

Predicting daily incoming Solar Energy output using weather data

Project report submitted in partial fulfillment of the requirement for the
degree of Bachelor of Technology
in
Computer Science and Engineering/Information Technology

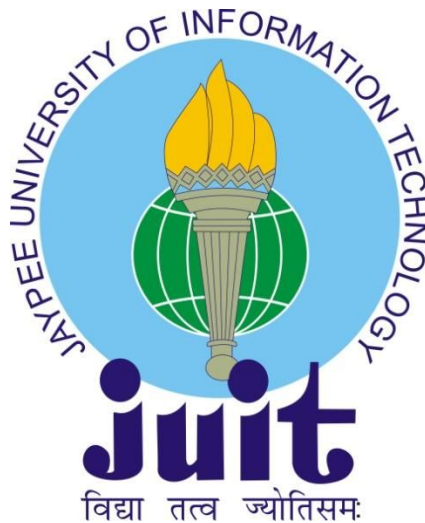
By

Sudhanshu Sharma(151341)

Under the supervision of

Dr. Ravindara Bhatt

to



Department of Computer Science & Engineering and Information
Technology

**Jaypee University of Information Technology Waknaghat,
Solan-173234, Himachal Pradesh**

CERTIFICATE

I hereby declare that the work presented in this report entitled “**Prediction of incoming solar energy using weather data**” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2018 to December 2018 under the supervision of **Ravindara Bhatt**, Assistant Professor, Department of Computer Science and IT. The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Sudhanshu Sharma, 151341

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Ravindara Bhatt

Assistant Professor

Department of Computer Science and IT

Dated:

ACKNOWLEDGEMENT

We would like to articulate our deep gratitude **Dr Ravindra Bhatt JUIT** for enabling us to gain an immensely enriching professional experience. We would like to thank him for his endless support and relentless supervision without which this could not have been possible.

We would also like to convey our sincerest gratitude and indebtedness to all other **Faculty member, Course Coordinator** and **Staff** of Department of Computer Science JUIT, who showed their great efforts and guidance at required times without which it would have been very difficult to carry out our project.

ABSTRACT

This project 'Prediction of Solar energy using weather data' is built using machine learning algorithms. Solar energy is most useful renewable source of energy so accurately predicting solar energy is required as there is a need to make up deficits from traditional fuel plants to match the total power generation with the instantaneous power consumption. Project main focus is based on prediction of Solar energy output recorded by solar photovoltaic panels using weather metrics like Temperature, Cloud coverage, humidity, visibility etc on a particular day. Type of problem of this project is regression based machine learning problem. Various famous machine learning algorithms like Random Forest, Lasso and Ridge and K-nearest neighbours are used in order to accurately forecast solar PV output. Dataset was gathered from National Oceanic and Atmospheric Administration's (NOAA) Global Ensemble Forecast System (GEFS) which generally forecasts weathers.

Every machine learning algorithm is evaluated based on error evaluation metrics like mean squared error, mean absolute error and R-squared score. Many visualizations plots are also depicted in order to have better understanding of model. Feature selection is done on the dataset in order to select the best predictor features from a range of features. At last results and model comparison is done in order to evaluate which model among the three is best fitted for Solar PV output forecast.

TABLE OF CONTENTS

Chapters	Page
No.	
1. Introduction	1
1.1 Introduction	1-4
1.2 Problem Statement	4-5
1.3 Objectives	5
1.4 Methodology	5-6
1.4.1 Lasso and Ridge	6
1.4.2 K-nearest neighbours	6
1.4.3 Random Forest	6-7
1.5 Organisation	7-8
2. Literature Survey	9-10
3. System Development	11
3.1 Data Gathering	11-12
3.2 Understanding data	12-17
3.3 Data pre-processing	17-19
3.4 Feature selection	20-21
3.5 System design flow diagram	21-22
3.6 Evaluation Metrics	22-26
3.7 Operating environment	26-28
3.8 Design issues	28-29
4. Performance Analysis	30
4.1 Lasso and Ridge	30
4.1.1 Ridge regression	30-33
4.1.2 Lasso regression	33-36
4.2 K-nearest neighbours	36-41

4.3 Random Forest	41-45
4.4 Results and Model comparison	45
5. Conclusion	46
5.1 Conclusion	46
5.2 Future Scope	46
References	47

LIST OF FIGURES

- (i) - Current, voltage and power curve of solar power
- (ii) - Ensemble Techniques
- (iii) - Dataset weather_data.csv
- (iv) - Correlation graph
- (v) - Correlation matrix
- (vi) - Scatter plots of Solar energy vs features
- (vii) - Descriptive analysis of weather data
- (viii) - System design flow diagram
- (ix) - 7 fold cross-validation process
- (x) - Lasso coefficients vs alpha plot
- (xi) - Nearest neighbourhoods for each point of a training dataset
- (xii) - RMSE value vs k value variation plot
- (xiii) - Random Forest model structure
- (xiv) - Variable importances bar plot

Chapter-1

INTRODUCTION

1.1 Introduction

Climatic and energy crisis has led to the development of sustainable energy resources. Solar energy is identified as the most promising, clean and rich candidate for volume power production. In recent years, there has been a large increase in the use of *solar energy*. Solar technology has also evolved during last 10 years. The solar industry is developing at quite a fast rate which in turn has decreased the production costs, increased the availability and also, now people are more aware about the infinite benefits of using renewable sources of energy as opposed to traditional fossil fuels. Photovoltaic solar systems are being used widely across the world and they are one of the most propitious renewable energy sources. Renewable energy sources provide a significant advantage especially solar energy over fossil fuels because they are less expensive, less volatile, cost effective, available in abundance, cause no health effects, no waste by products, etc. but the main problem is accurate prediction of these energy resources, which is quite difficult as solar radiation, which is an important parameter in predicting solar energy, depends a lot on the atmospheric conditions i.e. sun's path, the scattering process, cloud cover, the characteristics of solar energy plant and many other parameters, and therefore the solar power output cannot be fully controlled or planned in advance. Accurately predicting solar energy is required as there is a need to make up deficits from traditional fuel plants to match the total power generation with the instantaneous power consumption and to securely fuse PV systems into smart grid, as correct predictions are critical element of energy management systems. If accurate prediction is flawed or inadequate, any fluctuation in solar energy capacity may have compelling effects on daily the operations, profitability, physical health of the entire grid and may also negatively affect the quality of life of the energy consumers. We need to maintain a mix of traditional power sources and renewable power sources for a proper demand-supply cycle. As the use of solar energy is continuously increasing, old traditional methods to extract the energy are not sufficient to handle large demand of consumers, hence, more efficient and fast methods are required, which takes in consideration the weather and temporal parameters, to predict

the variable output of PV systems to satisfy the needs. Solar forecasting commonly outputs solar irradiance or PV.

Solar irradiance or insolation is the measure of solar radiation energy received on a given surface and recorded during a given period of time. It's measured in megajoules per meter square (MJ/m²). Insolation is also known as solar irradiation. It is one of the important parameter in predicting solar energy output. When radiation strikes the planet, some of the radiation is absorbed and remaining is reflected. In case of man made system, the absorbed energy may be converted to other form of energy like electrical energy. The amount of radiation absorbed or reflected depends upon the objects reflectivity. Solar radiation varies non linearly due to atmospheric events like cloudy weather, rain, humidity, etc. Therefore it's estimation is a challenging task and quite an interesting issue in solar energy field. Numerically solar insolation can be calculated as follows:

$$I(\text{Solar irradiation}) = S * \cos Z - (i)$$

Here, S is the solar constant and it's roughly equal to 1000 W/m².

$$Z(\text{Zenith angle}) = \cos^{-1}(\sin X * \sin Y + \cos X * \cos Y * \cos H) - (ii)$$

X is latitude and Y is solar declination.

$$H(\text{Hour angle}) = 15^\circ * (\text{time} - 12) - (iii)$$

Time equals the hour of day from midnight. For example, noon equals 12 and 4 p.m. equals 16.

The maximum power output from PV cell can be calculated as follows:

$$Pr = \eta SI[1 - 0.05(t_0 - 25)] - (iv)$$

where η represents the conversion efficiency (%) of the solar cell array; S is the array area (m²); I is the solar irradiation (kW/m²); and t_0 is the outside air temperature (°C). Recording maximum power point (MPPT) is an important factor to improve efficiency of PV systems. In MPPT method, voltage V_r and the current I_r is calculated automatically, where the PV array operated efficiently to obtain maximum power output P_r under a given temperature and irradiance, as expressed in *fig (i)*.

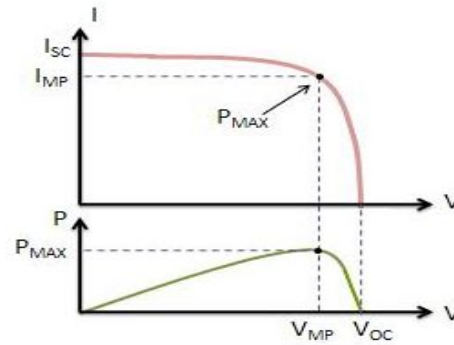


fig (i) – Current, voltage and power curves of solar panel

PV generation forecast models can be classified into four methods, i.e., statistical approach, artificial intelligence method, physical approach, and hybrid method. In statistical methods historical data is formulated to predict solar time series. AI approaches use advanced machine learning and AI algorithms to forecast solar energy. In physical models satellite images or numerical weather prediction (NWP) models are used to predict solar irradiance and PV generation. Finally, hybrid methods use combination of the above three approaches to predict solar output. Generally, different forecasting approaches are used to calculate incoming solar radiation depending on different scales of prediction horizons to meet the requirements of decision making process. For instance, short term prediction of upto 48-72 hours is important for different decision making problems in the electricity market and power system operation, including unit commitment, storage, automatic generation control, etc. Therefore, most studies focus on short-term prediction models.

In our project, we will be using different Machine Learning Regression techniques like Random forest regression, K-nearest neighbours (KNN) regression and Lasso Ridge regression to derive prediction models for the incoming solar energy using multiple forecast metrics. It is known that the main causes of difference between actual and predicted values in any machine learning algorithm we use are noise, error and bias so we are going to use techniques which are able to remove above three factors effectively. We will also compare and contrast these techniques to find out which algorithm is most suited for this particular task. Machine learning algorithms are nothing new, important thing here is that the computational capabilities of machines has improved quite significantly in recent years. Now, large amount of datasets can be processed in shorter duration of time

to fetch useful information. We can leverage this fact to our advantage and efficiently predict the future solar output with the help of historical solar data gathered from forecasting systems. Table below give us a brief idea about solar energy generation in India.

Table 1. Statistics of Global Solar Radiation in India [9]

Particulars	Value (MJ/m²/year)	Place
Average Global Solar Radiation	7000	India
	7200	Peninsular India
Annual highest receiving of Global Solar Radiation	8000	Rann of Kutch
Daily average of Global Solar Radiation	22	Rann of Kutch
	16.5	Kashmir Valley
	15	North East India
Daily average of GSR during month January	<5	Kashmir Valley
	≅ 20	Deccan Plateau
	<15	North Indian Plains
Daily average of GSR during month of April	>20	India
Maximum daily average GSR during month April	>25	Rajasthan & Saurashtra

1.2 Problem Statement

Based on our introduction, the problem statement can be encapsulated with the following points:

- To Predict total daily incoming solar radiation at 98 Oklahoma Mesonet sites using various machine learning algorithms in conjunction with the provided historical daily basis weather data from forecasting system.

- Discover which machine learning algorithms and techniques provide best short term forecasting of solar PV energy output.

1.3 Objectives

Main goal of this project research is to benchmark different forecasting techniques to predict the solar PV energy output by using relevant statistical and machine learning techniques and algorithms to learn the relationship between weather conditions and solar PV panel energy outputs later performing some feature engineering aside of the forecast features to increase the prediction accuracy.

Five machine learning techniques are benchmarked from the existing solar PV data systems installations which will be required feature engineering methodologies to increase the overall prediction accuracy.

1.4 Methodology

In our project we are going to use several machine learning techniques: random forests, lasso and ridge and k-nearest neighbours. We are going to train the models for solar intensity prediction which are based on above mentioned algorithms using processed historical weather data and then compare the results with each other using some error metrics to determine which algorithm has best accuracy among these and is best suitable for short-term solar intensity or PV prediction.

Brief about the steps that we are going to follow are given as follows:

- First we are going to gather data from online source.
- Then we are going to process the data according to the need of our machine learning algorithms.
- Third step is to find out correlation between different forecast metrics and the solar intensity to determine which variable(s) affect solar irradiation the most. In a nutshell, we are going to predict the selectors using feature engineering.

- Lastly, we are going to train the models on a certain percentages of train and test split (which will be different for every algorithm) of above processed dataset and then analyse the results to find the outcome.

We are going to use python and various inbuilt as well as external tools to implement above mentioned steps.

1.4.1 Lasso and ridge

Instead of using simple linear regression as our first algorithm we are going to use a better alternative known as lasso and ridge. This technique reduces the coefficient estimates towards zero, and hence, reducing the variance, thus giving more accurate results. Both algorithms use different functions for minimization, which we are going to explain in detail in chapter 4. Both of these algorithms are most commonly used to approximate the result of those equations with no unique solution where we chose the best and most accurate fit.

1.4.2 K-nearest neighbours

Next, we use KNN(k-nearest neighbours) which is also a regression and classification technique. In KNN the entire dataset is the model. We use data structures like k-d trees for look up and matching patterns used for predictions. Since, the entire data represents the model we may want to think about consistency problem that may arise due to this fact. So, it is important to continuously update it as new data is available and remove useless data. Prediction using KNN can be made for a new instance, let's say x , by traversing the whole dataset and finding the k most similar occurrences in the dataset.

1.4.4 Random Forest

We also use another ensemble technique known as random forest. It is an ensemble bagging technique in which we make many independent variables/models/learners and combine them using model averaging methods. In this method we divide the dataset into

small bootstrapped datasets so that each model/learner we are going to build gets different data to work on. This method reduces variance for reducing error.

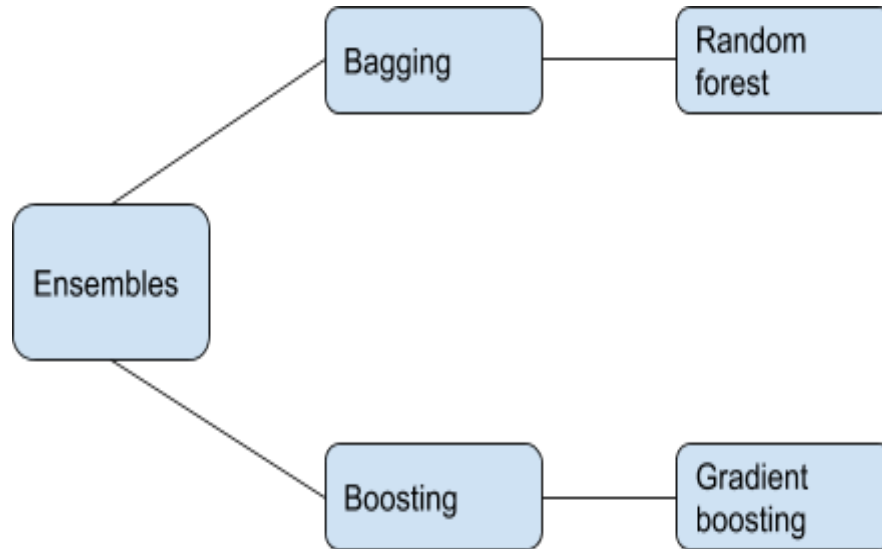


fig (ii) – Ensemble techniques.

1.5 Organisation

The project report is constructed as follows:

Chapter 1 includes brief introduction about the project topic, its importance in solar industry, solar energy, solar insolation, different methods to forecast solar irradiation or PV. It also includes project statement and objectives along with methodology used to build the project.

Chapter 2 is literature survey. It contains related information about our project topic, solar energy, sun earth astronomy, factors affecting solar energy prediction, findings for solar insolation or PV forecasting methods developed in recent years and their comparison, gathered from various literatures, books and internet websites.

In chapter 3, we discuss about the system development of our project i.e. system architecture, design, analysis, flow diagram, etc.

Chapter 4 presents various algorithms used in our project to predict the incoming solar energy using weather data.

Test plan, data set and metrics used are presented in chapter 5.

Subsequently, chapter 6 discusses and evaluates results obtained in chapter 3,4 and 5 and it also presents analysis of the system.

Finally, chapter 7 concludes the project discussing about the project success, its future scope and possible applications.

Chapter-2

LITERATURE SURVEY

In this chapter, articles related to solar forecasting are presented. Further, important findings for machine learning techniques for solar forecasting is also mentioned. The literature survey for this report is mainly based on the Irvin et al. [1], Inman et al. [2], Hong et al. [6] and Saberian et al. [4]. These articles provide a broad understanding of solar panel and solar forecasting methods.

Photovoltaic systems have been widely deployed in recent times to meet the increased electricity demand as a renewable energy source. A key target of smart grid initiative is to notably increase the fraction of energy contributed by renewable sources but due to the variable nature of solar output its integration in grids is quite difficult, and hence a lot of research is being done in the field of solar forecasting methods to accurately calculate short-term as well as long term solar output.

Hong et al. [6] provide a brief of various forecasting techniques used for solar energy forecasting. The study stresses that many various time series and ML techniques have been used in various energy situations. The study is based on a competition for energy forecasting and evaluates the methods used in the competition (Global Energy Forecasting Competition 2014). The main findings are that for load demand prediction, both machine learning and traditional time series methods have been commonly used. Similarly, for electricity prices, a broad variety of time series and ML techniques have been used. The article also provides a brief of methods used in the competition for windmills and solar PV panels power forecasting. In counterpoint to load demand and electricity prices, the span of examined forecasting methods for windmills and solar PV panels power output is smaller. The use of machine learning techniques are limited while traditional time series have been used vastly. For solar PV panels, the Random Forest algorithm and the Support Vector Machine (SVM) algorithm were used.

Hizam et al. [4] used artificial neural networks to predict solar power output. The authors used two neural network structures, namely, general regression neural network (GRNN) feedforward backpropagation (FFBP), to model a photovoltaic panel output power and

estimate the generated power. They concluded that, both of these neural networks show good modelling performance; however, FFBP provide a better performance compared to GRNN.

Inman et al. [2] present an considerable survey of theory for RES forecasting. Theories from different fields are presented, such as on how to model irradiance, air masses, and clearness indices. Theoretical background of commonly used machine learning techniques and solar forecasting methods are also given. Furthermore, theories on irradiance and satellite images are demonstrated to give an understanding of how they can be utilized. The work serves as steering for both theory and methodologies from a physics, statistical and ML standpoint.

Chapter-3

SYSTEM DEVELOPMENT

This chapter includes various aspects of system design document such as operating environment, system architecture, processing logic, input formats, etc.

3.1 Data Gathering

The historical dataset we used here was taken from the National Oceanic and Atmospheric Administration's (NOAA) Global Ensemble Forecast System (GEFS). The time duration of the dataset for weather attributes (predictor variables) and solar energy (target variable) starts from Feb 1, 2016 ends on October 31, 2017 in total of 637 entries and 10 variables.

Table 2. Weather dataset features and their units

Weather Features	Units
Date	mm/dd/yy
Cloud coverage	% range
Visibility	miles
Temperature	°C
Dew point	°C
Relative humidity	%
Wind speed	Mph
Station pressure	inchHg
Altimeter	inchHg
Solar energy(PV panel)	kWh

	Date	Cloud coverage	Visibility	Temperature	Dew point	Relative humidity	Wind speed	Station pressure	Altimeter	Solar energy
0	02/01/2016	0.10	9.45	3.11	0.32	79.46	4.70	29.23	30.02	20256
1	02/02/2016	0.80	3.94	6.99	6.22	93.60	13.29	28.91	29.70	1761
2	02/03/2016	0.87	8.70	1.62	0.02	85.00	16.73	29.03	29.82	2775
3	02/04/2016	0.37	10.00	-2.47	-5.89	74.52	9.46	29.46	30.26	28695
4	02/05/2016	0.52	9.21	-2.00	-4.15	82.03	5.92	29.55	30.35	9517
5	02/06/2016	0.13	8.12	0.91	-1.62	81.03	5.48	29.44	30.24	26973
6	02/07/2016	0.21	10.00	4.24	0.54	73.80	12.71	29.09	29.88	22365
7	02/08/2016	0.87	7.84	-3.33	-5.05	83.53	14.46	28.96	29.75	4995
8	02/09/2016	1.00	6.01	-7.80	-10.39	78.34	17.27	29.10	29.89	6641
9	02/10/2016	0.25	10.00	-9.72	-13.98	67.70	10.30	29.29	30.08	20758
10	02/11/2016	0.34	10.00	-7.64	-13.73	60.36	3.85	29.48	30.28	30910
11	02/12/2016	0.62	9.21	-5.78	-10.00	69.76	12.42	29.46	30.26	8105
12	2/13/2016	0.08	10.00	-10.12	-17.77	52.32	7.44	29.73	30.54	30358
13	2/14/2016	1.00	6.37	-7.29	-9.99	78.47	13.79	29.38	30.18	5814
14	2/15/2016	0.94	3.86	-2.42	-2.68	94.02	6.43	29.09	29.88	7570

fig (iii) – Dataset – weather_data.csv

3.2 Understanding the Data

It is often very important to understand and explore what data is telling us or to understand kind of influence does each feature have with the target variable. Datatypes of each feature, kind of correlation they show with target, presence of outliers, standard deviation, mean, median and range values are some important metrics to evaluation scheme. Certain plots such as scatter plot help us in better visualization between predictors and target.

We ran describe and info functions of python in order to evaluate descriptive analysis of our weather data. We found that most of the variables in dataset were float type and did not contain any null or missing values.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 637 entries, 0 to 636
Data columns (total 9 columns):
Cloud coverage      637 non-null float64
Visibility          637 non-null float64
Temperature         637 non-null float64
Dew point          637 non-null float64
Relative humidity   637 non-null float64
Wind speed         637 non-null float64
Station pressure    637 non-null float64
Altimeter          637 non-null float64
Solar energy       637 non-null int64
dtypes: float64(8), int64(1)
memory usage: 44.8 KB
```

We did correlation analysis using Python pandas library between weather parameters for quantitative analysis and plotted heat map to get better visualization of correlation as shown in Fig().

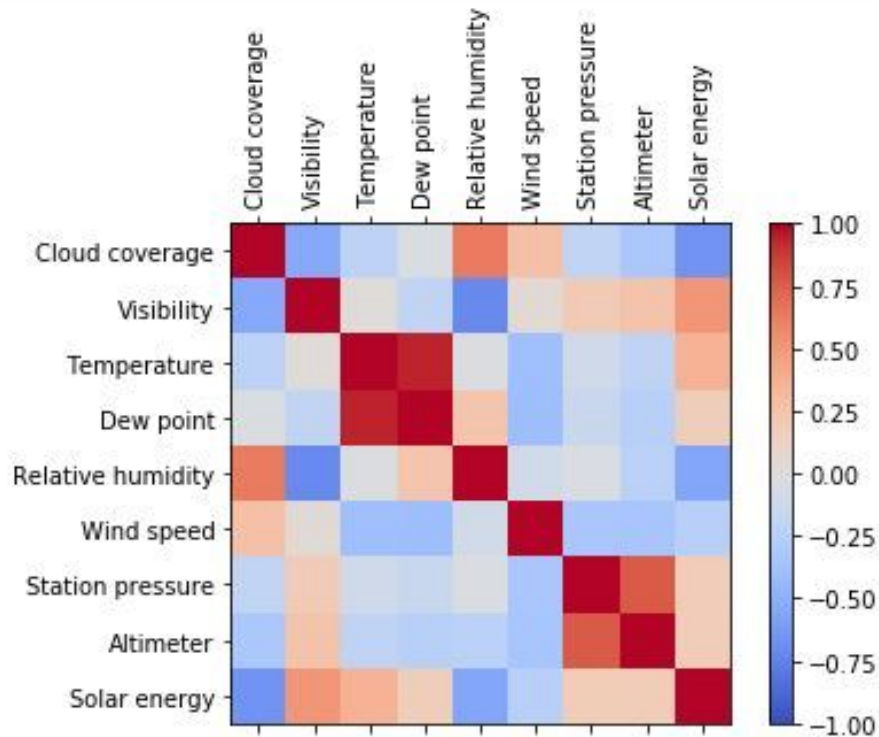


fig (iv) – Correlation graph with hot color coding

Here in the above correlation graph is plotted using color coding. Here the color is gray when there is no correlation between two variables (correlation is 0 or near 0), strong positive correlation between variables is dark red color coded and strong negative

correlation is depicted by blue color code. In contrast to above Fig() below is correlation matrix between data variables.

	Cloud coverage	Visibility	Temperature	Dew point	Relative humidity	Wind speed	Station pressure	Altimeter	Solar energy
Cloud coverage	1.000000	-0.547053	-0.244460	-0.064063	0.612688	0.293058	0.096449	-0.325371	-0.687961
Visibility	-0.547053	1.000000	0.156268	-0.008976	-0.547587	-0.015784	0.006682	0.202740	0.468652
Temperature	-0.244460	0.156268	1.000000	0.951895	0.013574	-0.342758	0.160661	-0.265984	0.399998
Dew point	-0.064063	-0.008976	0.951895	1.000000	0.272376	-0.340463	0.141527	-0.325546	0.207741
Relative humidity	0.612688	-0.547587	0.013574	0.272376	1.000000	-0.054044	0.456894	-0.249580	-0.472128
Wind speed	0.293058	-0.015784	-0.342758	-0.340463	-0.054044	1.000000	-0.110455	-0.369035	-0.259589
Station pressure	0.096449	0.006682	0.160661	0.141527	0.456894	-0.110455	1.000000	-0.012116	0.169627
Altimeter	-0.325371	0.202740	-0.265984	-0.325546	-0.249580	-0.369035	-0.012116	1.000000	0.178743
Solar energy	-0.687961	0.468652	0.399998	0.207741	-0.472128	-0.259589	0.169627	0.178743	1.000000

fig (v) – Correlation matrix

We concluded from above that the variables like Cloud coverage and Relative humidity showed strong negative correlation with the target variable i.e. Solar energy while Visibility and temperature showed sufficiently positive correlation with Solar energy.

Further in order to get better understanding we used scatter plots between variables and target to observe for any linearity or multicollinearity. Below we present some scatter plots regarding the same.

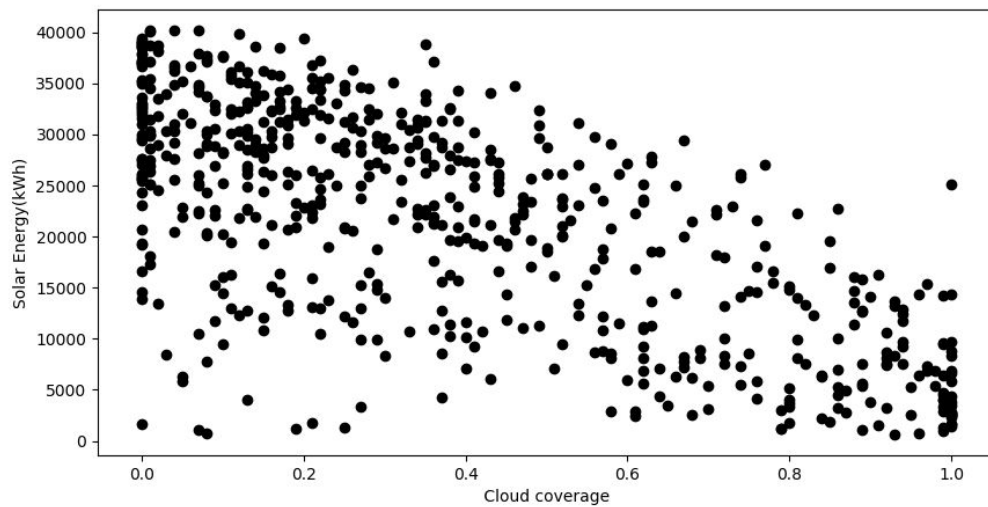


fig (vi – a) – Solar energy showed strong negative correlation with Cloud coverage

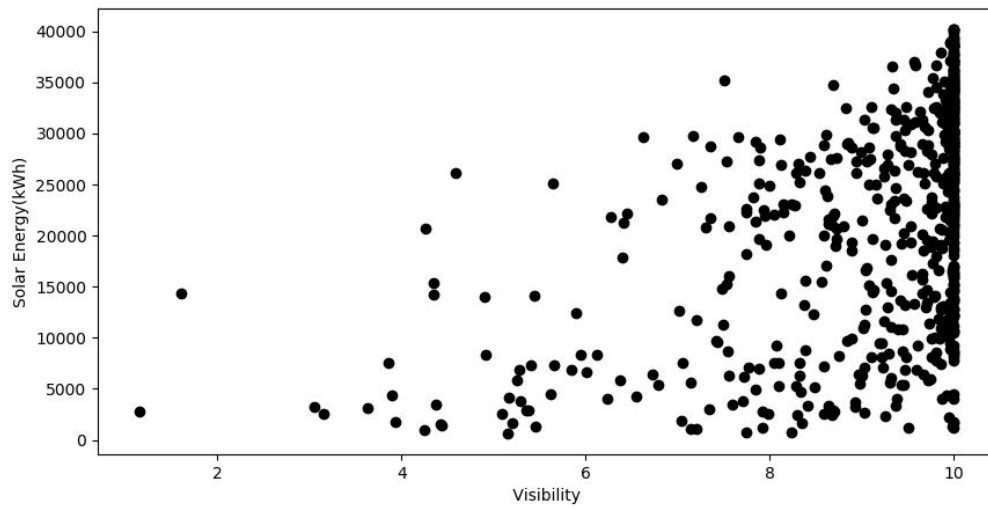


fig (vi – b) – Solar energy showed strong positive correlation with V isibility

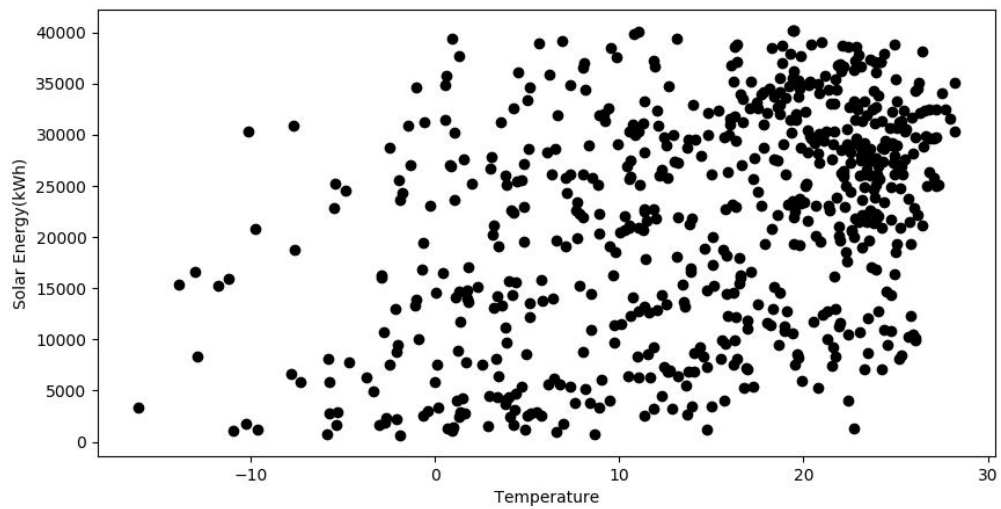


fig (vi – c) Solar energy showed some positive correlation with T emperature at high temperatures

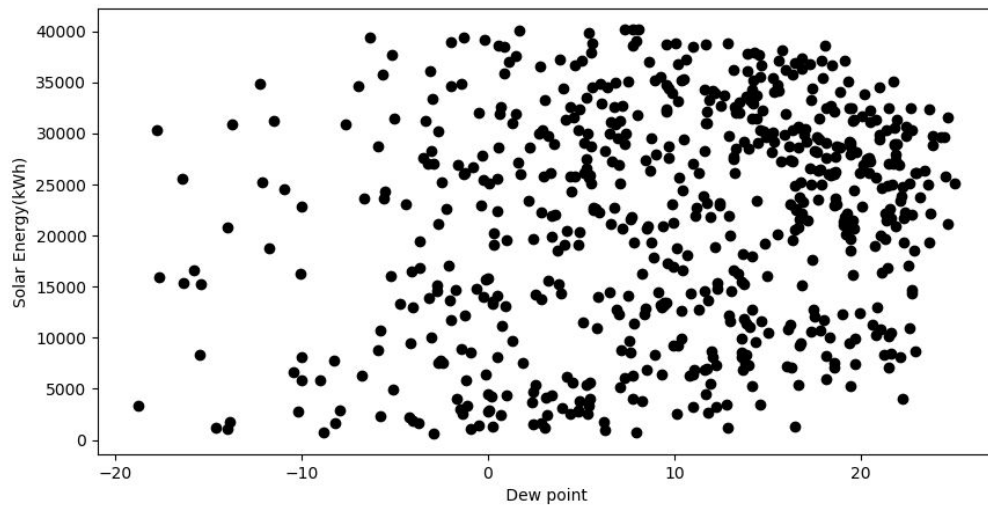


fig (vi - d) – Solar energy showed little correlation with Dew point at high dew points

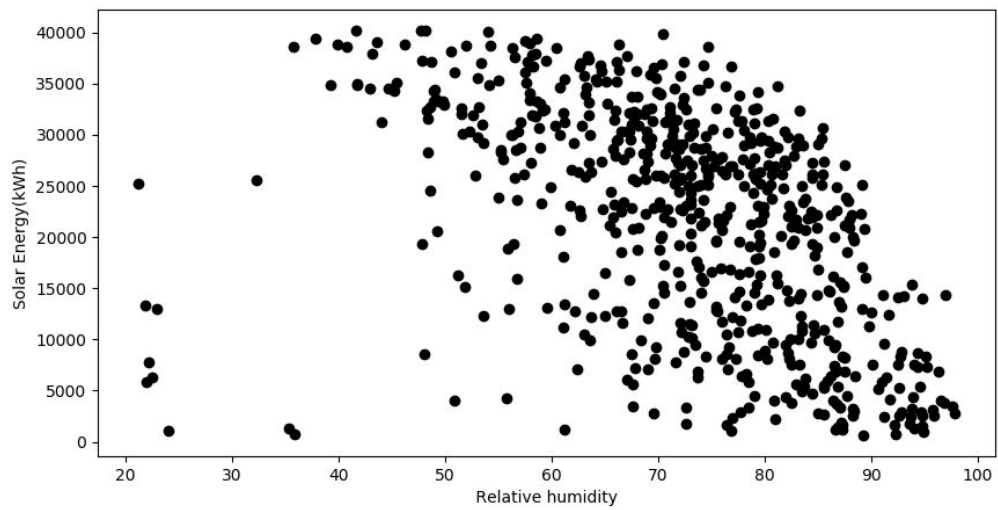


fig (vi - e) – Solar energy showed strong negative correlation with Relative humidity

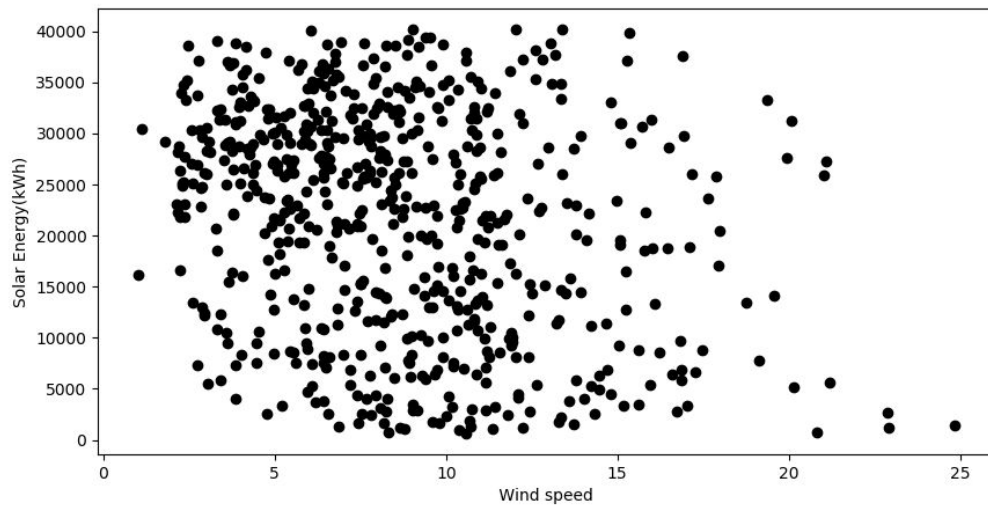


fig (vi – f) – Solar energy showed little correlation with Wind speed

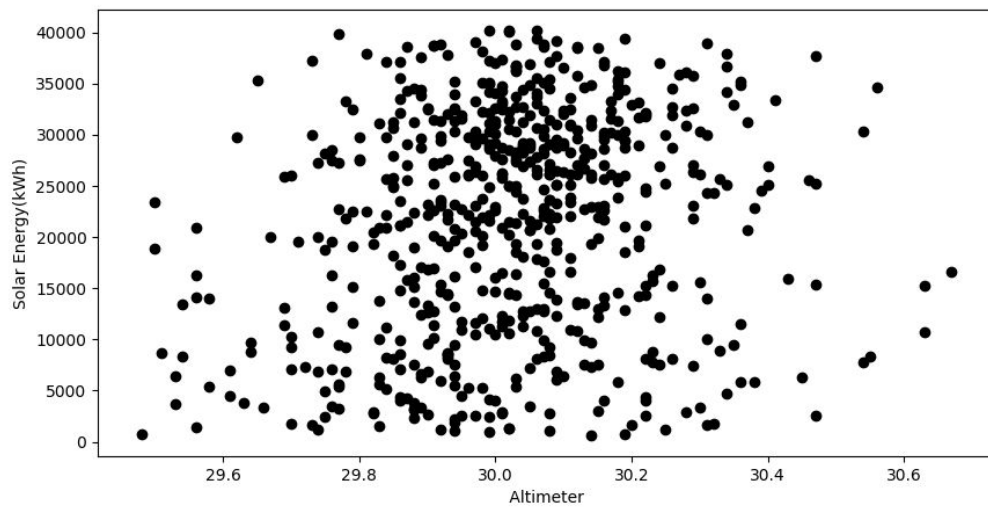


fig (vi – g) – Solar energy showed little correlation with Altimeter

3.3 Data pre-processing

Adaptation of the raw data according to our need is important before implementing the regression model. Since, raw data is most of the time inconsistent or incomplete or lacking in certain behaviour or lacking in attributes or may contain noises. So we need to

remove all these abnormalities and convert the dataset into something which can be used by the machine learning algorithms. So we processed data obtained from online sources to obtain useful numerical weather metrics, related to solar intensity output, on daily basis which can be used to train our models. The weather data which we downloaded was in xlsx format. So we had to convert the xlsx file format to csv format which is usual format used to train machine learning models. Further sometimes data contain various inconsistencies such as null values or missing values, noisy data which model cannot interpret and value dominances of a variable over another which can cause model's inconsistency to predict accurately.

In our data we observed that some data variables range over different values so we applied various imports to scale data values. We used two approaches from python sklearn.preprocessing module's MinMaxScaler and StandardScaler to scale features over similar value ranges before any training in order to avoid overfitting and underfitting problems.

3.3.1 Data scaling using MinMaxScaler approach

MinMaxScaler is the python sklearn preprocessing module which scales feature values over a given range by individually estimating and scaling the feature values to a given range i.e. between 0 and 1.

Transformation is given as:

```
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min
```

Transformation is calculated as:

```
X_scaled = scale * X + min - X.min(axis=0) * scale
where scale = (max - min) / (X.max(axis=0) - X.min(axis=0))
```

here, min and max are feature range.

Below is our dataset feature values snippet after scaling:

0	1	2	3	4	5	6	7	
0	0.17	0.995480	0.932414	0.882434	0.645561	0.279901	0.962876	0.336134

1	0.01	1.000000	0.613472	0.542877	0.623238	0.264516	0.982613	0.689076
2	0.94	0.306215	0.308318	0.369756	0.950000	0.267990	0.963346	0.336134
3	0.00	1.000000	0.638788	0.472337	0.413185	0.465509	0.977444	0.596639
4	0.87	0.755932	0.287749	0.315122	0.813055	0.666501	0.957237	0.226891
5	0.00	0.983051	0.769892	0.702167	0.602089	0.271464	0.978853	0.621849

3.3.2 Data scaling using StandardScaler

StandardScaler is the python sklearn preprocessing module which standardizes features over mean and standard deviation of 0 and 1 independently over every feature by running transform method over training data samples.

Standard score calculation of a sample:

$$\text{score} = (\text{sample} - \text{sample_mean}) / \text{sample_standard_deviation}$$

Standardized scaling is often considered as a common practice in building machine learning models so that data look more standard normally distributed and with mean and std to unit variance.

Below is our dataset feature values snippet after standardization:

```
[-0.90508978  0.22023946 -1.17054973 ... -0.96374205  0.23766664
-0.01565998]
[ 1.33155139 -3.67363464 -0.76114199 ...  1.14191663  0.1183269
-1.72268193]
[ 1.5552155 -0.3097797 -1.32777075 ..... 1.98516062  0.1630793
-1.0825487 ]
.....
[ 1.61911954  0.60892018 -0.88776294 ....  1.22280922  0.18172614
-0.81582652]
```

[1.9386397 0.37571175 -1.13678415 ... 0.25454941 0.20783171

-0.44241547]

[0.62860702 0.60892018 -0.97534243 ... -0.76028492 0.17799677

-0.86917096]]

3.4 Feature Selection

There are various factors which affect the result of machine learning on a given work. The first and foremost is the quality and representation of the dataset. Feature selection is the process of identifying and removing redundant data as much as possible then selection from various features to features which we want to use in building our predictive regression model. The performance and effectiveness of machine learning model may be affected by useless information present in the dataset. The model results after doing feature selection becomes more accurate, easily understandable and improves model performance. Recent research have also shown that useless variables affect the performance of some algorithms for e.g. in simple nearest neighbour algorithm, the sample complexity increases exponentially with the number of useless attributes. So in this part we identify the variables or attributes which are strongly related to the solar intensity output. In data preprocessing phase we ran correlation analysis between the various weather metrics and solar energy output to discard the variables which are not useful. Some machine learning algorithms like Lasso Ridge, Random forests also helps in evaluating variable importances.

3.4.1 Using Amount of variation approach

We can drop any feature if variation of that feature is very low or zero variance (unary) since if any variable represent low variability scales it will not have that much effect in predictive analysis. Either we can standardize variables or use standard deviation to account for variables with low a scales. For our cause we ran descriptive analysis to observe and evaluate these metrics below snippet is regarding the same. We can observe

that the variable Altimeter has very low standard deviation value of 0.187608 hence we were able to discard it but instead we used Standardized scaling over whole data because of high variance shown by Solar Energy parameter.

	Cloud coverage	Visibility	Temperature	Dew point	Relative humidity	Wind speed	Station pressure	Altimeter	Solar energy
count	637.000000	637.000000	637.000000	637.000000	637.000000	637.000000	637.000000	637.000000	637.000000
mean	0.383265	9.138352	14.203422	9.623626	72.416876	8.631570	28.592716	30.022936	21521.733124
std	0.313215	1.416155	9.484552	9.334436	13.711642	4.082689	2.683527	0.187608	10843.988912
min	0.000000	1.150000	-16.060000	-18.720000	21.250000	1.030000	8.590000	29.480000	580.000000
25%	0.120000	8.840000	7.360000	3.220000	65.310000	5.670000	29.060000	29.910000	12268.000000
50%	0.320000	9.850000	16.260000	10.800000	73.690000	8.170000	29.200000	30.030000	23094.000000
75%	0.620000	10.000000	22.340000	17.330000	82.330000	10.910000	29.310000	30.130000	30307.000000
max	1.000000	10.000000	28.180000	25.020000	97.850000	24.830000	29.870000	30.670000	40245.000000

fig (vii) – Descriptive analysis of weather data

3.4.2 Using Correlation analysis approach

In order to summarize features importance we also used correlation analysis approach. If correlation is found between two variables it means that when there is a systematic change in one variable, there is also a systematic change in the other; the variables alter together over a certain period of time. If there is correlation found, depending upon the numerical values measured, this can be either positive or negative. We found out that the variables like Cloud coverage, Relative humidity, Visibility and Temperature showed positive and negative correlation associations thus have to be considered while building model. Variable importance analysis shown by Random forest also depicted the same

3.5 System Design Flow Diagram

The following diagram depicts the flow diagram of our system development process:

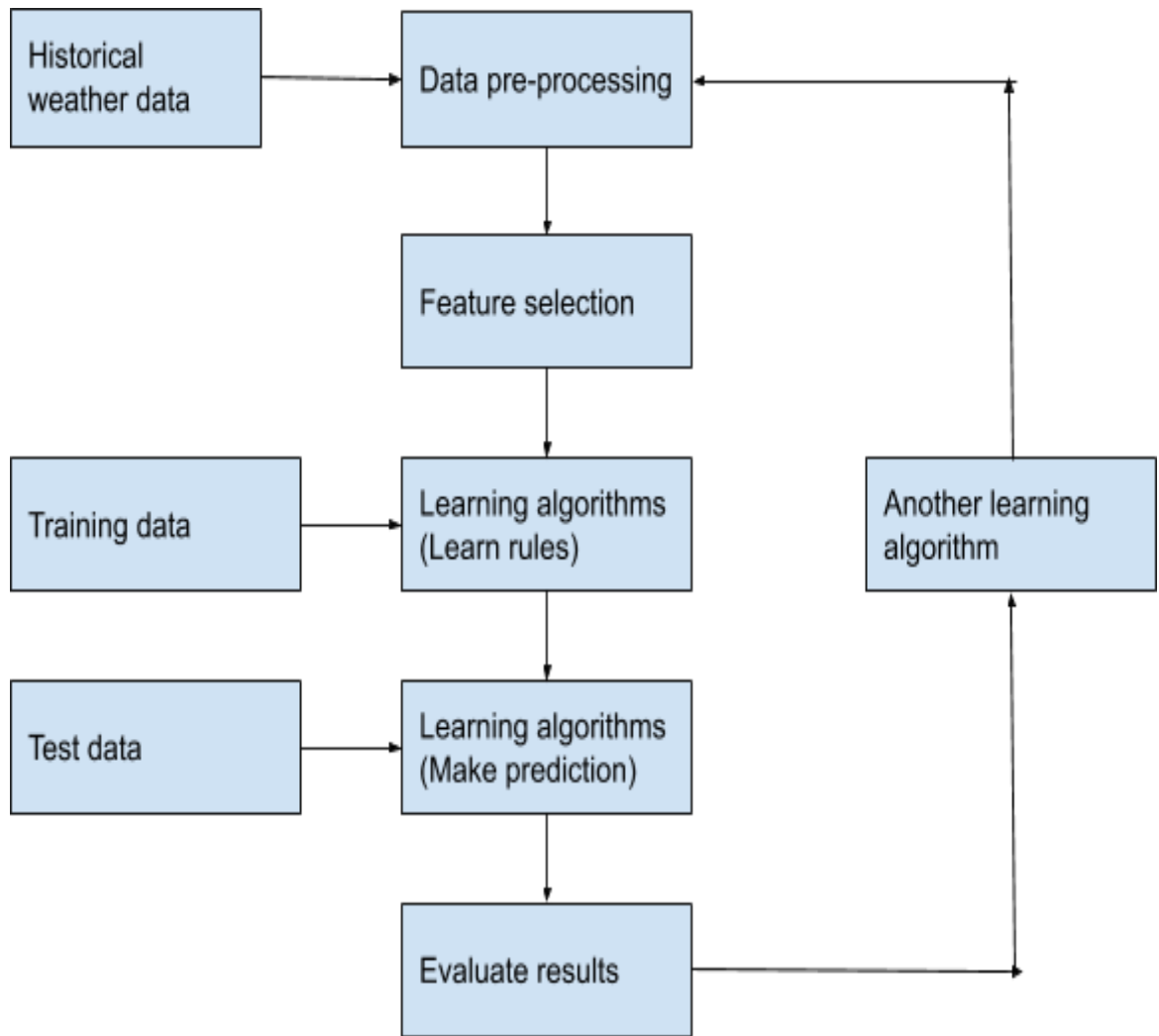


fig (viii) – System design flow diagram

3.6 Evaluation Metrics

Various types of model evaluation metrics are used for Regression models. Here, I used mean squared error. As already discussed in introduction, the various methods used to predict the incoming solar energy. Various well defined numerical weather prediction models are already in use to predict the incoming solar energy. In this project we have explored the third method i.e. prediction with the help of machine learning models to forecast the PV or solar insolation output. We have compared the performances of the three machine learning algorithms discussed in chapter 4. We are going to measure the performance using three error metrics namely, mean absolute error (MAE), mean bias

error (MBE) and root mean square error (RMSE). We also did Cross-validation techniques in order to choose best parameters for number of decision trees to use in Random forest ,max depth of tree,tree split factors etc. Apart from these in we used cross-validation to estimate best value of k in KNN and to define best value of alpha in Lasso Ridge.

3.6.1 Mean Absolute Error (MAE)

Mean Absolute Error is an evaluation metric mostly used for measuring error difference between true and measured value. Absolute error is used to calculate model's forecast accuracy. It is the difference in the predicted value and true value.

$$\Delta x = |predicted\ value - true\ value|$$

So as the name suggests Mean Absolute error is the average value of the absolute errors in set of predictions. It tells us on an average how much one can expect error in forecast. and is measured on the same scale as the data.

Calculation of MAE:

$$MAE = \frac{1}{N} \sum_{i=1}^N \left(\left| \frac{D_{ie} - D_{im}}{D_{im}} \right| \right)$$

where,

D_{ie} – Estimated value

D_{im} – Measured value

N – Number of observations

Problem which is related to the MAE calculation is that it requires more complicated tools (linear programming) to evaluate gradient and is more sensitive to outliers so in order to remove this inconsistency we calculate MAE in percentage terms which is called as Mean Absolute Percentage Error(MAPE).

3.6.2 Mean Square Error (MSE)

Mean square error of an estimator is measure of squared average of the errors i.e. average squared difference of squared values of true and measured values. It is exploring of how close is the fitted line to the data points. MSE is always positive is due to randomness. The lower is MSE value of the model is better.

Calculation of MSE:

$$MSE = \sum_{i=1}^N (D_{ie} - D_{im})^2 / N$$

where,

D_{ie} – Estimated value

D_{im} – Measured value

N – Number of observations

3.6.3 Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) value is just the square root of the mean square error. It is the average distance measure of data points from the fitted line along the vertical axis.

Calculation of RMSE:

$$RMSE = \sqrt{\sum_{i=1}^N (D_{ie} - D_{im})^2 / N}$$

where,

D_{ie} – Estimated value

D_{im} – Measured value

N – Number of observations

3.6.4 k-Fold Cross-validation

k-Fold Cross-validation is a technique derived from cross-validation modelling technique. Cross-validation is used to measure skill of the machine learning model over a limited dataset. k-Fold cross validation says to divide data sample into k groups. These later methods are used to estimate model performance on unseen data other than training sample. If model is divided in k=7 groups then it is called as 7-fold cross validation. We train model on 6 samples and validate on 1 (sample the grey box), over next iteration we train it over on other six samples and validate on different sample then in the first iteration.

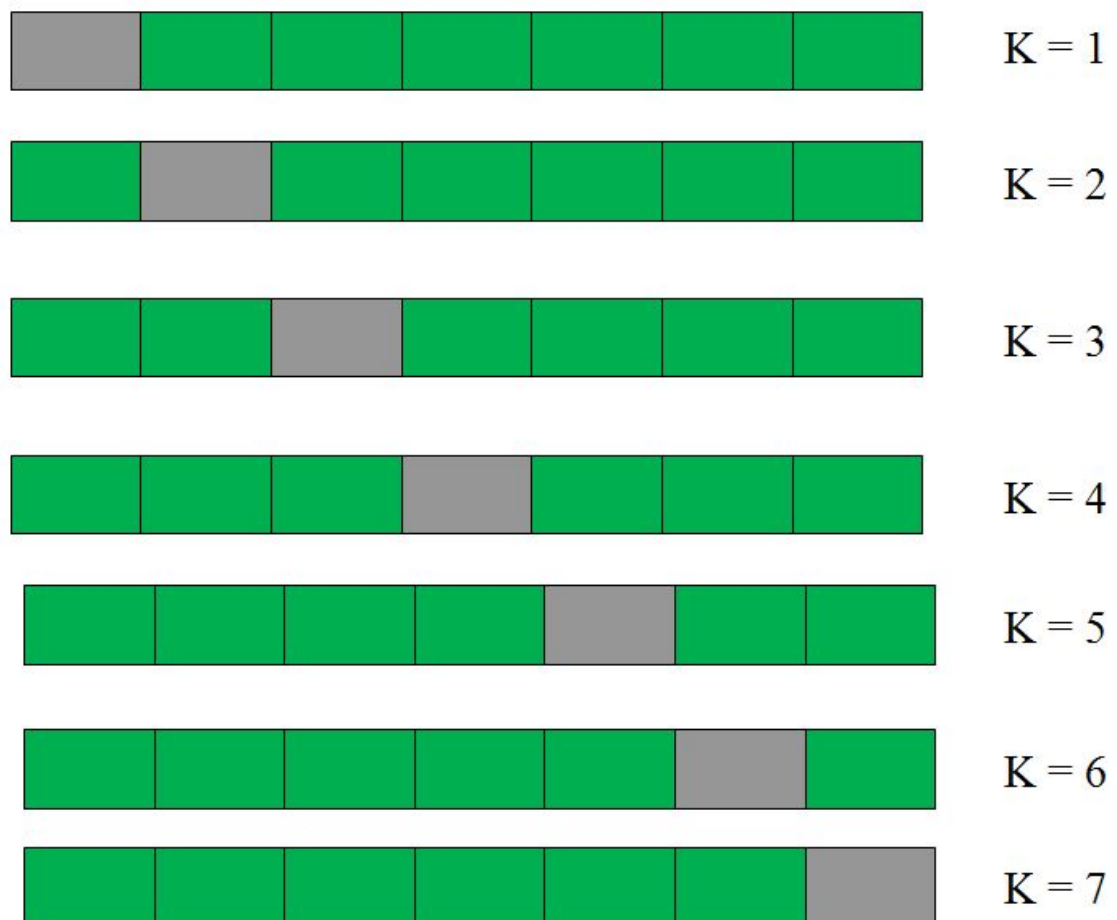


fig (ix) 7 – fold cross – validation process

Cross-validation technique is also used in estimation of best parameter coefficients of model. There are two types of parameters model parameters and hyperparameters (which are learning parameters) so within each fold in k we fix the later parameters and try to learn the hyperparameters later average them to gain the set of best estimates of parameters and then train our model over them. In our implementations of KNN, Random

Forests and Lasso Ridge regression models we used Grid search cross-validation and k-Fold cross validation in tuning best learning parameters for values of k,number of decision trees and alpha coefficients from a range of values.

3.6.5 R-squared score

R-squared score or coefficient of determination can be interpreted as the percentage of variance of the dependent variables explained by the independent variables of regression model.It is the statistical method of measuring best fit line of data points in linear or multilinear models.It is also known as coefficient of determination,higher the R-square score better is the model and it is always between 0-100 %

Calculation of R-square:

$$R - square\ score = 1 - (explained\ variance / total\ variance)$$

Explained variance is calculated as the squared sum of errors that is differences of predicted values and true values.

Total variance is average true value minus average predicted value and then square the results.

3.7 Operating environment

3.7.1 Python 3.7.3

Python is an free open source programming language which can be termed as an interpreted language that is python script is not compiled like any other languages it has an interpreter which checks script's code line by line.All the python codes are executed line by line.Python works over nearly every platform known Windows, Linux, Mac OS.It is used to develop web based applications provide backend database connection applicability,software development etc. Python scripts are more readable and less complex than any other languages.

Python today is considered more by developers because of vast range or variety imports, packages and modules provided. It contains many modules for implementing machine learning models, visualization of plots and graphs and many more things. Python also allows us to install external modules using pip command even from the external sources thus if we need a module which is not in python source packages then maybe someone around the world corner may have implemented it and exported the module all we have to do is find it and install it using pip command. This python feature is most useful in today's world we can build modules and upload them so anyone can use it in their program. In our project we used various python modules like sklearn, pandas library, numpy and matplotlib.

3.7.2 Platforms Used

3.7.2-(a) Pycharm

Pycharm is an integrated development environment (IDE) which is mainly built to write and edit python scripts. Pycharm editor enables programmers to write top notch python codes. Pycharm contains an inbuilt package installer and interpreter, contains editor themes and automatic line indents, error traceback and code suggestions which make it more for programmers to easily manipulate or analyze their code. Further it allows developers to implement database connectivity for which support tools are already integrated in pycharm. Pycharm IDE also contains command terminal below the editor hence adds on to the more user convenience.

Pycharm also supports web based application development for python. It contains python's most popular web development modules like Web2Py and Pyramid frameworks. Pycharm also has visualization tools support making it easier for developers to analyze plots and graphs. Most important of all it has pre-installed pip setup tool for installing different modules and libraries conveniently. Like other IDEs pycharm also allows developers test their applications by using various types of testing frameworks Doctests and Nose.

3.7.2-(b) Jupyter Notebook

Jupyter notebook is most recent (released in 2014) open source web based project which provides live code sharing and editing. It supports various languages like Python, R and Julia. Unlike any python scripts which execute all code lines in run but jupyter notebook

allows us to execute only few lines since it uses cells. Cells are the space provided to write code. Jupyter notebook python files are saved as .ipynb extension which is only because files are run as kernel. Jupyter notebook allows developers to execute few lines of code and output them then run next cell separately and output it. Jupyter supports many features same as any IDE like visualization plots, computational features, modules importing etc. For our project we used Jupyter notebook in order to get familiar with particular technology.

3.8 Design issues

We faced many design issues while build the system for our project:

- 1) Choosing which machine learning algorithms for processing weather data was one of the main issues. We studied lot of papers to figure out the best algorithms for this particular topic. We choose a good mix of algorithms which can work on linear as well as non-linear datasets. We found that gradient boosting, ANN provide good results according to various research papers. Apart from those we decided to use lasso and ridge, random forest and k-nearest neighbours .
- 2) Initially we decided to use weather metrics as well data from solar production plants to feed our machine learning models but we choose to use only weather variables for prediction of solar energy output as processing and removing useless variables from two huge datasets and then training our models on them would have taken a lot of time.
- 3) Second issue we faced was deciding tree split factors for our Random Forest regression model which we resolved using Grid Search cross-validation technique.
- 4) We faced problem in choosing distance method used to compare similarity of input instance with data point. We decided to choose euclidean distance method as it is the most common method used in case of k-nearest neighbours
- 5) We faced similar problem of choosing best value of k in KNN so we used cross validation again with RMSE value as evaluation metric of model at particular k value.

- 6) We had two options while choosing programming language: python and R. We decided to go for python as most machine learning libraries are inbuilt in python. it is very well supported and there are lots of resources available on internet for machine learning with python for reference as compared to R.

Chapter-4

PERFORMANCE ANALYSIS

In this chapter, we provide detailed explanation about the algorithms that we are going to use in our project to model for solar insolation prediction. We are going to use five machine learning algorithms namely; Lasso and ridge, k-Nearest neighbours and Random forest.

4.1 Lasso and Ridge

4.1.1 Ridge regression

Ridge regression is just like lasso regression but instead of adding factor of sum of absolute of coefficients it adds a factor of sum of squares of coefficients in optimization objective. This is known as L2 regularization. We can make more stable predictions by decreasing the coefficient estimates to zero hence decreasing the variance. The function that ridge regression minimizes is:

$$\text{Objective} = \text{RSS} + \alpha * (\text{sum of square of coefficients})$$

or

$$O = \text{RSS}(\theta) + \alpha \sum_{k=1}^{304} \theta_k^2 \quad - \text{ (viii)}$$

Here, α (alpha) is a tuning parameter that equates to the amount of significance given to minimizing RSS as compared to sum of square of coefficients, by cross-verification of the training dataset we can get optimal value of α . Values of α can be taken as:

- $\alpha = 0$: In this case the value of objective is same as what we get in simple linear regression. Also the coefficients are similar to SLR.

- $\alpha = \infty$: In this case the coefficients become zero because the weightage given to the square of coefficients is infinite which will cause the objective to tend to infinite for any value less than zero.
- $0 < \alpha < \infty$: Here, the weightage which are given to different components of objective are evaluated on the basis of magnitude of α . For SLR, the coefficients will lie from 0 to ones.

Since, α is the hyperparameter of ridge regression so we have to set the value of alpha manually by plotting different ranges of value of α versus coefficients. Then as iterating through different ranges of values and taking the R-square of each α we can get the α with maximum R-value, which will be our shrinking factor with lowest error.

Cost function of the ridge regression:

$$\min(\|Y - X(\theta)\|_2^2 + \alpha\|\theta\|_2^2) \quad - (ix)$$

The extra term introduced in the equation is known to be as penalty term which is controlled by values of alpha. Larger the value of alpha, higher is the penalty and hence better reduction in the magnitude of the coefficients.

Ridge model implementation:

For our Lasso and Ridge regression implementations we divided our dataset in 75:25 training and testing by using sklearn train-test split function. Shapes of training and testing samples is given below.

Training Features Shape: (477, 8)

Training Labels Shape: (477, 1)

Testing Features Shape: (160, 8)

Testing Labels Shape: (160, 1)

here Labels corresponds to variable which we have to predict.

Since Ridge uses alpha as argument to tune the model's performance so in order to decide value of alpha instead of choosing it arbitrarily we defined an array of alpha values to choose value of alpha and later ran cross-validation to get the value. We generated a range of values of alpha from small to large values using

```
alpha = 10 ** np.linspace(10,- 2, 100) * 0.5
```

We implemented two models of Ridge() to make predictions with normalize parameter as True so that all features are in same scales. The first model was implemented over a value of alpha=0 and predictions were made below is code snippet

```
Prediction ridge 1(alpha = 0): [[26536.84668687]
 [36113.19523348]
 [ 6437.17047696]
 [27887.32806884]
 [19270.47313771]
 [29703.002131 ]
 [17526.70835744]
 [ 9389.25444659]
 [17747.76930025]
 [ 8793.08993908]
 [24339.65737745]
 [ 7505.88652297]
 [26195.00215203]
 [18098.73615148]
 [ 7929.95894656]
 [18731.78477363]
 [20937.95342405]
 [38281.95680629]
 [18948.03582192]
 [23791.36337995]
 [29617.32823708]
 [18188.94796695]
 [25521.49750135]
 [15954.16241826]
```

Evaluation of Ridge model 1(alpha=0) error metrics and coefficients snippet:

```
Ridge 1 coefficients: [[-4901.74833011  247.27743909  7725.87361412 -5639.85799163
 -1995.88963086 -421.60533939  2644.23862812 -166.74914848]]
MSE Ridge 1: 35414101.72933619
R squared value: 0.66
Mean absolute error: 4562.9
```

Before implementing the second model of Ridge first we did cross-validation in order to get an adequate value of alpha. We used both GridSearchCV and RidgeCV imports of sklearn for selecting value of alpha. Results were as follows:

- Value of alpha using RidgeCV resulting smallest cross-validation error: 0.05
- From Grid search cross-validation we got

ridge best parameters: {'alpha': 0.1}

As we can see there is not much difference in both results so we trained our second model over alpha value 0.05 made predictions and evaluated metrics.

```
MSE:35326740.35787199
Mean absolute error: 4516.39
R squared value: 0.66
Ridge 2 coefficients over full dataset:[[-4627.99563657  -60.41102307  3419.61656027  -907.61866955
 -3705.52727093  -395.96267254  3495.96774466   28.55634623]]
```

Here results we got of mean absolute error, mean square error and R square score improved then what was from Ridge model 1. Coefficients also got reduced by great factor of model 2 then what model 1 gave, but none of the coefficients reduced to zero since ridge does not perform feature selection.

4.1.2 Lasso regression

Lasso implies for Least Absolute Shrinkage and Selection Operator. Two basic keywords are used in Lasso which are absolute and selection. It adds a factor of sum of absolute value of coefficients in optimization objective which is known as L1 regularization. The function that lasso regression optimizes is given as:

$$\text{Objective} = \text{RSS} + \alpha * (\text{sum of absolute value of coefficients})$$

In lasso we use both the mean square error value and R-square value for our model for better predicting than that of ridge where on increasing the value of alpha coefficients limited to zero but no absolute zero but in case of lasso regression coefficients were

reducing to absolute zero even for lower values of alpha this is known as feature selection.

Cost function for lasso regression:

$$\min(\|Y - X\theta\|_2^2 + \alpha\|\theta\|_1) \quad - \quad (xi)$$

we can see that unlike like ridge approach of adding squares of θ lasso uses absolute value of θ .

Lasso Model implementation:

For our Lasso model implementation we divided our training and testing samples in a ratio same as what was in Ridge i.e. 75:25. The normalize parameter was set to be true since we don't want features be in different scales. Same as ridge before building lasso model we did 10 fold cross-validation and max iterations of 100000 for alpha value using both grid search and LassoCV. Results generated were as follows

- Value of alpha after Lasso cross-validation: 6.576069739510901
- Value of alpha after Grid Search cross-validation

lasso best parameters: {'alpha': 10.0}

We got two different results for the alpha, so we trained our models on both the alpha value and evaluated the metrics in order to decide which alpha value is giving us improved results.

Metrics given by alpha=6.5760

```
Lasso coefficients: [-4854.14476019  17.39513344  2285.53297563  0.
-3772.67975388 -176.26310411  3376.91207593  -0.      ]
MSE Lasso: 35251459.17617577
Model score: 0.6610640484074302
R2 value: 0.6610640484074302
Mean Absolute error: 4518.81
```

Metrics given by alpha=10.0

```
Lasso coefficients: [-4901.75215972  16.93606183  2238.98492353  0.
-3606.76085008  -109.154935  3244.91177156  -0.  ]
MSE Lasso: 35430440.59932844
C:\Users\Sudhanshu\PycharmProjects\SolarPrediction\venv\lib\site-packages\s
warnings.warn(CV_WARNING, FutureWarning)
Model score: 0.6593431766934262
R2 value: 0.6593431766934262
Mean Absolute error: 4552.28
```

As we can see that values of errors increased at $\alpha=10$. Thus we used alpha value of 6.5706 given by lasso cross-validation and refitted the model in order to make predictions.

```
Lasso model prediction: [26481.36387757 35338.73740106 6602.6197141 27781.95788836
19304.72036019 29501.19240315 18185.86937633 9695.27078601
17902.27699284 8543.53549613 24346.29150353 7653.64126815
26536.42570205 17754.52698719 8007.36368726 18769.46595122
21001.10527858 37623.06263145 18591.23912373 23762.66756523
29157.94474909 17903.20307214 25449.18498926 16370.48101209
24439.42710589 21642.990897 29442.73690226 26331.93272199
25056.10579576 32554.8605142 11709.30859618 20833.25956548
30143.12065461 28415.58141055 24048.17897394 24468.60326309
8228.79178973 23844.35416902 7075.95508651 7894.32695295
8072.93583945 22367.97955668 37923.84530581 19835.55027059
34283.43089428 29652.80740877 4125.79741657 34691.33093387
28059.48165718 24588.76208552 21482.1816089 26959.66111526
12564.25886034 13683.70580231 17373.55991198 18260.53753495
15210.82202761 9814.12379273 13560.90060354 3981.02690857
17675.11841263 7536.81356348 24426.23057132 22228.41054733
27653.38376827 18299.04579777 33046.8497824 10373.59814054
22319.15626544 6881.70127704 29157.44850536 18117.44315641
29950.83303549 23272.60367003 30655.73312089 17388.2818815
29237.78230131 25405.31674861 24053.22704697 17581.31953806
8183.40770944 20809.59026039 26130.4937721 24741.64423821
33200.66903867 25100.97813424 3599.24767952 28567.59743595
17279.1882743 33628.43727755 22727.63156033 20856.93180639
13824.93821186 27256.1486529 13933.47873887 8721.58201664
9998.77427528 32961.12166669 18275.67812201 24780.37426072
```

We can see that the some coefficients given by Lasso are reduced to exactly 0 (fig) then the coefficients given by the Ridge since Lasso also performs feature selection. The features with 0 coefficients were Dew point and Altimeter Also we can see that the Lasso model returned improved or slightly better results than the Ridge.

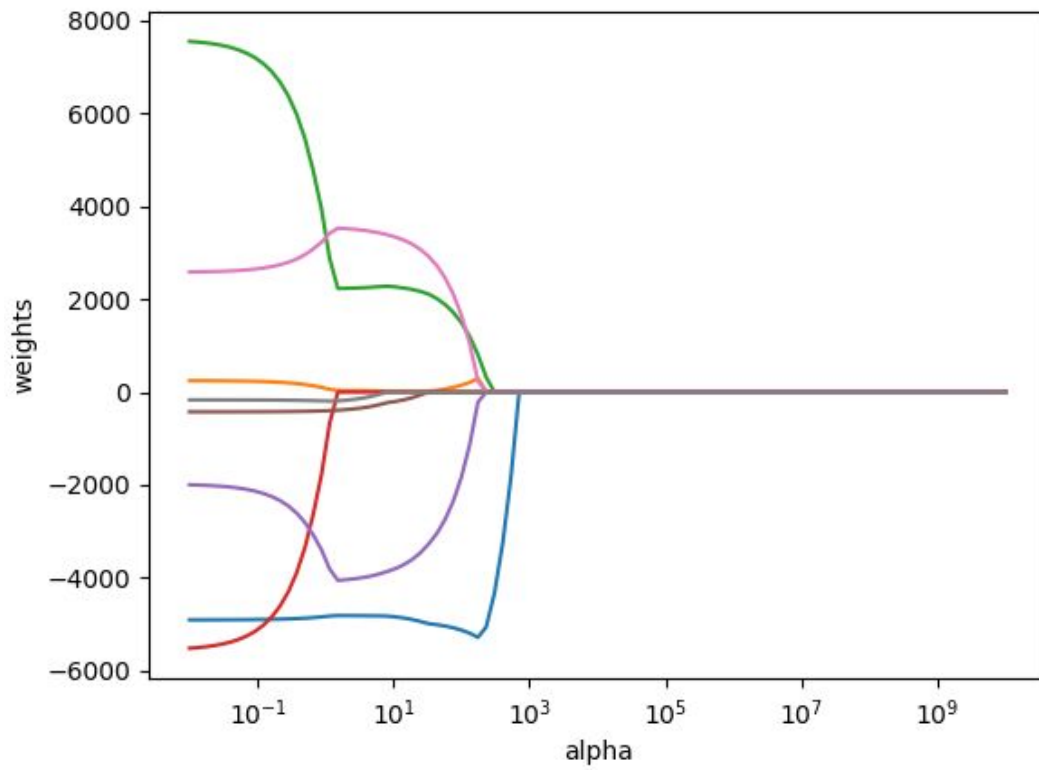


fig (x) Lasso coefficients vs alpha plot

4.2 K-nearest neighbours:

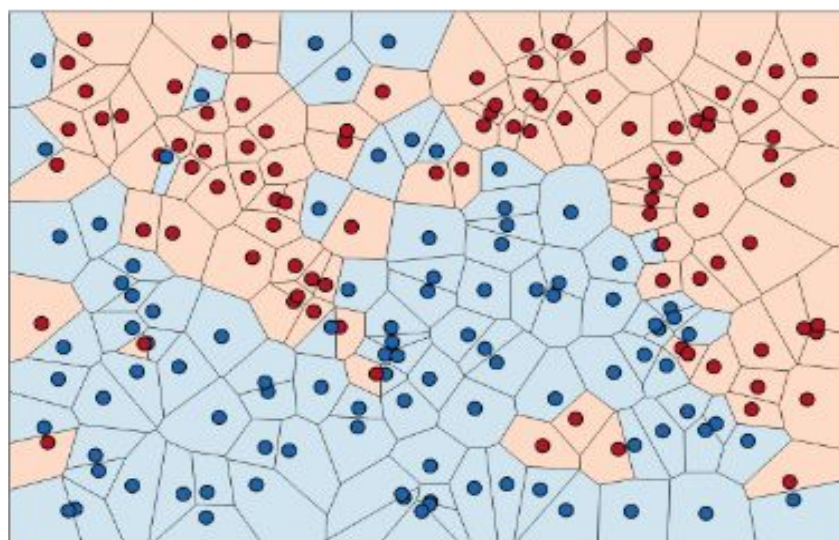


fig (xi) – Nearest neighbourhoods for each point of an training dataset [11]

It is one of the simplest algorithm (classifier and regression based) used in the industry for performing variety of tasks like predictions for economy, in the genetic industry for performing prediction involving genes, it can also be used for weather prediction. In our case, we are using it for incoming insolation output prediction using weather metrics. Functioning and performance of this classifier can be compared to more difficult and complex algorithms like artificial neural networks.

For this algorithm to work, the dataset is divided into different classes which in turn are used to predict the class of new sample point given to the algorithm. It doesn't make an assumption based on how the data is distributed in our database that's why it is called a non-parametric algorithm. The model for KNN is evaluated based only on the structure of the data. This algorithm works best when we have very little or no knowledge about our data. As shown in the diagram above KNN traverses the whole dataset and finds the k most values (represented by red regions) which are similar to the instance provided to the algorithm.

Not everything is perfect about this method as storing the entire data as training data, and then traversing & comparing it to find similar classes for a given input dataset is a computational expensive task. A lot of computing power is required to do this. This may also take a lot of time to predict output.

Now, how do we find similar instances of the given input instance in previously unseen data. This is usually done by using distance methods which take two points as input. One of the most common method is the Euclidean distance technique which is represented by the formula given below:

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \dots + (x_n - x'_n)^2} \quad - (xii)$$

There are three other distance techniques which we can use instead of the euclidean distance namely:

- 1) Hamming distance

- 2) Minkowski distance
- 3) Manhattan distance

Depending on the dataset and situation, which distance formula is most suitable is decided.

Basically, KNN algorithms perform three steps which are described below:

- 1) Given an input point x , first it finds distance d between x and the training data points and finds the set B containing k elements of observation points which are closest to input x .
- 2) Then conditional probability for each class is calculated :

$$P(y = j | X = x) = \frac{1}{k} \sum_{i \in B} I(y^{(i)} = j) \quad - \quad (xii)$$

- 3) Then, it assigns x to the class with maximum probability.

Model implementation:

For implementing KNN regression model at first we scaled the data and then for this algorithm we decided to change splitting of test and train samples. We did 75:25 test and train split considering dimensions of our dataset.

Training Features Shape: (318, 8)

Training Labels Shape: (318,)

Testing Features Shape: (319, 8)

Testing Labels Shape: (319,)

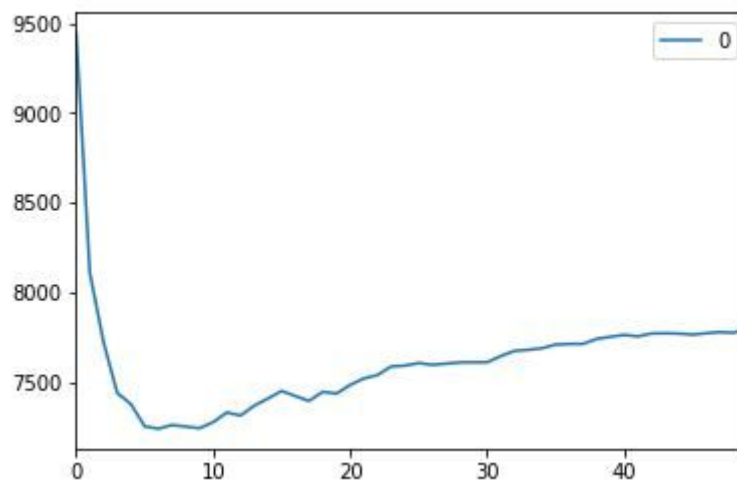
Next, we did both Grid Search cross-validation and RMSE value analysis in order to choose value of k .

- For rmse value analysis we iterated over 1-50 values for k and fitted them over our KNN model for each value. We observed that after $k=7$ the RMSE value again picked up and didn't show improvement after $k=7$.


```

RMSE value for k= 1 is: 9449.884011993237
RMSE value for k= 2 is: 8109.019933949166
RMSE value for k= 3 is: 7721.486625679916
RMSE value for k= 4 is: 7437.895239899908
RMSE value for k= 5 is: 7375.487076597266
RMSE value for k= 6 is: 7251.554973494442
RMSE value for k= 7 is: 7238.401599759698
RMSE value for k= 8 is: 7258.843660866896
RMSE value for k= 9 is: 7250.189891466594
RMSE value for k= 10 is: 7240.808900678446
RMSE value for k= 11 is: 7274.254972704223
RMSE value for k= 12 is: 7328.90894843238
RMSE value for k= 13 is: 7311.944487869804
RMSE value for k= 14 is: 7366.578212135854
RMSE value for k= 15 is: 7406.6982871261125
RMSE value for k= 16 is: 7448.826423790949
RMSE value for k= 17 is: 7420.7717939070735
RMSE value for k= 18 is: 7392.726239931539
RMSE value for k= 19 is: 7443.943833237867
RMSE value for k= 20 is: 7434.63026333259
RMSE value for k= 21 is: 7482.374241044659
RMSE value for k= 22 is: 7518.624833895682
RMSE value for k= 23 is: 7537.7421465803545
RMSE value for k= 24 is: 7585.752294597718
RMSE value for k= 25 is: 7590.138915767089
RMSE value for k= 26 is: 7604.920615106677
RMSE value for k= 27 is: 7594.784507434279
RMSE value for k= 28 is: 7601.402594520623
RMSE value for k= 29 is: 7608.092181371109
RMSE value for k= 30 is: 7609.215159495102
RMSE value for k= 31 is: 7608.776044696831

```



. fig (xii) – RMSE value vs k value plot

- Grid Search 10 fold cross-validation over values of k gave same result as RMSE value evaluation.

`{'n_neighbors': 7}`

At last we trained and tested our KNN model over k=7 in order to make predictions and evaluate metrics. Some of predictions are as follows.

[28361.28571429 32150.57142857 6250.85714286 21034.85714286
 20407.14285714 32446.57142857 21344.14285714 12070.57142857
 28010.71428571 7626. 21284.57142857 9495.
 20675.28571429 25428. 10413.42857143 20470.28571429
 20560.42857143 34896.42857143 21489. 21595.85714286
 29999.85714286 23100.71428571 29596.42857143 23662.28571429
 26381. 26087.85714286 29936.42857143 32890.
 21298.42857143 29489.42857143 14665.42857143 21421.42857143
 29436.42857143 26089.28571429 25934.71428571 23806.57142857
 7769. 27745.57142857 5803.57142857 7892.28571429
 8230. 25734.71428571 33534.14285714 23704.57142857
 30123.57142857 24086.42857143 5467.14285714 29964.
 33674.71428571 24806.28571429 20778.85714286 21100.14285714
 13266.57142857 11085.71428571 20148.28571429 17608.42857143

Evaluation of metrics:

Mean Absolute Error: 5632.89

Mean Squared Error: 52394457.72

R-squared scores: 0.57

KNN model score: 0.657

We considered Lasso coefficients for dropping features. We dropped Wind speed and Visibility because Lasso coefficients of both were least and Altimeter and Dew points coefficients were zero. After dropping these features we repeated the same process, first we did cross-validation for optimal k value we got k=16. Then we refitted and trained our model over this k value and made predictions and evaluated error metrics.

Evaluation of error metrics after dropping features:

Mean Absolute Error: 4776.18

Mean Squared Error: 39247293.29

R-squared scores: 0.62

KNN model score: 0.63

Accuracy: 49.76 %.

We can see here that after features selection our model results improved than they were before feature selection.

4.3 Random Forest

Random Forest algorithm is a supervised classification algorithm which goes for both regression and classification. It ensembles a randomised forest, a decision tree model without use of any adjusting parameter. Its easiness and capability to even work for inconsistent data makes it most commonly used among any other regression algorithms.

Random Forest algorithm involves building of numerous possible decision trees from different input samples and variables and later merging all of them together to get a more stable result. More the number of the trees, more accurate is prediction result. In random forest decision tree act as a backup tool, we set some input parameters, target values and features from training dataset and decision tree randomly devises some set rules based on features and relationship between variables which are used to find possible relations and outcome from the devised tree graph. It extracts the best sets of features from the subsets of decision trees.

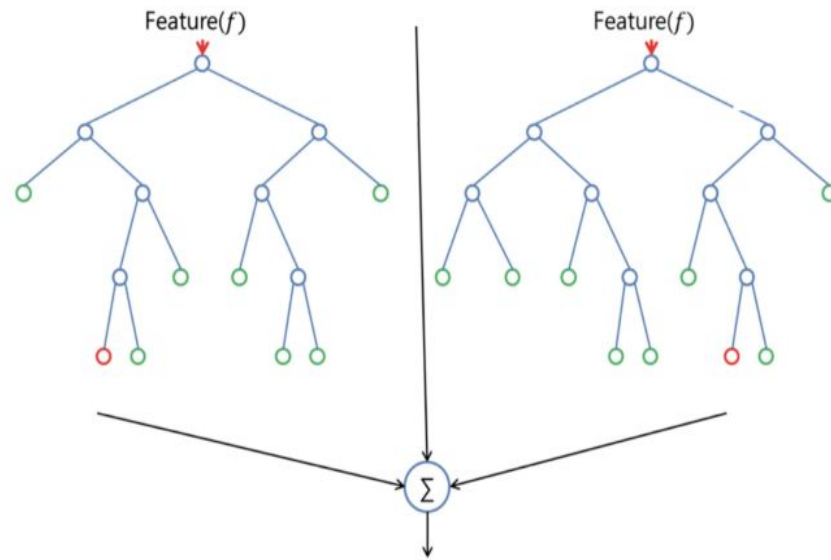


fig (xiii) Random forest model structure

Now how random forest algorithm works, first a 'n' forest of trees are randomly generated from 'm' number of features then a node 'x' which gives us best root node is selected among 'm' and is splitted into children nodes further to leaf nodes .Then features that hold similarity are extracted and combined for prediction analysis. Afterwards ,final prediction is deduced by extraction of most relative features or most voted feature based on our prerequisites from decision trees.

Model implementation:

For Random forest implementation we divided our training and testing data samples in a ratio of 75:25 considering size of dataset. For deciding the maximum depth and number of trees parameters for our model we implemented Grid Search 5 fold cross-validation over ranges of values of number of trees and maximum depth. We got optimal values of both parameters as number of trees = 1000 and max depth =70. Based on these results we trained our random forest regressor and made predictions as follows.

[22318.672, 23843.813, 10536.529, 24009.025, 21410.114, 24043.065,

16012.261, 11315.736, 21217.23 , 10973.309, 23849.972, 12152.552,
23837.437, 20533.873, 10973.309, 21663.08 , 21528.621, 23843.813,
21522.807, 23864.645, 15115.423, 21182.378, 24159.909, 22867.876,
21974.231, 22318.672, 23799.916, 20767.913, 23837.437, 23843.813,
20956.475, 21522.807, 24158.788, 23788.866, 23309.836, 22629.057,
13739.815, 24159.909, 9903.699, 12074.505, 12152.552, 23919.188,
23863.075, 21968.419, 24044.578, 23793.662, 10878.583, 23843.813,
20968.913, 15341.99 , 21663.08 , 23919.188, 14053.377, 16063.631,
21410.114, 17027.015, 17027.015, 12152.552, 14538.22 , 10973.309,
21667.978, 11420.401, 23309.836, 22318.672, 23910.248, 17022.739,
23843.813, 15118.73 , 23309.836, 10659.475, 23919.188, 21583.134,
23811.562, 23958.618, 23843.813, 21583.134, 23932.988, 23938.217,
21968.419, 14538.22 , 10627.383, 21310.135, 23977.465, 23837.437,
23910.248, 23966.222, 10685.018, 23811.562, 21583.134, 23843.813,
23898.143, 21667.978, 13750.686, 15155.627, 14856.934, 12505.84 ,
12010.096, 23863.075, 21528.621, 23843.813, 21410.812, 12074.505,

We evaluated error metrics and got results as follows

Mean Absolute Error: 5343.19

Mean Squared Error: 46874980.18

R-squared scores: 0.61

Random forest model score: 0.94

For more further analysis we evaluated variable importances using previous model in order to drop some feature and refit the random forest model to watch for any improvements. We got results for variable importances as

Variable: Cloud coverage Importance: 0.52

Variable: Relative humidity Importance: 0.12

Variable: Temperature Importance: 0.08

Variable: Station pressure Importance: 0.08

Variable: Visibility Importance: 0.07

Variable: Wind speed Importance: 0.06

Variable: Dew point Importance: 0.05

Variable: Altimeter Importance: 0.03

We can see that Altimeter ,Dew point ,Wind speed and Visibility are the variables which hold least importances.We considered Lasso coefficients also while deciding features.We dropped Wind speed and Visibility because Lasso coefficients of both were least and Altimeter and Dew points coefficients were zero.

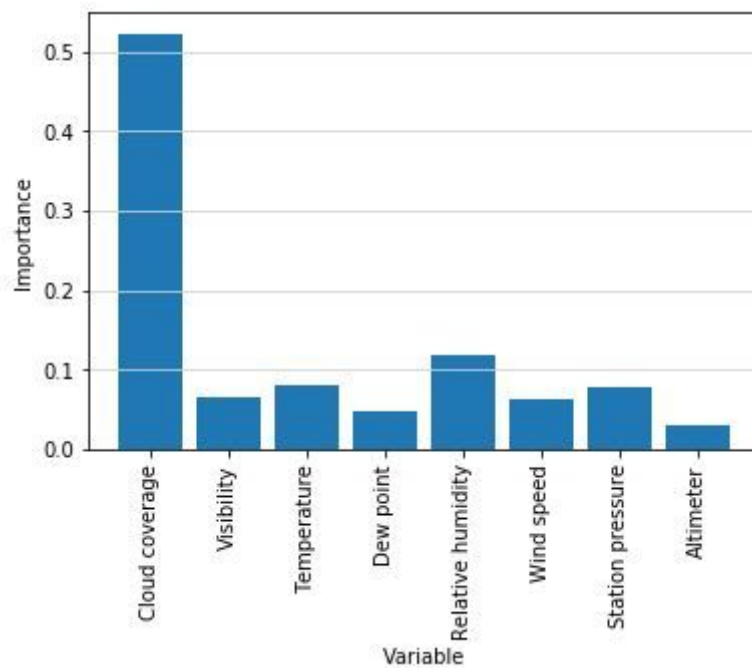


fig (xiv) – Variable importances bar plot

Model evaluation after feature selection we observed that the results improved further both in accuracy and MAE values.

Mean Absolute Error: 4934.29

Mean Squared Error: 40772436.43

R-squared scores: 0.61

Random forest model score: 0.9425107240800619

Accuracy: 48.81 %.

Predictions were as follows:

	Cloud coverage	Temperature	Relative humidity	Station pressure	Solar energy	prediction
107	0.39	13.68	57.10	29.39	28780	26030.165
74	0.00	16.00	48.43	29.37	31522	29493.289
373	0.85	-2.69	87.17	29.15	1896	7997.184
565	0.16	23.02	73.06	29.17	25978	27423.715
550	0.52	16.30	76.83	29.21	22924	16618.141
114	0.15	23.59	65.76	28.98	28642	20647.186
347	0.33	-2.75	77.50	29.82	10681	20275.720
24	0.88	1.29	77.66	29.20	14633	7810.425
325	0.30	1.78	83.18	29.51	14016	23095.279
635	0.99	3.43	77.87	29.15	6400	8082.120

4.4 Results and Model comparison

Table 3. Model Comparison

Regression Model	MSE	MAE	R-squared score	Model score
Lasso	35251459.17	4518.81	0.66	0.66
Ridge	35326740.35	4516.39	0.66	0.66
KNN	39247293.29	4776.18	0.62	0.63
Random Forest	40772436.43	4934.29	0.61	0.94

Chapter 5

CONCLUSION

5.1 Conclusion

This project was our first project in machine learning field. We were able to understand regression modelling techniques using some of the most versatile algorithms, we were able to understand model evaluation metrics, how to do feature selection from large number of features, how to scale them and cross-validate them to get optimal tuning parameters and how to plot and interpret visualization graphs.

We concluded that Cloud coverage, Temperature, Relative humidity and Station pressure features played important role in forecasting Solar energy output over solar PV panels. We also concluded that feature selection techniques used using Lasso coefficients and Random Forest variable importance did yield results quite similar to each other.

We concluded that Lasso regression model forecasts yielded least error in forecasting Solar output. We also considered Random Forest model to be the second best based on its model scores and due to quite less difference between its error and Lasso's errors. We also conclude that dimensions of dataset (how many entries dataset have) as well as model training split are important when building a regression model.

5.2 Future Scope

According to what we learned and observed in this project we can say that solar output forecasting can be pre-estimated using machine learning prediction models. In reference to some of the research papers Artificial neural networks (ANN) technique can be proven more efficient for Solar forecasting.

References

- 1) Navin Sharma, Pranshu Sharma, David Irwin, and Prashant Shenoy. Predicting Solar Generation from Weather Forecasts Using Machine Learning of Department of Computer Science University of Massachusetts Amherst, Massachusetts 01003 {nksharma,pranshus,irwin,shenoy}@cs.umass.edu
- 2) Rich H. Inman, Hugo T.C. Pedro, and Carlos F.M. Coimbra. Solar forecasting methods for renewable energy integration. *Progress in Energy and Combustion Science*, 39(6):535 – 576, 2013.
- 3) Dimuthu Dharshana Kodippili Arachchige Supervisor Professor Hans George Beye. An Approach to Day Ahead Forecasting of Solar Irradiance with an Application to Energy Gain in Solar Thermal Collectors.
- 4) Aminmohammad Saberian, H. Hizam, M. A. M. Radzi, M. Z. A. Ab Kadir, and Maryam Mirzaei. Modelling and Prediction of Photovoltaic Power Output Using Artificial Neural Networks of Department of Electrical and Electronic Engineering, Universiti Putra Malaysia (UPM), 43400 Serdang, Malaysia.
- 5) Daniel O’Leary and Joel Kubby. Feature Selection and ANN Solar Power Prediction.
- 6) Tao Hong, Pierre Pinson, Shu Fan, Hamidreza Zareipour, Alberto Troccoli, and Rob J. Hyndman. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond. *International Journal of Forecasting*, 32(3):896 – 913, 2016
- 7) Can Wan, Jian Zhao, Yonghua Song, ,Zhao Xu, Jin Lin, and Zechun Hu Photovoltaic and Solar Power Forecasting for Smart Grid Energy Management
- 8) Navin Sharma, Pranshu Sharma, David Irwin, and Prashant Shenoy Predicting Solar Generation from Weather Forecasts Using Machine Learning Department of Computer Science University of Massachusetts Amherst
- 9) Anamika, Rajagopal Peesapati, and Niranjana Kumar, Estimation of GSR to ascertain solar electricity cost in context of deregulated electricity markets, *Renewable Energy*, vol. 87, pp. 353-363, 2016. (Article)
- 10) <https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/>
- 11) <http://scott.fortmann-roe.com/docs/BiasVariance.html>

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 19/05/2019

Type of Document (Tick): Ph.D Thesis M.Tech Dissertation/ Report B.Tech Project Report Paper

Name: Sudhanshu Sharma Department: CSE Enrolment No 161341

Contact No. 7355282277 E-mail. sudhanshu0511max@gmail.com

Name of the Supervisor: Dr. Ravindara Bhatt


Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): PREDICTION OF DAILY SOLAR ENERGY OUTPUT USING WEATHER DATA

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, If I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

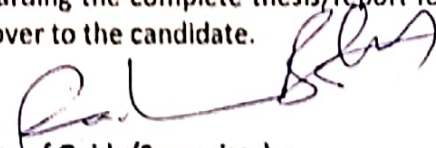
Complete Thesis/Report Pages Detail:

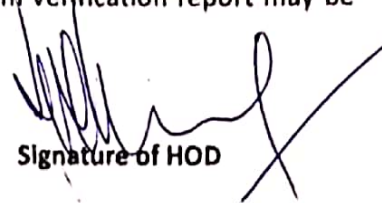
- Total No. of Pages = 547
- Total No. of Preliminary pages = 6
- Total No. of pages accommodate bibliography/references = 2


(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found Similarity Index at12.....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.


(Signature of Guide/Supervisor)

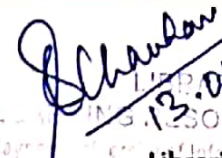

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
<u>13.05.2019</u>	<ul style="list-style-type: none"> • All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String 	<u>6%</u>	Word Counts	<u>8,363</u>
Report Generated on			Character Counts	<u>45,376</u>
<u>13.05.2019</u>		Submission ID	Total Pages Scanned	<u>48</u>
	<u>1129558788</u>	File Size	<u>1.73M</u>	

Checked by
Name & Signature


LIBRARIAN
13.05.2019
LIBRARY RESOURCE CENTER
Jaypee University of Information Technology
Wazirpur, Distt. Meerut (Uttaranchal Pradesh)
Pin Code - 220522

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com