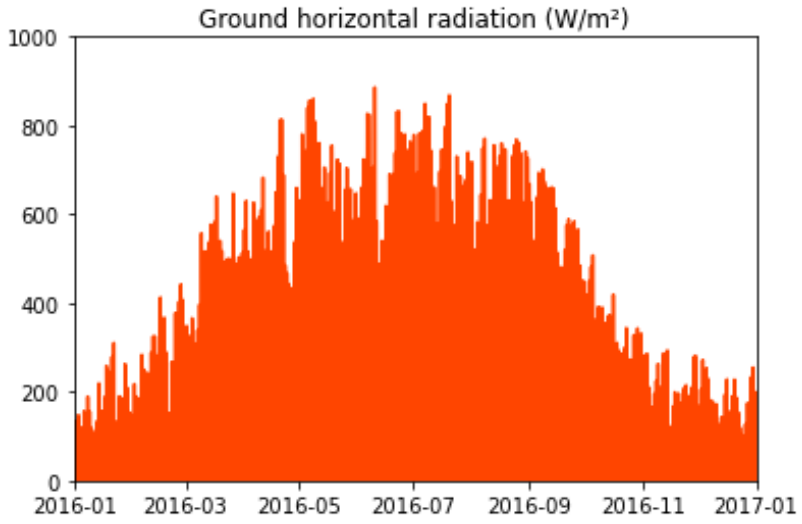
The above graph shows the roughness in meters, the distribution of the roughness is little left skewed, It is increasing from the month march and stays at its peak.at the starting of the year it is minimal.

2.5 Total top-of-the-atmosphere horizontal radiation [W/m²]:



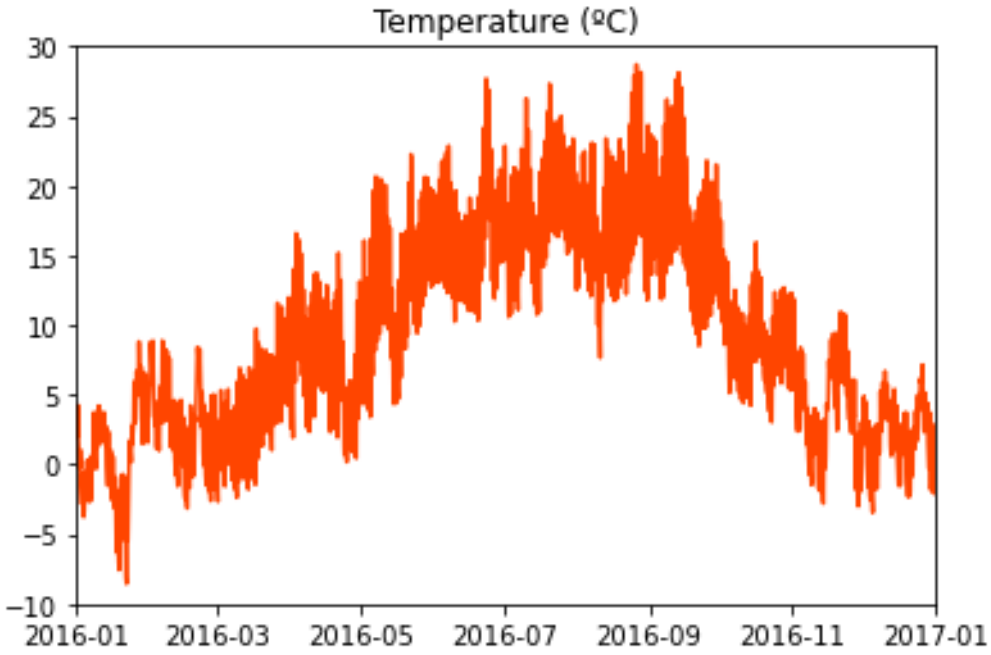
This graph is the horizontal radiation top of the atmosphere
The distribution is similar to that of ground radiation but the intensity is increased upto 1200 W/m2. This is also at its peak in the mid-year.

2.6 Total ground horizontal radiation [W/m²]:



The following is the time series for the Ground horizontal radiation
The radiation appears high in the months of April to September that is in the mid year
Radiation reaches high up to 900 w/m2 in the month of June.

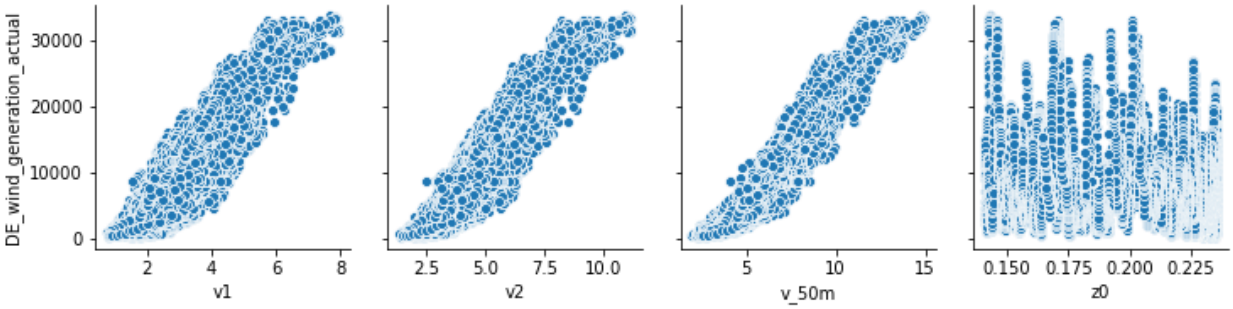
2.6Temperature (C):



Change in temperature results the atmosphere. Above graph shows the temperature in Celsius The maximum temperature is 29C. Again in the months from May to October the the temperature is high

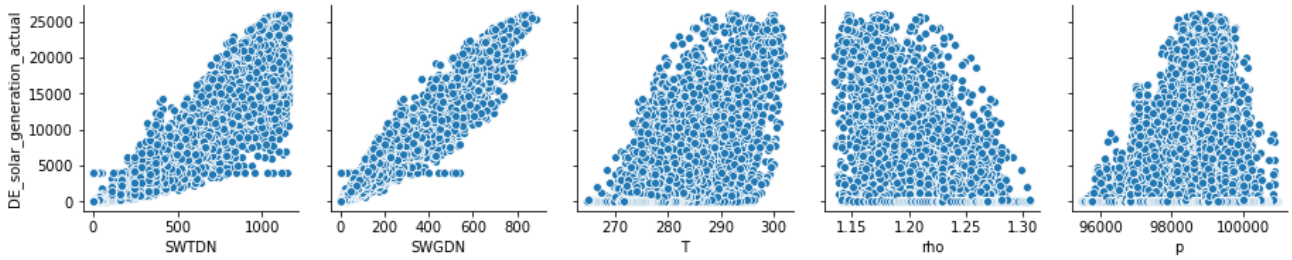
- **Scatter Plot :**

1. Scatter Plot for Wind Generation Parameters:



Above graph shows the Scatter plot for v1,v2, v50 and the roughness z0 for wind generation parameter. So, it has been observed that v1, v2, v_50 are highly co related with the response variable DE_wind_generation_actual and z0 has no co-relation with DE_wind_generation_actual.

2. Scatter Plot for Solar Generation Parameters :



Above graph shows the Scatter plot for SWTDN, SWGDN, T, rho and p for solar generation parameter. So, it has been observed that SWTDN, SWGDN, T are highly co related with the response variable DE_solar_generation_actual whereas rho and p has no co-relation with DE_wind_generation_actual.

CHAPTER 6

Model Building

6.1 Algorithm Research and Selection:

The following algorithm is applied on the dataset:

- Linear Regression
- KNN Regression
- Ridge Regression
- Elastic Net Regression
- Bagging Regression

1. Linear Regression:

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is

changing according to the value of the independent variable.

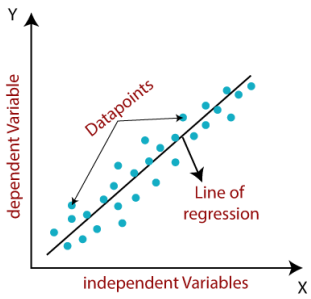


Fig.5Linear Regression

2. KNN Regression:

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems. K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data. It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

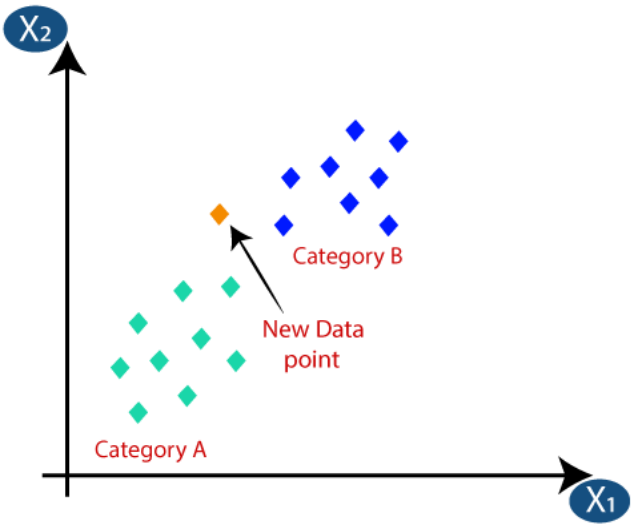


Fig.6.KNN Regression

3.Ridge Regression:

- Ridge regression is one of the types of linear regression in which a small amount of bias is introduced so that we can get better long-term predictions.
- Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as **L2 regularization**.
- In this technique, the cost function is altered by adding the penalty term to it. The amount of bias added to the model is called **Ridge Regression penalty**. We can calculate it by multiplying with the lambda to the squared weight of each individual feature.
- The equation for the cost function in ridge regression will be:

$$\sum_{i=1}^{M}(y_i - y'_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{n}\beta_j * x_{ij}\right)^2 + \lambda\sum_{j=0}^{n}\beta_j{}^2$$

In the above equation, the penalty term regularizes the coefficients of the model, and hence ridge regression reduces the amplitudes of the coefficients that decreases the complexity of the model. As we can see from the above equation, if the values of **λ tend to zero, the equation becomes the cost function of the linear regression model.** Hence, for the minimum value of λ, the model will resemble the linear regression model. A general linear or polynomial regression will fail if there is high collinearity between the independent variables, so to solve such problems, Ridge regression can be used. It helps to solve the problems if we have more parameters than samples.

3. Elastic Net Regression :

Elastic net is basically a combination of both L1 and L2 regularization. So if you know elastic net, you can implement both Ridge and Lasso by tuning the parameters. So it uses both L1 and L2 penalty term, therefore its equation look like as follows:

$$\min\left(\left|\left|Y - X\theta\right|\right|_2^2 + \lambda_1||\theta||_1 + \lambda_2||\theta||_2^2\right)$$

Linear regression refers to a model that assumes a linear relationship between input variables and the target variable.

With a single input variable, this relationship is a line, and with higher dimensions, this relationship can be thought of as a hyperplane that connects the input variables to the target variable. The coefficients of the model are found via an optimization process that seeks to minimize the sum squared error between the predictions (*yhat*) and the expected target values (*y*).

- loss = sum i=0 to n (y_i – yhat_i)^2

A problem with linear regression is that estimated coefficients of the model can become large, making the model sensitive to inputs and possibly unstable. This is particularly true for problems with few observations (*samples*) or more samples (*n*) than input predictors (*p*) or variables (so-called *p >> n problems*).

One approach to addressing the stability of regression models is to change the loss function to include additional costs for a model that has large coefficients. Linear regression models that use

these modified loss functions during training are referred to collectively as penalized linear regression.

One popular penalty is to penalize a model based on the sum of the squared coefficient values. This is called an L2 penalty. An L2 penalty minimizes the size of all coefficients, although it prevents any coefficients from being removed from the model.

- l2_penalty = sum j=0 to p beta_j^2
  Another popular penalty is to penalize a model based on the sum of the absolute coefficient values. This is called the L1 penalty. An L1 penalty minimizes the size of all coefficients and allows some coefficients to be minimized to the value zero, which removes the predictor from the model.

- l1_penalty = sum j=0 top abs(beta j)
  Elastic net is a penalized linear regression model that includes both the L1 and L2 penalties during training.

4. Bagging Regression :

The idea behind bagging is combining the results of multiple models (for instance, all decision trees) to get a generalized result. Here's a question: If you create all the models on the same set of data and combine it, will it be useful? There is a high chance that these models will give the same result since they are getting the same input. So how can we solve this problem? One of the techniques is bootstrapping.

Bootstrapping is a sampling technique in which we create subsets of observations from the original dataset, **with replacement**. The size of the subsets is the same as the size of the original set. Bagging (or Bootstrap Aggregating) technique uses these subsets (bags) to get a fair idea of the distribution (complete set). The size of subsets created for bagging may be less than the original set.

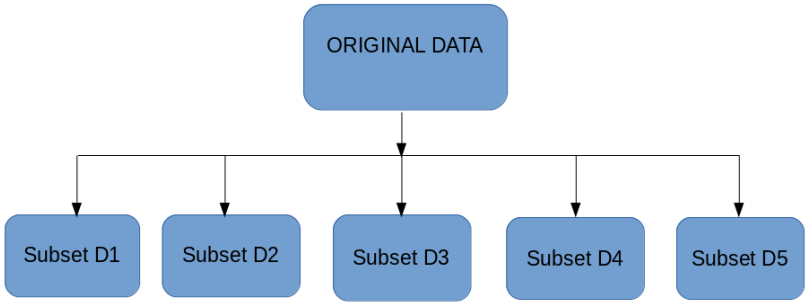
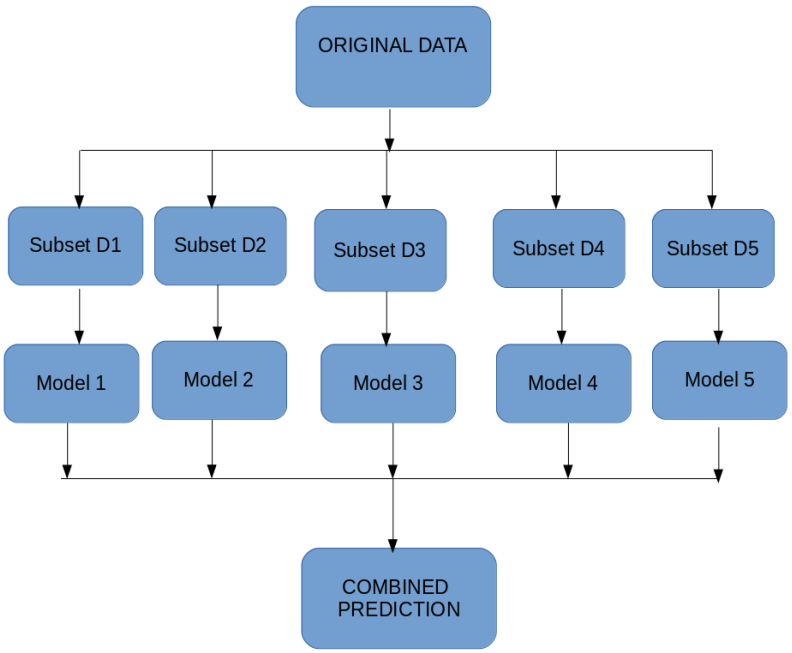


Fig.7 Bagging Regression

Multiple subsets are created from the original dataset, selecting observations with replacement. A base model (weak model) is created on each of these subsets. The models run in parallel and are independent of each other. The final predictions are determined by combining the predictions from all the models.



- **Algorithms based on Bagging:**

Bagging are two of the most commonly used techniques in machine learning. In this section, we will look at them in detail. Following are the algorithms we will be focusing on:

Bagging algorithms:

- Bagging meta-estimator
- Random forest

- **Bagging meta-estimator**

Bagging meta-estimator is an essembling algorithm that can be used for both classification (Bagging Classifier) and regression (Bagging Regressor) problems. It follows the typical bagging technique to make predictions. Following are the steps for the bagging meta-estimator algorithm:

1. Random subsets are created from the original dataset (Bootstrapping).
2. The subset of the dataset includes all features.
3. A user-specified base estimator is fitted on each of these smaller sets.
4. Predictions from each model are combined to get the final result.

- **Parameters used in the algorithms:**

- **base_estimator**:
    - It defines the base estimator to fit on random subsets of the dataset.
    - When nothing is specified, the base estimator is a decision tree.
- **n_estimators**:
    - It is the number of base estimators to be created.

  - The number of estimators should be carefully tuned as a large number would take a very long time to run, while a very small number might not provide the best results.
- **max_samples**:
  - This parameter controls the size of the subsets.
  - It is the maximum number of samples to train each base estimator.
- **max_features**:
  - Controls the number of features to draw from the whole dataset.
  - It defines the maximum number of features required to train each base estimator.
  -
- **n_jobs**:
  - The number of jobs to run in parallel.
  - Set this value equal to the cores in your system.
  - If -1, the number of jobs is set to the number of cores.

- **random_state**:
  - It specifies the method of random split. When random state value is same for two models, the random selection is same for both models.
  - This parameter is useful when you want to compare different models.

- # Best Fit Model & R2_Score: (Bagging Regressor)

- **FOR WIND GENERATION :**

```
#-------------------------------------------------------------------------------#
 #----------------Bagging Regression------------------#

 ### FOR WIND GENERATION ###
# Default: Tree Regressor
from sklearn.ensemble import BaggingRegressor

model_wind = BaggingRegressor(random_state=2020,oob_score=True,
                              max_features = X_wind_train.shape[1],
                              max_samples=X_wind_train.shape[0])


model_wind.fit( X_wind_train , y_wind_train )

print("Out of Bag Score = " + "{:.4f}".format(model_wind.oob_score_))

y_wind_pred = model_wind.predict(X_wind_test)

from sklearn.metrics import r2_score
print("Wind r2 score : ",r2_score(y_wind_test, y_wind_pred))
```

- **FOR SOLAR GENERATION :**

```
### FOR SOLAR GENERATION ###
# Default: Tree Regressor
from sklearn.ensemble import BaggingRegressor

model_slr = BaggingRegressor(random_state=2020,oob_score=True,
                             max_features = X_solar_train.shape[1],
                             max_samples=X_solar_train.shape[0])


model_slr.fit( X_solar_train , y_solar_train )

print("Out of Bag Score = " + "{:.4f}".format(model_slr.oob_score_))

y_solar_pred = model_slr.predict(X_solar_test)

from sklearn.metrics import r2_score
print("Solar r2 score : ",r2_score(y_solar_test, y_solar_pred))


#-------------------------------------------------------------------------------#
```

Fig 8-Solar and wind Generation

CHAPTER 7

Results

After applying different models, it has been found that the Bagging Regressor is best fit model for the dataset.

The result i.e. R2_score of the Bagging Regressor for Wind and Solar Generation is:

- **FOR WIND GENERATION :**

```
In [3]:
   ...:
   ...: from sklearn.ensemble import BaggingRegressor
   ...:
   ...: model_wind = BaggingRegressor(random_state=2020,oob_score=True,
   ...:                               max_features = X_wind_train.shape[1],
   ...:                               max_samples=X_wind_train.shape[0])
   ...:
   ...:
   ...: model_wind.fit( X_wind_train , y_wind_train )
   ...:
   ...: print("Out of Bag Score = " + "{:.4f}".format(model_wind.oob_score_))
   ...:
   ...: y_wind_pred = model_wind.predict(X_wind_test)
   ...:
   ...: from sklearn.metrics import r2_score
   ...: print("Wind r2 score",r2_score(y_wind_test, y_wind_pred))
Out of Bag Score = 0.9223
Wind r2 score 0.9542092105246168
G:\Anaconda3\lib\site-packages\sklearn\ensemble\_bagging.py:1066: UserWarning: Some inputs do
not have OOB scores. This probably means too few estimators were used to compute any reliable
oob estimates.
  warn("Some inputs do not have OOB scores. "
```

The R2_score for the Wind Energy Generation is 95.42 %.

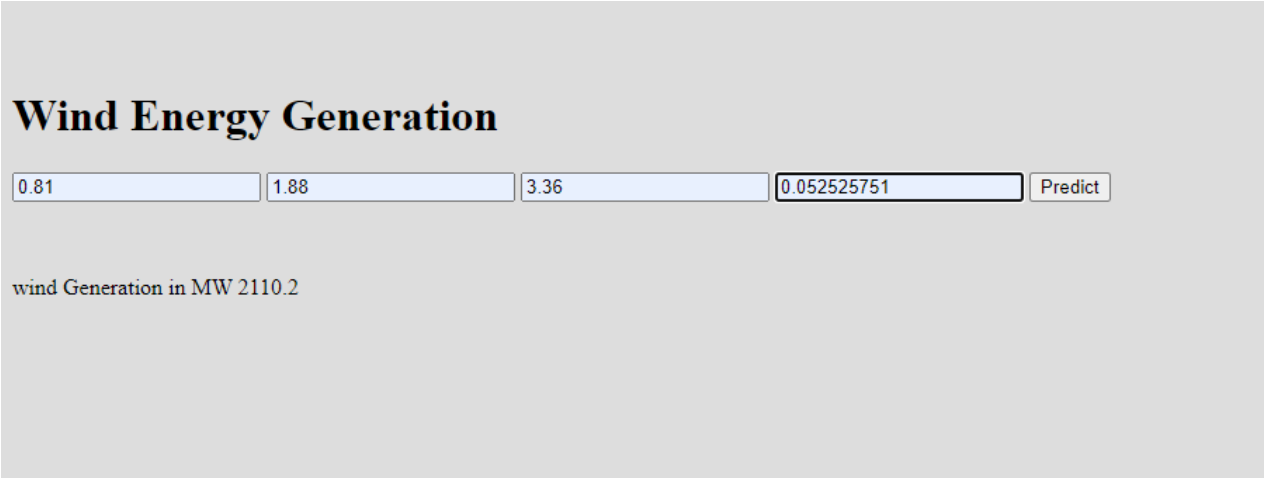
- **FOR SOLAR GENERATION :**

```
In [4]:
   ...:
   ...: from sklearn.ensemble import BaggingRegressor
   ...:
   ...: model_slr = BaggingRegressor(random_state=2020,oob_score=True,
   ...:                              max_features = X_solar_train.shape[1],
   ...:                              max_samples=X_solar_train.shape[0])
   ...:
   ...:
   ...: model_slr.fit( X_solar_train , y_solar_train )
   ...:
   ...: print("Out of Bag Score = " + "{:.4f}".format(model_slr.oob_score_))
   ...:
   ...: y_solar_pred = model_slr.predict(X_solar_test)
   ...:
   ...: from sklearn.metrics import r2_score
   ...: print("Solar r2 score",r2_score(y_solar_test, y_solar_pred))
Out of Bag Score = 0.9350
Solar r2 score 0.9574727165337372
G:\Anaconda3\lib\site-packages\sklearn\ensemble\_bagging.py:1066: UserWarning: Some inputs do
not have OOB scores. This probably means too few estimators were used to compute any reliable
oob estimates.
  warn("Some inputs do not have OOB scores. "
```

The R2_score for the Solar Energy Generation is 95.74 %.

- **PREDICTION OF SOLAR AND WIND GENERATION ON USER INTERFACE:**

- **<u>FOR WIND GENERATION</u> :**



- **<u>FOR SOLAR GENERATION</u> :**

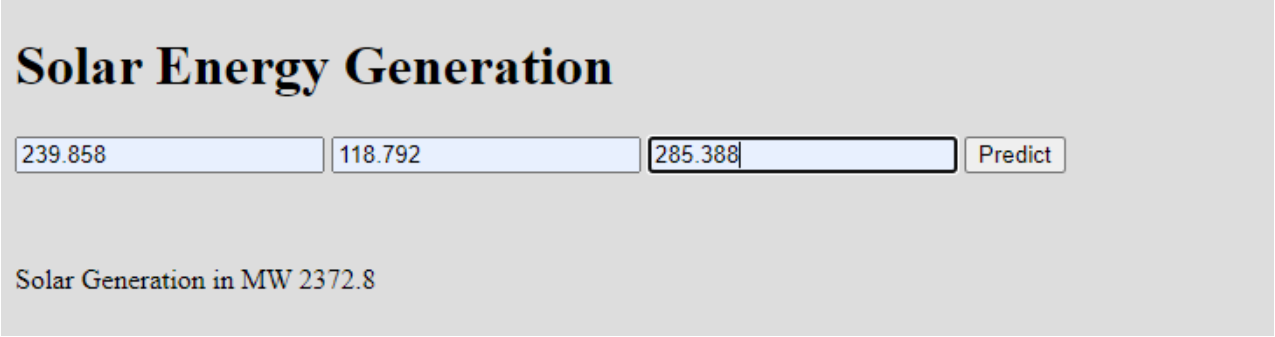


Fig 9-User Interface

FUTURE SCOPE

•There is wide-variety of future scope of this project. Non-renewable energy is limited resources that will eventually run out over the time frame and it also causes pollution in the environment. Thus, the thermal power plant is get shutdown in future scenario. So, all the power generation is shifted to renewable energy generation. In future, this project is also deployed to predict the generation of power from the hydro energy as well as tidal energy on the basis of hydro energy parameters and the tidal energy parameters.

CONCLUSION

Models	Wind r2_score	Solar r2_score
1.Linear Regression	0.9055	0.9552
2. KNN Regression	0.9112 (k=18)	0.9586 (k=19)
3. Ridge Regression	0.9054	0.9584
4. Elastic Net Regression	0.9054	0.9552
5. Bagging Regression	**0.9542**	**0.9574**

Table1-Model Conclusion

• Thus, Wind and solar are also promising renewable sources that have experienced a fast pace of growth in the recent years. An inherent feature of these resources is that the energy production capacity is not fully controllable or even predictable, So, to give the proper prediction about the renewable energy generation is really helpful for the power plant authorities.

• After applying the different models we have found that the Bagging Regressor is the best fit model for predicting the Renewable energy generation with overall r2_score of 95.42% for Wind Generation and 95.74% for Solar Generation.

• On deployment of model on Web, the results obtain are predicted correctly.

CHAPTER 8

REFERENCES

[1] A Survey of Machine Learning Models in Renewable Energy Predictions.Jung-Pin Lai 1, Yu-Ming Chang 1,2, Chieh-Huang Chen 1 and Ping-Feng Pai 1,3,*

[2] Machine Learning Techniques for Supporting Renewable Energy Generation and Integration:A SurveyKasun S. Perera1, Zeyar Aung2?, and Wei Lee Woon2

[3] Machine Learning-Based Approach to Predict Energy Consumption of Renewable and Nonrenewable Power Sources Prince Waqas Khan 1 , Yung-Cheol Byun 1,* , Sang-Joon Lee 1,*, Dong-Ho Kang 2, Jin-Young Kang 3 and Hae-Su Park 3

[4] Foley, A.M., Leahy, P.G., Marvuglia, A., McKeogh, E.J.: Current methods and advances in forecasting of wind power generation. Renewable Energy 37 (2012)

[5] Sharma, N., Sharma, P., Irwin, D., Shenoy, P.: Predicting solar generation from weather forecasts using machine learning. In: Proceedings of the 2011 IEEE International Conference on Smart Grid Communications (SmartGridComm). (2011)
528

[6] Zendehboudi, A.; Baseer, M.A.; Saidur, R. Application of support vector machine models for forecasting solar and wind energy resources: A review. J. Clean. Prod. **2**018, 199, 272–285.

[7] Feng, C.; Zhang, J. Hourly-similarity based solar forecasting using multi-model machine learning blending. arXiv 2018, arXiv:1803.03623.

[8] Torres-Barrán, A.; Alonso, Á.; Dorronsoro, J.R. Regression tree ensembles for wind energy and solar radiation prediction. Neurocomputing 2019, 326, 151–160.

[9] Wang, H.; Lei, Z.; Zhang, X.; Zhou, B.; Peng, J. A review of deep learning for renewable energy forecasting.Energy Convers. Manag. 2019, 198, 111799.

[10] Pérez-Ortiz, M.; Jiménez-Fernández, S.; Gutiérrez, P.A.; Alexandre, E.; Hervás-Martínez, C.; Salcedo-Sanz, S. A review of classification problems and algorithms in renewable energy applications. Energies 2016, 9, 607