Report

On

Solar Power Forecasting Using ML-Model

Submitted in partial fulfillment of the

requirement for the award of the degree

of Computer Science and Engineering

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

Dr. Bassety Malligarjuna

Associate professor

Submitted by

Abhishek Tripathi(18SCSE1010057)
Ankit Sharma(18SCSE1010055)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"SOLAR POWER FORECASTING USING ML-MODEL"** in partial fulfillment of the requirements for the award of the B. Tech (CSE) submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of Jan 2022 to May 2022, under the supervision of Dr. Sampath Kumar K, Department of Computer Science and Engineering, of School of Computing Science and Engineering , Galgotias University, Greater Noida.

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Abhishek Tripathi (18SCSE1010057)

Ankit Sharma (18SCSE1010055)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Dr. Bassety Malligarjuna
Associate professor

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of ABHISHEK TRIPATHI (18SCSE1010057) and ANKIT SHARMA (18SCSE1010055) has been held on and his/her work is recommended for the award of B Tech(CSE).

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: May,2022

Place: Greater Noida

Table of Contents

Figure no.	Title	Page No.
1.	Abstract	02
2.	Introduction	02
3.	Literature Survey	03
4.	Problem Statement	03
5.	Project Diagram	04
6.	Database approch	05
7.	Data Science	06

8.	**Data**	**10**
9.	**Data Cleaning**	**12**
10.	**Data Exploration**	**15**
11.	**Data Visualization**	**18**
12.	**Tableau**	**21**
13.	**Machine Learning**	**25**
14.	**Machine Learning Model Using Python**	**33**
15.	**Implementation**	**41**
16.	**Refrences**	**50**

Abstract

The increased competitiveness of solar PV panels as a renewable energy source has increased the number of PV panel installations in recent years. In the meantime, higher availability of data and computational power have enabled machine learning algorithms to perform improved predictions. As the need to predict solar PV energy output is essential for many actors in the energy industry, machine learning and time series models can be employed towards this end. In this study, a comparison of different machine learning techniques and time series models is performed across five different sites in Sweden. We find that employing time series models is a complicated procedure due to the non-stationary energy time series. In contrast, machine learning techniques were more straightforward to implement. In particular, we find that the Artificial Neural Networks and Gradient Boosting Regression Trees perform best on average across all sites.
Estimation of solar-powered energy is becoming an important issue in relation to environmentally friendly energy sources, and machine learning algorithms play an important role in this area. Sunlight-based energy estimation can be viewed as a period series waiting problem, using standardized data. In addition, energy determination based on sunlight can be obtained from the

Mathematical Climate Assessment Model (NWP). Our purpose is centered around the final approach. We focus on the concept of sunlight based energy from the NWP registered with the GEFS, the Global Ensemble Forecast System, which assesses weather factors to focus on in the matrix. In this case, it would be helpful to know how estimation accuracy improves based on the size of the lattice hubs used in conjunction with AI methods. AI (ML) calculations have shown exceptional results over time, which can be used as model data sources to predict lightning with weather conditions. Use of various AI, Deep Learning and Simulated Brain Network methods for solar based energy decisions. Here is the relapse model featuring Machine Resistor Assist Vector, Anomalous Forest Area Registrar and Straight Relax Model from AI Techniques, of which the Arbitrary Backwoods Resistor beats the other two Relax Models with incredible accuracy

Keywords: Machine learning, Classification, Data Science

INTRODUCTION

The increased competitiveness of solar PV panels as a renewable energy source has increased the number of PV panel installations in recent years. In the meantime, higher availability of data and computational power have enabled machine learning algorithms to perform improved predictions. As the need to predict solar PV energy output is essential for many actors in the energy industry, machine learning and time series models can be employed towards this end. In this study, a comparison of different machine learning techniques and time series models is performed across five different sites in Sweden. We find that employing time series models is a complicated procedure due to the non-stationary energy time series. In contrast, machine learning techniques were more straightforward to implement. In particular, we find that the Artificial Neural Networks and Gradient Boosting Regression Trees perform best on average across all sites.
Estimation of solar-powered energy is becoming an important issue in relation to environmentally friendly energy sources, and machine learning algorithms play an important role in this area. Sunlight-based energy estimation can be viewed as a period series waiting problem, using standardized data. In addition, energy determination based on sunlight can be obtained from the Mathematical Climate Assessment Model (NWP). Our purpose is centered around the final approach. We focus on the concept of sunlight based energy from the NWP registered with the

GEFS, the Global Ensemble Forecast System, which assesses weather factors to focus on in the matrix. In this case, it would be helpful to know how estimation accuracy improves based on the size of the lattice hubs used in conjunction with AI methods. AI (ML) calculations have shown exceptional results over time, which can be used as model data sources to predict lightning with weather conditions. Use of various AI, Deep Learning and Simulated Brain Network methods for solar based energy decisions. Here is the relapse model featuring Machine Resistor Assist Vector,

Anomalous Forest Area Registrar and Straight Relax Model from AI Techniques, of which the Arbitrary Backwoods Resistor beats the other two Relax Models with incredible accuracy

Literature Survey

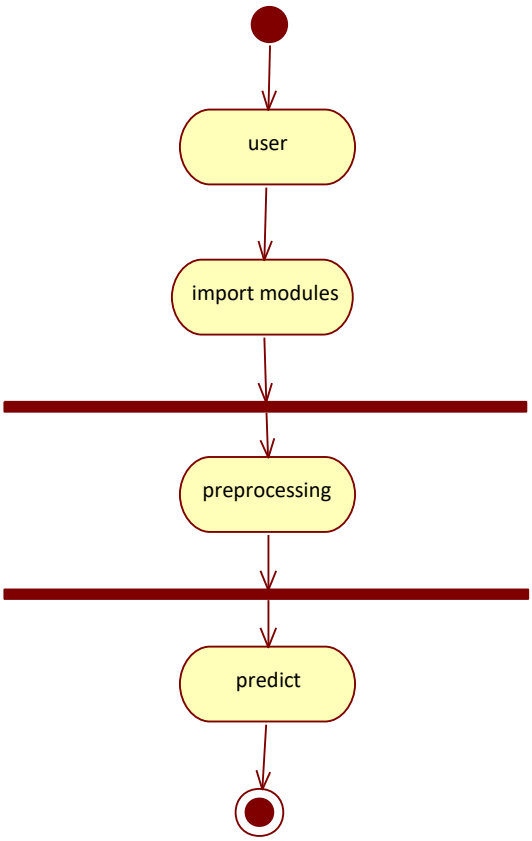
Sun oriented energy has many advantages, yet in addition have Solar-powered energy has many advantages, although their initial venture was more than just offering sunlight-powered chargers, moreover, not everyone really wanted to manage their expenses. Sadly it is the lack of sunlight based chargers; Costs continue to fall, and the coming is great. Sun-powered chargers are currently moderately high; In any case, new government projects and Innovations are making them less expensive. Although photovoltaic cells are considered to be a vast source of potential energy production, their low profits and high cost prevent them from being widely used. The high starting price prevents them from being used normally. Since photovoltaic cells convert sun-based energy into electrical energy, the measurement of solar-based energy generated each day affects the size of the photovoltaic structure, as well as how much energy solar-based radiation distributes each day. Is. It is affected by variables, for example, region, time and weather conditions. Solar-based radiation is the energy used from the sun to the unit area by electromagnetic radiation over the frequency range of a sun-based cell. How much energy a PV framework creates is corresponding to meteorological boundaries including overcast cover, sun power, and site-explicit circumstances, among other [3]. The sun-powered charger varies for different weather patterns. However, if a hurricane occurs, the energy consumption situation will be very different. Lightning age largely depends on weather patterns, so they predict weather conditions. After that, how much energy is not fully impregnated in the rock by sun-based radiation on a guaranteed day cannot be fully determined by various factors, for example, region, time and weather conditions. We will focus on the problem of creating models that accurately estimate the permanent age in natural light.
Public Weather Service Estimates (NWS). Using recorded NWS gauge information and information generated by Sun based boards, we try different things with the classification of the machine.
Learning methods to promote assessment models.
In AI, SVM plays an important role in simultaneously managing information and maintaining weather patterns. Linking to weather conditions from the photovoltaic energy age, as indicated by the positive position of the photovoltaic energy age. svm Provides probed information for repetitive checks such as characterization and clockwork. By using Hyperplane we can aggregate accurate results from a sunlight based charger taking into account weather conditions. Random woods, on the other hand, are an arrangement process that uses a large number of selected trees to classify information. To produce tree-lined forests with a reliable board estimate of more than a single tree. In addition, randomization is highlighted in the correction of each individual tree. It offers a variety of options, integrating all the selected peaks for different weather conditions for example for summer, storm, winter. Related errors are used to estimate model validity (RRMSE), tilt fault (MBE), batch inside and outside meaning (MAE), root mean square blunder (RMSE), relative MBE (RMBE), average rate batch (MPE) and RMSE. Direct Relapse is a practice-based AI approach that is performed. It performs relapse testing. In light of autonomous factors, regression patterns are consistent with objective expectation. It is used extensively to assess how factors are related. The recurrence patterns differ between the type of connection analyzed between the dependent and

autonomous factors, as well as the amount of free factors used.

Problem Statement

- Based on our introductory discussion, the problem can be summarized with the following research question: • How do time series and machine learning techniques perform in short-term (0-5 hours) forecasting of solar PV energy output?

DFD Daigram



What Is Data Science?

Data science is the domain of study that deals with vast volumes of data using modern tools and techniques to find unseen patterns, derive meaningful information, and make business decisions. Data science uses complex machine learning algorithms to build predictive models.

The data used for analysis can come from many different sources and presented in various formats.

Now that you know what data science is, let's see why data science is essential to today's IT landscape.

The Data Science Lifecycle

Data science's lifecycle consists of five distinct stages, each with its own tasks:

1. Capture: Data Acquisition, Data Entry, Signal Reception, Data Extraction. This stage involves gathering raw structured and unstructured data.
2. Maintain: Data Warehousing, Data Cleansing, Data Staging, Data Processing, Data Architecture. This stage covers taking the raw data and putting it in a form that can be used.
3. Process: Data Mining, Clustering/Classification, Data Modeling, Data Summarization. Data scientists take the prepared data and examine its patterns, ranges, and biases to determine how useful it will be in predictive analysis.
4. Analyze: Exploratory/Confirmatory, Predictive Analysis, Regression, Text Mining, Qualitative Analysis. Here is the real meat of the lifecycle. This stage involves performing the various analyses on the data.
5. Communicate: Data Reporting, Data Visualization, Business Intelligence, Decision Making. In this final step, analysts prepare the analyses in easily readable forms such as charts, graphs, and reports.

Prerequisites for Data Science

Here are some of the technical concepts you should know about before starting to learn what is data science.

1. Machine Learning

Machine learning is the backbone of data science. Data Scientists need to have a solid grasp of ML in addition to basic knowledge of statistics.

2. Modeling

Mathematical models enable you to make quick calculations and predictions based on what you already know about the data. Modeling is also a part of Machine Learning and involves identifying which algorithm is the most suitable to solve a given problem and how to train these models.

3. Statistics

Statistics are at the core of data science. A sturdy handle on statistics can help you extract more intelligence and obtain more meaningful results.

4. Programming

Some level of programming is required to execute a successful data science project. The most common programming languages are Python, and R. Python is especially popular because it’s easy to learn, and it supports multiple libraries for data science and ML.

5. Databases

A capable data scientist needs to understand how databases work, how to manage them, and how to extract data from them.

What Does a Data Scientist Do?

A data scientist analyzes business data to extract meaningful insights. In other words, a data scientist solves business problems through a series of steps, including:

- Before tackling the data collection and analysis, the data scientist determines the problem by asking the right questions and gaining understanding.
- The data scientist then determines the correct set of variables and data sets.
- The data scientist gathers structured and unstructured data from many disparate sources—enterprise data, public data, etc.
- Once the data is collected, the data scientist processes the raw data and converts it into a format suitable for analysis. This involves cleaning and validating the data to guarantee uniformity, completeness, and accuracy.
- After the data has been rendered into a usable form, it's fed into the analytic system—ML algorithm or a statistical model. This is where the data scientists analyze and identify patterns and trends.
- When the data has been completely rendered, the data scientist interprets the data to find opportunities and solutions.
- The data scientists finish the task by preparing the results and insights to share with the appropriate stakeholders and communicating the results.

Now we should be aware of some machine learning algorithms which are beneficial in understanding data science clearly.

Why Become a Data Scientist?

According to Glassdoor and Forbes, demand for data scientists will increase by 28 percent by 2026, which speaks of the profession's durability and longevity, so if you want a secure career, data science offers you that chance.

So, if you're looking for an exciting career that offers stability and generous compensation, then look no further!

Where Do You Fit in Data Science?

Data science offers you the opportunity to focus on and specialize in one aspect of the field. Here's a sample of different ways you can fit into this exciting, fast-growing field.

Data Scientist

- Job role: Determine what the problem is, what questions need answers, and where to find the data. Also, they mine, clean, and present the relevant data.
- Skills needed: Programming skills (SAS, R, Python), storytelling and data visualization, statistical and mathematical skills, knowledge of Hadoop, SQL, and Machine Learning.

Data Analyst

- Job role: Analysts bridge the gap between the data scientists and the business analysts, organizing and analyzing data to answer the questions the organization poses. They take the technical analyses and turn them into qualitative action items.
- Skills needed: Statistical and mathematical skills, programming skills (SAS, R, Python), plus experience in data wrangling and data visualization.

Data Engineer

- Job role: Data engineers focus on developing, deploying, managing, and optimizing the organization's data infrastructure and data pipelines. Engineers support data scientists by helping to transfer and transform data for queries.
- Skills needed: NoSQL databases (e.g., MongoDB, Cassandra DB), programming languages such as Java and Scala, and frameworks (Apache Hadoop).

Data Science Tools

The data science profession is challenging, but fortunately, there are plenty of tools available to help the data scientist succeed at their job.

- Data Analysis: SAS, Jupyter, R Studio, MATLAB, Excel, RapidMiner
- Data Warehousing: Informatica/ Talend, AWS Redshift
- Data Visualization: Jupyter, Tableau, Cognos, RAW

- Machine Learning: Spark MLib, Mahout, Azure ML studio

The Basic Skills You Need to Become a Data Scientist

- Mathematical Expertise: There is a commonly circulated meme about grownups realizing that studying algebra was useless because there are no opportunities to use it in everyday life. Surprise! Data scientists need to understand linear algebra, as well as quantitative techniques.

- A Strong Business Acumen: Data scientists are supposed to derive information that is useful to businesses and share it with the appropriate individuals and teams. So, data scientists need to have a solid business understanding so they can have the correct perspective when making these determinations.

- Technology Skills: Data scientists work with sophisticated tools and complex algorithms. They also may be called on to code and develop solutions prototypes rapidly. These expectations mean the data scientist should have proficiency in languages like SQL, R, Python, and SAS, and occasionally in Java, Scala, and Julia.

- Project Management: Data scientists must oversee projects that rely heavily on the data they collect and process. It’s up to the data scientists to ensure that things are moving forward and everyone is communicating with each other.

- In computing, data is information that has been translated into a form that is efficient for movement or processing. Relative to today's computers and transmission media, data is information converted into binary digital form. It is acceptable for data to be used as a singular subject or a plural subject. Raw data is a term used to describe data in its most basic digital format.

- The concept of data in the context of computing has its roots in the work of Claude Shannon, an American mathematician known as the father of information theory. He ushered in binary digital concepts based on applying two-value Boolean logic to electronic circuits. Binary digit formats underlie the CPUs, semiconductor memories and disk drives, as well as many of the peripheral devices common in computing today. Early computer input for both control and data took the form of punch cards, followed by magnetic tape and the hard disk.

- Early on, data's importance in business computing became apparent by the popularity of the terms "data processing" and "electronic data processing," which, for a time, came to encompass the full gamut of what is now known as information technology. Over the history of corporate computing, specialization occurred, and a distinct data profession emerged

along with growth of corporate data processing.

How data is stored

- Computers represent data, including video, images, sounds and text, as binary values using patterns of just two numbers: 1 and 0. A bit is the smallest unit of data, and represents just a single value. A byte is eight binary digits long. Storage and memory is measured in megabytes and gigabytes.

- The units of data measurement continue to grow as the amount of data collected and stored grows. The relatively new term "brontobyte," for example, is data storage that is equal to 10 to the 27th power of bytes.

- Data can be stored in file formats, as in mainframe systems using ISAM and VSAM. Other file formats for data storage, conversion and processing include comma-separated values. These formats continued to find uses across a variety of machine types, even as more structured-data-oriented approaches gained footing in corporate computing.

- Greater specialization developed as database, database management system and then relational database technology arose to organize information.

Types of data

Growth of the web and smartphones over the past decade led to a surge in digital data creation. Data now includes text, audio and video information, as well as log and web activity records. Much of that is unstructured data.

The term big data has been used to describe data in the petabyte range or larger. A shorthand take depicts big data with 3Vs -- volume, variety and velocity. As web-based e-commerce has spread, big data-driven business models have evolved which treat data as an asset in itself. Such trends have also spawned greater preoccupation with the social uses of data and data privacy.

Data has meaning beyond its use in computing applications oriented toward data processing. For example, in electronic component interconnection and network communication, the term data is often distinguished from "control information," "control bits," and similar terms to identify the main content of a transmission unit. Moreover, in science, the term data is used to describe a gathered body of facts. That is also the case in fields such as finance, marketing, demographics and health.

Data management and use

With the proliferation of data in organizations, added emphasis has been placed on ensuring data quality by reducing duplication and guaranteeing the most accurate. The many steps involved with modern data management include data cleansing, as well as extract, transform and load (ETL) processes for integrating data. Data for processing has come to be complemented by metadata, sometimes referred to as "data about data," that helps administrators and users understand database and other data. Analytics that combine structured and unstructured data have become useful, as organizations seek to capitalize on such information. Systems for such analytics increasingly strive for real- time performance, so they are built to handle incoming data consumed at high ingestion rates, and to process data streams for immediate use in operations.Over time, the idea of the database for operations and transactions has been extended to the database for reporting and predictive data analytics. A chief example is the data warehouse, which is optimized to process questions about operations for business analysts and business leaders. Increasing emphasis on finding patterns and predicting business outcomes has led to the development of data mining techniques.

Data professionals

The database administrator profession is an offshoot of IT. These database experts work on designing, tuning and maintaining the database.The data profession took firm root as the relational database management system (RDBMS) gained wide use in corporations, beginning in the 1980s. The relational database's rise was enabled in part by the Structured Query Language (SQL). Later, non-SQL databases, known as NoSQL databases, arose as an alternative to established RDBMSes.

Today, companies employ data management professionals or assign workers the role of data stewardship, which involves carrying out data usage and security policies as outlined in data governance initiatives. A distinct title -- the data scientist -- has appeared to describe professionals focused on data mining and analysis. The benefit of presenting data science in an evocative manner has even given rise to the data artist; that is, an individual adept at graphing and visualizing data in creative ways.

What is data cleaning?

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities for data to be duplicated or mislabeled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct. There is no one absolute way to prescribe the exact steps in the data cleaning process because the processes will vary from dataset to dataset. But it is crucial to establish a template for your data cleaning process so you know you are doing it the right way every time.

What is the difference between data cleaning and data transformation?

Data cleaning is the process that removes data that does not belong in your dataset. Data transformation is the process of converting data from one format or structure into another. Transformation processes can also be referred to as data wrangling, or data munging, transforming and mapping data from one "raw" data form into another format for warehousing and analyzing. This article focuses on the processes of cleaning that data.

How do you clean data?

While the techniques used for data cleaning may vary according to the types of data your company stores, you can follow these basic steps to map out a framework for your organization.

Step 1: Remove duplicate or irrelevant observations

Remove unwanted observations from your dataset, including duplicate observations or irrelevant observations. Duplicate observations will happen most often during data collection. When you combine data sets from multiple places, scrape data, or receive data from clients or multiple departments, there are opportunities to create duplicate data. De-duplication is one of the largest areas to be considered in this process. Irrelevant observations are when you notice observations that do not fit into the specific problem you are trying to analyze. For example, if you want to analyze data regarding millennial customers, but your dataset includes older generations, you might remove

those irrelevant observations. This can make analysis more efficient and minimize distraction from your primary target—as well as creating a more manageable and more performant dataset.

Step 2: Fix structural errors

Structural errors are when you measure or transfer data and notice strange naming conventions, typos, or incorrect capitalization. These inconsistencies can cause mislabeled categories or classes. For example, you may find “N/A” and “Not Applicable” both appear, but they should be analyzed as the same category.

Step 3: Filter unwanted outliers

Often, there will be one-off observations where, at a glance, they do not appear to fit within the data you are analyzing. If you have a legitimate reason to remove an outlier, like improper data-entry, doing so will help the performance of the data you are working with. However, sometimes it is the appearance of an outlier that will prove a theory you are working on. Remember: just because an outlier exists, doesn’t mean it is incorrect. This step is needed to determine the validity of that number. If an outlier proves to be irrelevant for analysis or is a mistake, consider removing it.

Step 4: Handle missing data

You can’t ignore missing data because many algorithms will not accept missing values. There are a couple of ways to deal with missing data. Neither is optimal, but both can be considered.

1. As a first option, you can drop observations that have missing values, but doing this will drop or lose information, so be mindful of this before you remove it.
2. As a second option, you can input missing values based on other observations; again, there is an opportunity to lose integrity of the data because you may be operating from assumptions and not actual observations.
3. As a third option, you might alter the way the data is used to effectively navigate null values.

Step 5: Validate and QA

At the end of the data cleaning process, you should be able to answer these questions as a part of basic validation:

- Does the data make sense?
- Does the data follow the appropriate rules for its field?
- Does it prove or disprove your working theory, or bring any insight to light?
- Can you find trends in the data to help you form your next theory?
- If not, is that because of a data quality issue?

False conclusions because of incorrect or “dirty” data can inform poor business strategy and decision-making. False conclusions can lead to an embarrassing moment in a reporting meeting when you realize your data doesn’t stand up to scrutiny. Before you get there, it is important to create a culture of quality data in your organization. To do this, you should document the tools you might use to create this culture and what data quality means to you.

Components of quality data

Determining the quality of data requires an examination of its characteristics, then weighing those characteristics according to what is most important to your organization and the application(s) for which they will be used.

5 characteristics of quality data

1. Validity. The degree to which your data conforms to defined business rules or constraints.
2. Accuracy. Ensure your data is close to the true values.
3. Completeness. The degree to which all required data is known.

4. Consistency. Ensure your data is consistent within the same dataset and/or across multiple data sets.
5. Uniformity. The degree to which the data is specified using the same unit of measure.

Benefits of data cleaning

Having clean data will ultimately increase overall productivity and allow for the highest quality information in your decision-making. Benefits include:

- Removal of errors when multiple sources of data are at play.
- Fewer errors make for happier clients and less-frustrated employees.
- Ability to map the different functions and what your data is intended to do.
- Monitoring errors and better reporting to see where errors are coming from, making it easier to fix incorrect or corrupt data for future applications.
- Using tools for data cleaning will make for more efficient business practices and quicker decision-making.

Data cleaning tools and software for efficiency

Software like Tableau Prep can help you drive a quality data culture by providing visual and direct ways to combine and clean your data. Tableau Prep has two products: Tableau Prep Builder for building your data flows and Tableau Prep Conductor for scheduling, monitoring, and managing flows across your organization. Using a data scrubbing tool can save a database administrator a significant amount of time by helping analysts or administrators start their analyses faster and have more confidence in the data. Understanding data quality and the tools you need to create, manage, and transform data is an important step toward making efficient and effective business decisions. This crucial process will further develop a data culture in your organization. To see how Tableau Prep can impact your organization, read about how marketing agency Tinuiti centralized 100-plus data sources in Tableau Prep and scaled their marketing analytics for 500 clients.

What is Data Exploration?

DATA EXPLORATION DEFINITION: Data exploration refers to the initial step in data analysis in which data analysts use data visualization and statistical techniques to describe dataset characterizations, such as size, quantity, and accuracy, in order to better understand the nature of the data.

Data exploration techniques include both manual analysis and automated data exploration software solutions that visually explore and identify relationships between different data.

variables, the structure of the dataset, the presence of outliers, and the distribution of data values in order to reveal patterns and points of interest, enabling data analysts to gain greater insight into the raw data.

Data is often gathered in large, unstructured volumes from various sources and data analysts must first understand and develop a comprehensive view of the data before extracting relevant data for further analysis, such as univariate, bivariate, multivariate, and principal components analysis.

Data Exploration Tools

Manual data exploration methods entail either writing scripts to analyze raw data or manually filtering data into spreadsheets. Automated data exploration tools, such as data visualization software, help data scientists easily monitor data sources and perform big data exploration on otherwise overwhelmingly large datasets. Graphical displays of data, such as bar charts and scatter plots, are valuable tools in visual data exploration.A popular tool for manual data exploration is Microsoft Excel spreadsheets, which can be used to create basic charts for data exploration, to view raw data, and to identify the correlation between variables. To identify the correlation between two continuous variables in Excel, use the function CORREL() to return the correlation. To identify the correlation between two categorical variables in Excel, the two-way table method, the stacked column chart method, and the chi-square test are effective. There is a wide variety of proprietary automated data exploration solutions, including business intelligence tools, data visualization software, data preparation software vendors, and data exploration platforms. There are also open source data exploration tools that include regression capabilities and visualization features, which can help businesses integrate diverse data sources to enable faster data exploration. Most data analytics software includes data visualization tools.

Why is Data Exploration Important?

Humans process visual data better than numerical data, therefore it is extremely challenging for data scientists and data analysts to assign meaning to thousands of rows and columns of data points and communicate that meaning without any visual components.

Data visualization in data exploration leverages familiar visual cues such as shapes, dimensions, colors, lines, points, and angles so that data analysts can effectively visualize and define the metadata, and then perform data cleansing. Performing the initial step of data exploration enables data analysts to better understand and visually identify anomalies and relationships that might otherwise go undetected.

What is Exploratory Data Analysis?

Exploratory Data Analysis (EDA), similar to data exploration, is a statistical technique to analyze data sets for their broad characteristics. Visualization tools for exploratory data analysis such as OmniSci's Immerse platform enable interactivity with raw data sets, giving analysts increased visibility into the patterns and relationships within the data.

Data Exploration in Machine Learning

A Machine Learning project is as good as the foundation of data on which it is built. In order to perform well, machine learning data exploration models must ingest large quantities of data, and model accuracy will suffer if that data is not thoroughly explored first. Data exploration steps to follow before building a machine learning model include:

- Variable identification: define each variable and its role in the dataset
- Univariate analysis: for continuous variables, build box plots or histograms for each variable independently; for categorical variables, build bar charts to show the frequencies
- Bi-variable analysis - determine the interaction between variables by building visualization tools
- ~Continuous and Continuous: scatter plots
- ~Categorical and Categorical: stacked column chart
- ~Categorical and Continuous: boxplots combined with swarmplots
- Detect and treat missing values
- Detect and treat outliers

The ultimate goal of data exploration machine learning is to provide data insights that will inspire subsequent feature engineering and the model-building process. Feature engineering facilitates the machine learning process and increases the predictive power of machine learning algorithms by creating features from raw data.

Interactive Data Exploration

Advanced visualization techniques are employed throughout a variety of disciplines to empower users to visualize patterns and gain insight from complex data flows, and make subsequent data-driven decisions. Industries from engineering to medicine to education are learning how to do data exploration.

In big data exploration tools, interactivity is an important component in the perception of data exploration visual technologies and the dissemination of insights. The manner in which users perceive and interact with visualizations can heavily influence their understanding of the data as well as the value they place on the visualization system in general.

Interactive data exploration emphasizes the importance of collaborative work and facilitates human interaction with the integration of advanced interaction and visualization technologies. Accelerated multimodal interaction platforms equipped with graphical user interfaces that prioritize human-to-human properties facilitate big data exploration through visual analytics, accelerate the sharing of opinions, remove the data bottleneck of individual analysis, and reduce discovery time.

What is Data Visualization?

Data visualization is a graphical representation of quantitative information and data by using visual elements like graphs, charts, and maps.

Data visualization convert large and small data sets into visuals, which is easy to understand and process for humans.

Data visualization tools provide accessible ways to understand outliers, patterns, and trends in the data.

In the world of Big Data, the data visualization tools and technologies are required to analyze vast amounts of information.

Data visualizations are common in your everyday life, but they always appear in the form of graphs and charts. The combination of multiple visualizations and bits of information are still referred to as Infographics.

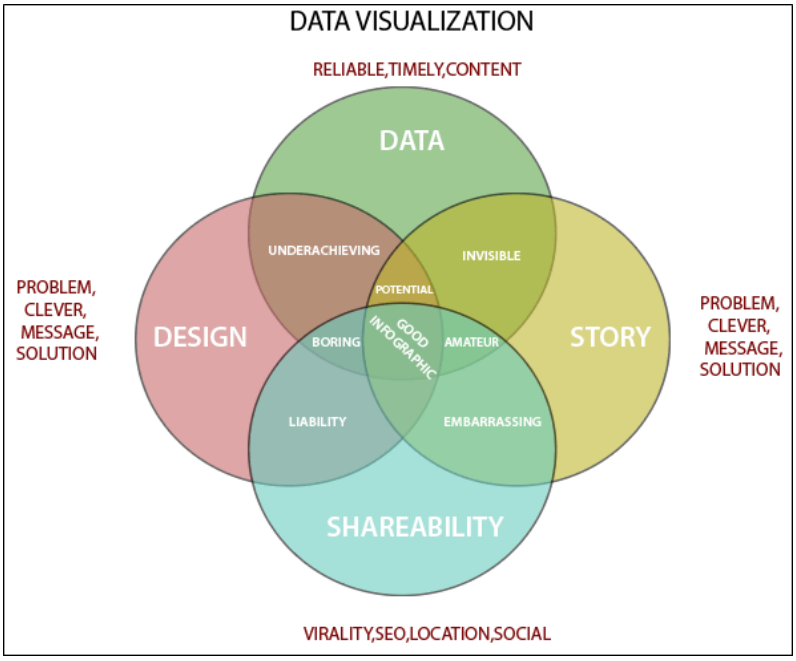
Data visualizations are used to discover unknown facts and trends. You can see visualizations in the form of line charts to display change over time. Bar and column charts are useful for observing relationships and making comparisons. A pie chart is a great way to show parts-of-a-whole. And maps are the best way to share geographical data visually.

Today's data visualization tools go beyond the charts and graphs used in the Microsoft Excel spreadsheet, which displays the data in more sophisticated ways such as dials and gauges, geographic maps, heat maps, pie chart, and fever chart.

What makes Data Visualization Effective?

Effective data visualization are created by communication, data science, and design collide. Data visualizations did right key insights into complicated data sets into meaningful and natural.

American statistician and Yale professor Edward Tufte believe useful data visualizations consist of ?complex ideas communicated with clarity, precision, and efficiency.



Importance of Data Visualization

Data visualization is important because of the processing of information in human brains. Using graphs and charts to visualize a large amount of the complex data sets is more comfortable in comparison to studying the spreadsheet and reports.

Data visualization is an easy and quick way to convey concepts universally. You can experiment with a different outline by making a slight adjustment.

Why Use Data Visualization?

1. To make easier in understand and remember.
2. To discover unknown facts, outliers, and trends.
3. To visualize relationships and patterns quickly.
4. To ask a better question and make better decisions.
5. To competitive analyze.
6. To improve insights.

What is Tableau?

Tableau is a powerful and fastest growing data visualization tool used in the Business Intelligence Industry. It helps in simplifying raw data in a very easily understandable format. Tableau helps create the data that can be understood by professionals at any level in an

organization. It also allows non-technical users to create customized dashboards.
Data analysis is very fast with Tableau tool and the visualizations created are in the form of dashboards and worksheets.

The best features of Tableau software are

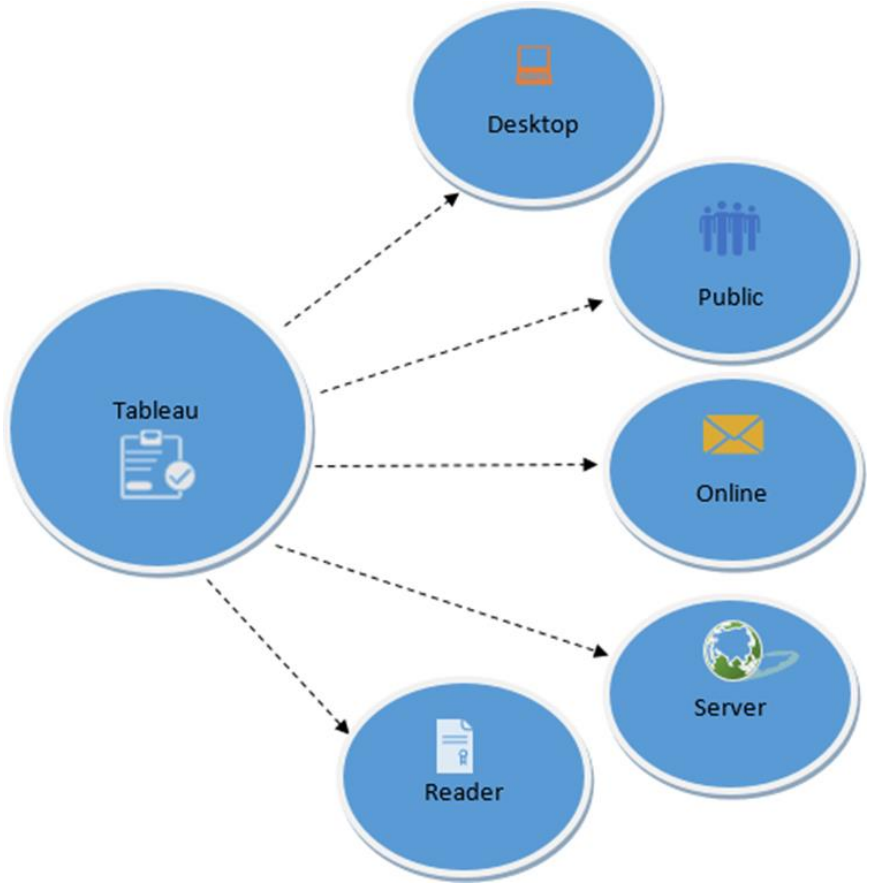
- Data Blending
- Real time analysis
- Collaboration of data

The great thing about Tableau software is that it doesn't require any technical or any kind of programming skills to operate. The tool has garnered interest among the people from all sectors such as business, researchers, different industries, etc.

Tableau Product Suite

The Tableau Product Suite consists of

- Tableau Desktop
- Tableau Public
- Tableau Online
- Tableau Server
- Tableau Reader



For a clear understanding, data analytics in Tableau tool can be classified into two section.

1. Developer Tools: The Tableau tools that are used for development such as the creation of dashboards, charts, report generation, visualization fall into this category. The Tableau products, under this category, are the Tableau Desktop and the Tableau Public.
2. Sharing Tools: As the name suggests, the purpose of these Tableau products is sharing the visualizations, reports, dashboards that were created using the developer tools. Products that fall into this category are Tableau Online, Server, and Reader.

Tableau Desktop

Tableau Desktop has a rich feature set and allows you to code and customize reports. Right from creating the charts, reports, to blending them all together to form a dashboard, all the necessary work is created in Tableau Desktop.

For live data analysis, Tableau Desktop provides connectivity to Data Warehouse, as well as other various types of files. The workbooks and the dashboards created here can be either shared locally or publicly.

Based on the connectivity to the data sources and publishing option, Tableau Desktop is

classified into

- Tableau Desktop Personal: The development features are similar to Tableau Desktop. Personal version keeps the workbook private, and the access is limited. The workbooks cannot be published online. Therefore, it should be distributed either Offline or in Tableau Public.

- Tableau Desktop Professional: It is pretty much similar to Tableau Desktop. The difference is that the work created in the Tableau Desktop can be published online or in Tableau Server. Also, in Professional version, there is full access to all sorts of the datatype. It is best suitable for those who wish to publish their work in Tableau Server.

Tableau Public

It is Tableau version specially build for the cost-effective users. By the word "Public," it means that the workbooks created cannot be saved locally; in turn, it should be saved to the Tableau's public cloud which can be viewed and accessed by anyone.There is no privacy to the files saved to the cloud since anyone can download and access the same. This version is the best for the individuals who want to learn Tableau and for the ones who want to share their data with the general public.

Tableau Server

The software is specifically used to share the workbooks, visualizations that are created in the Tableau Desktop application across the organization. To share dashboards in the Tableau Server, you must first publish your work in the Tableau Desktop. Once the work has been uploaded to the server, it will be accessible only to the licensed users.However, It's not necessary that the licensed users need to have the Tableau Server installed on their machine. They just require the login credentials with which they can check reports via a web browser. The security is high in Tableau server, and it is much suited for quick and effective sharing of data in an organization.The admin of the organization will always have full control over the server. The hardware and the software are maintained by the organization.

Tableau Online

As the name suggests, it is an online sharing tool of Tableau. Its functionalities are similar to Tableau Server, but the data is stored on servers hosted in the cloud which are maintained by the Tableau group.There is no storage limit on the data that can be published in the Tableau Online. Tableau Online creates a direct link to over 40 data sources that are hosted in the cloud such as the MySQL, Hive, Amazon Aurora, Spark SQL and many more.To publish, both Tableau Online and Server require the workbooks created by Tableau Desktop. Data that is streamed from the web applications say Google Analytics, Salesforce.com are also supported by Tableau Server and Tableau Online.

Tableau Reader

Tableau Reader is a free tool which allows you to view the workbooks and visualizations created using Tableau Desktop or Tableau Public. The data can be filtered but editing and modifications are restricted. The security level is zero in Tableau Reader as anyone who gets the workbook can view it using Tableau Reader.If you want to share the dashboards that you have created, the receiver should have Tableau Reader to view the document.

How does Tableau work?

Tableau connects and extracts the data stored in various places. It can pull data from any platform imaginable. A simple database such as an excel, pdf, to a complex database like Oracle, a database in the cloud such as Amazon webs services, Microsoft Azure SQL database, Google Cloud SQL and various other data sources can be extracted by Tableau.When Tableau is launched, ready data connectors are available which allows you to connect to any database. Depending on the version of Tableau that you have purchased the number of data connectors supported by Tableau will vary.The pulled data can be either connected live or extracted to the Tableau's data engine, Tableau Desktop. This is where the Data analyst, data engineer work with the data that was pulled up and develop visualizations. The created dashboards are shared with the users as a static file. The users who receive the dashboards views the file using Tableau Reader.The data from the Tableau Desktop can be published to the Tableau server. This is an enterprise platform where collaboration, distribution, governance, security model, automation features are supported. With the Tableau server, the end users have a better experience in accessing the files from all locations be it a desktop, mobile or email.

Tableau Uses

Following are the main uses and applications of Tableau:

- Business Intelligence
- Data Visualization
- Data Collaboration
- Data Blending
- Real-time data analysis
- Query translation into visualization
- To import large size of data
- To create no-code data queries
- To manage large size metadata

What Is Machine Learning?

Machine learning is the study of using algorithms and data that allow computers to perform tasks without instructions or input from human users. Different experts have created their own definitions to describe machine learning, but at its core, machine learning is characterized by

computers performing autonomous improvement using real-world examples and data to do so, rather than a continual human input.

AI vs Machine Learning

At first glance, machine learning seems to have an almost interchangeable definition with "artificial intelligence" (AI). After all, Merriam-Webster defines AI as "a branch of computer science dealing with the simulation of intelligent behavior in computers; the capability of a machine to imitate intelligent human behavior." However, upon closer inspection, it's clear that these two terms refer to entirely distinct things.

AI encompasses many different processes and practices, including things like neural networks and image processing; machine learning is one of these subsets of AI. So while AI can take many different forms, such as a self-driving car or a digital assistant like Siri or Alexa, machine learning describes a particular aspect of AI function: computers learning autonomously.

How Do Computers Learn?

Put simply, a human user puts data into the computer, which then analyzes the data and looks for patterns in it. When the computer finds a pattern, it adjusts how it processes or manages data to reflect what it found. After the computer finds enough patterns, it can begin to make predictions. Generally, if a larger amount of training data is put in, the computer will become more accurate, faster.

In practice, machine learning is more complicated than that and there are two main forms: supervised and unsupervised learning. Each form requires large amounts of input data to train the machine learning algorithm, but they differ in how they interact with the data.

- Supervised learning: For this form of machine learning, the training data is categorized or labeled with the "correct" outcome. The computer is then given unlabeled information to process. It will compare the new data with the old and then determine the outcome based on the previous example. Supervised learning is especially well- suited to classifying items into different categories and regression when the output is a real value such as "dollars" or "pounds." This is the most common form of machine learning and it's generally more reliable than unsupervised learning.

- Unsupervised learning: In unsupervised learning, the data is not labeled before being put into the algorithm. The machine then attempts to find patterns in the data on its own. Unsupervised learning works particularly well when identifying similarities in groups and clustering them together, as well as identifying anomalies or abnormalities in data. It's more difficult to measure the accuracy of an unsupervised learning algorithm since there are no training data to compare it to, but it can still provide valuable results and insights.

How Is Machine Learning Used?

Machine learning has a variety of applications in modern life. We've already found uses for it in industries ranging from healthcare to cybersecurity, and as this technology continues to develop, we'll likely find many more. Other common uses of machine learning include:

Machine Learning in Marketing

Machine learning may prove to be especially useful in marketing because of its ability to identify patterns in data that humans might not otherwise notice. This can be particularly helpful when looking at user behavior; machine learning algorithms can analyze massive amounts of user data from multiple sources, such as social media pages and interactions with a website, to better determine how marketers and brands can engage with customers.Among its many uses in marketing, machine learning can help marketers decide what ads to display to certain customers, identify the best time to send out individual offers or incentives, and even help improve the overall customer experience.

Search Engines

Search engines typically use algorithms to organize relevant results when users input a query. Machine learning is often used to update and refine these algorithms to improve the quality of results given to users. It can also be used to better understand searchers' queries, classify users to make searches more personalized, and determine the best rate for crawling different websites or data sets.

Machine Learning in Weather Prediction and Climate Science

Machine learning also has uses in meteorology and climatology, as it can be used to analyze and predict weather patterns.

. Websites and apps that use APIs to scrape weather data are useful for consumers, but that kind of technology also empowers weather prediction by autonomous computer systems, providing them with ever more information. Further, machine learning can also play a role in obtaining and analyzing larger climate patterns, which can aid in the development of more accurate and detailed climate prediction models.

Machine Vision: Recognizing Text, Images, and Faces

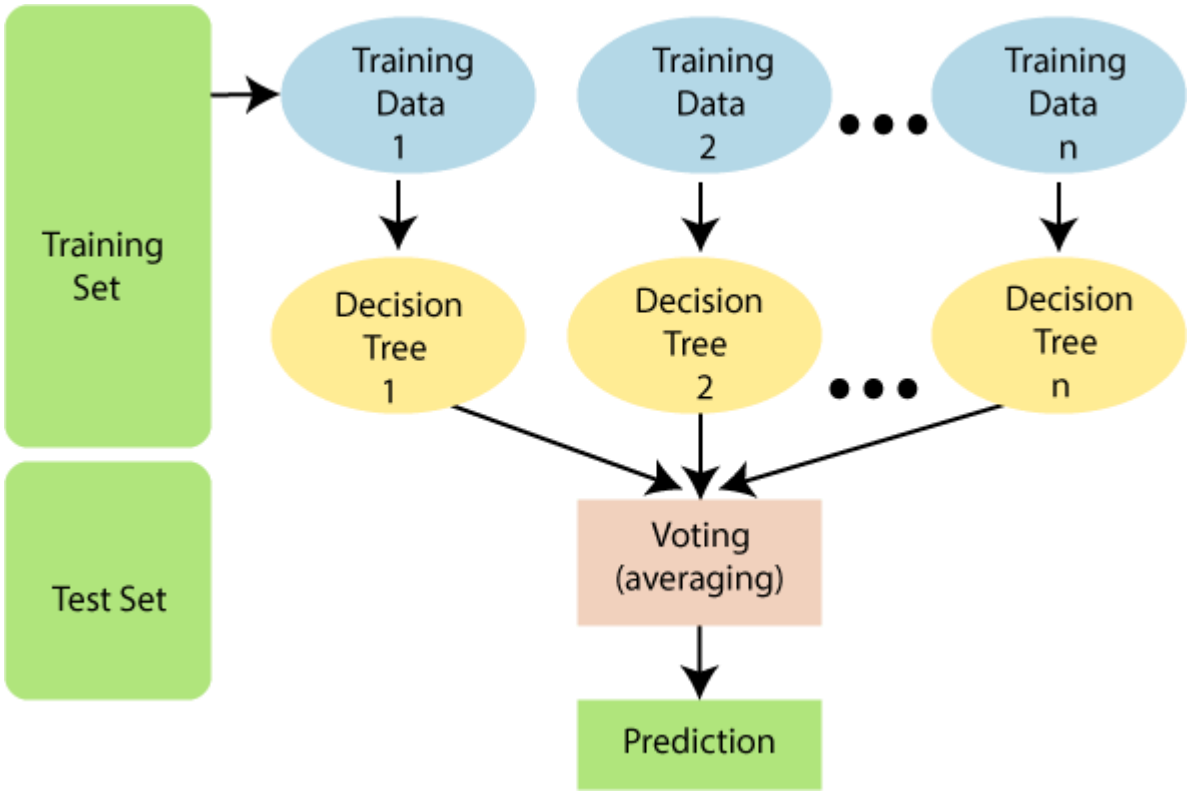
A specialized type of machine learning, machine or computer vision is a computer's ability to "see," inspect and analyze images or videos. By analyzing images and converting visual elements into data, machine vision can recognize text in an image, identify faces, and even improve or generate images. High-powered computers aren't the only devices capable of this kind of machine learning; using the right kind of API with optical character recognition, you can even turn your cell phone into an image reader, pull text out of physical images, even perform live translation of written text. Machine vision is being used in a variety of industries for a number of purposes, including social media and law enforcement.

Machine learning is an exciting new field in the world of AI, and though we've made great strides in developing this technology, it has many applications that have yet to be explored. As computer scientists continue to refine the capabilities of machine learning, we'll determine even more ways in which it can be used, and it may become even more important to daily life than it already is.

Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of COMBINING MULTIPLE CLASSIFIERS TO SOLVE A COMPLEX PROBLEM AND TO IMPROVE THE PERFORMANCE OF THE MODEL.

As the name suggests, *"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."* Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting



Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm:

<="" li="">

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:
Step-1: Select random K data points from the training set.
Step-2: Build the decision trees associated with the selected data points (Subsets).
Step-3: Choose the number N for decision trees that you want to build.
Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

The working of the algorithm can be better understood by the below example:

Example: Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision.

Applications of Random Forest

There are mainly four sectors where Random forest mostly used:

1. Banking: Banking sector mostly uses this algorithm for the identification of loan risk.
2. Medicine: With the help of this algorithm, disease trends and risks of the disease can be identified.
3. Land Use: We can identify the areas of similar land use by this algorithm.
4. Marketing: Marketing trends can be identified using this algorithm.

Advantages of Random Forest

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.

- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

Python Implementation of Random Forest Algorithm

Now we will implement the Random Forest Algorithm tree using Python. For this, we will use the same dataset "user_data.csv", which we have used in previous classification models. By using the same dataset, we can compare the Random Forest classifier with other classification models such as Decision tree Classifier,

KNN,
SVM,
Logistic Regression,
etc.

Implementation Steps are given below:

- Data Pre-processing step
- Fitting the Random forest algorithm to the Training set
- Predicting the test result
- Test accuracy of the result (Creation of Confusion matrix)
- Visualizing the test set result.

SPYDER

It is always necessary to have interactive environments to create software applications and this fact becomes very important when you work in the fields of Data Science, engineering, and scientific research. The Python Spyder IDE has been created for the same purpose. In this article, you will be learning how to install and make use of Spyder or the Scientific Python and Development IDE.

Before moving on, let’s take a look at all the topics that are discussed over here:

- What is Python Spyder IDE?
- Features of Spyder
- Python Spyder IDE Installation
- Creating a file/ Starting a Project
- Writing the Code
- Variable Explorer
- File Explorer
- Configuring Spyder
- Help

What is Python Spyder IDE?

Spyder is an open-source cross-platform IDE. The Python Spyder IDE is written completely in Python. It is designed by scientists and is exclusively for scientists, data analysts, and engineers. It is also known as the Scientific Python Development IDE and has a huge set of remarkable features which are discussed below.

Features of Spyder

Some of the remarkable features of Spyder are:

- Customizable Syntax Highlighting
- Availability of breakpoints (debugging and conditional breakpoints)
- Interactive execution which allows you to run line, file, cell, etc.
- Run configurations for working directory selections, command-line options, current/ dedicated/ external console, etc
- Can clear variables automatically (or enter debugging)
- Navigation through cells, functions, blocks, etc can be achieved through the Outline Explorer
- It provides real-time code introspection (The ability to examine what functions, keywords, and classes are, what they are doing and what information they contain)
- Automatic colon insertion after if, while, etc
- Supports all the IPython magic commands
- Inline display for graphics produced using Matplotlib
- Also provides features such as help, file explorer, find files, etc.

About Python

Python is one of those rare languages which can claim to be
both SIMPLE and POWERFUL. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in.

The official introduction to Python is:

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.I will discuss most of these features in more detail in the next section.

Story behind the name

Guido van Rossum, the creator of the Python language, named the language after the BBC show "Monty Python's Flying Circus". He doesn't particularly like snakes that kill animals for food by winding their long bodies around them and crushing them.

Features of Python

Simple

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

Easy to Learn

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

Free and Open Source

Python is an example of a FLOSS (Free/Libre and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

High-level Language

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

Portable

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like Kivy to create games for your computer AND for iPhone, iPad, and Android.

Interpreted

This requires a bit of explanation.A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.Python, on the other hand, does not need compilation to binary. You just RUN the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since
you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

Object Oriented

Python supports procedure-oriented programming as well as object-oriented programming (OOP). In PROCEDURE-ORIENTED languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In OBJECT-ORIENTED languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

Extensible

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

Embeddable

You can embed Python within your C/C++ programs to give SCRIPTING capabilities for your program's users.

Extensive Libraries

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions,documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the BATTERIES INCLUDED philosophy of Python.Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

Broadly, there are 3 types of Machine Learning Algorithms

1. Supervised Learning

How it works: This algorithm consist of a target / outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using these set of variables, we generate a function that map inputs to desired outputs. The training process continues until the model achieves a desired level of accuracy on the training data. Examples of Supervised Learning: Regression, Decision Tree, Random Forest, KNN, Logistic Regression etc.

2. Unsupervised Learning

How it works: In this algorithm, we do not have any target or outcome variable to predict / estimate. It is used for clustering population in different groups, which is widely used for segmenting customers in different groups for specific intervention. Examples of Unsupervised Learning: Apriori algorithm, K-means.

3. Reinforcement Learning:

How it works: Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it trains itself continually using trial and error. This machine learns from past experience and tries to capture the best possible knowledge to make accurate business decisions. Example of Reinforcement Learning: Markov Decision Process

List of Common Machine Learning Algorithms

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

1. Linear Regression
2. Logistic Regression
3. Decision Tree
4. SVM
5. Naive Bayes
6. kNN
7. K-Means
8. Random Forest
9. Dimensionality Reduction Algorithms
10. Gradient Boosting algorithms
    1. GBM
    2. XGBoost
    3. LightGBM
    4. CatBoost

1. Linear Regression

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a *X + b

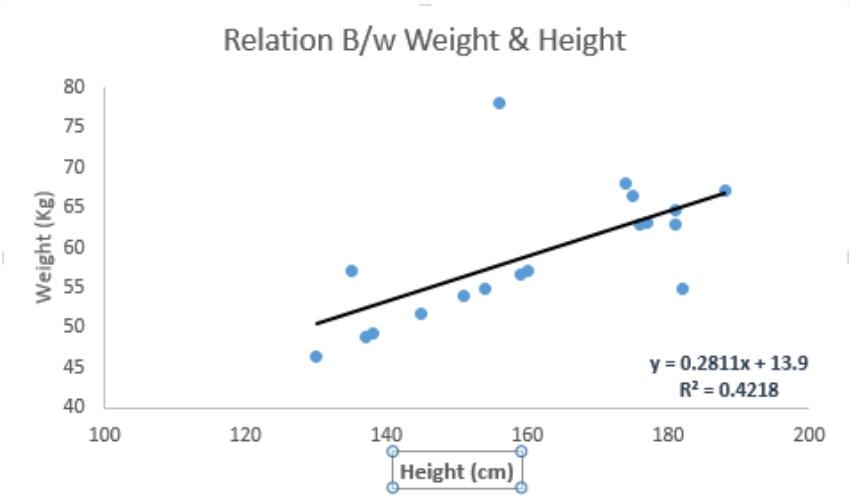
The best way to understand linear regression is to relive this experience of childhood. Let us say, you ask a child in fifth grade to arrange people in his class by increasing order of weight, without asking them their weights! What do you think the child will do? He / she would likely look (visually analyze) at the height and build of people and arrange them using a combination of these visible parameters. This is linear regression in real life! The child has actually figured out that height and build would be correlated to the weight by a relationship, which looks like the equation above.

In this equation:

- Y – Dependent Variable
- a – Slope
- X – Independent variable
- b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line

Look at the below example. Here we have identified the best fit line having linear equation y=0.2811x+13.9. Now using this equation, we can find the weight, knowing the height of a person.

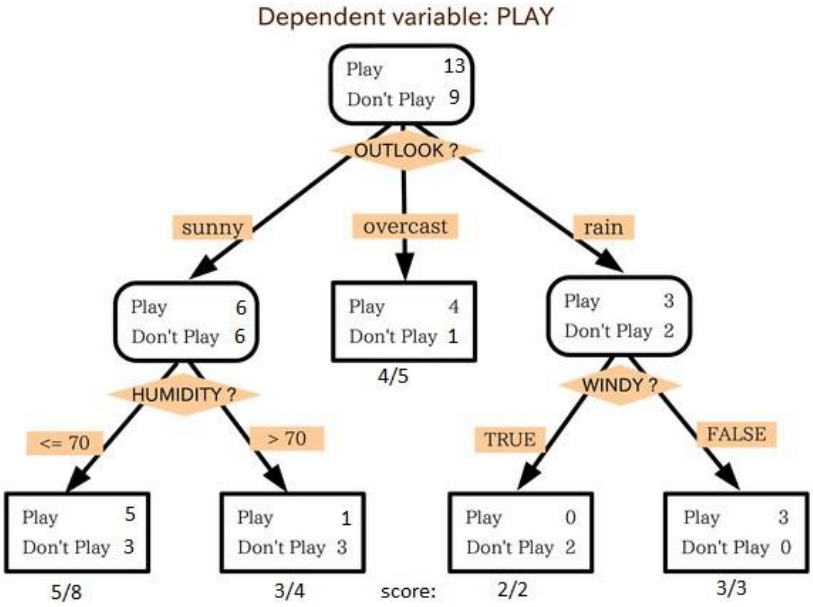


Linear Regression is mainly of two types: Simple Linear Regression and Multiple Linear Regression. Simple Linear Regression is characterized by one independent variable. And, Multiple Linear Regression(as the name suggests) is characterized by multiple (more than 1) independent variables. While finding the best fit line, you can fit a polynomial or curvilinear

regression. And these are known as polynomial or curvilinear regression.

3. Decision Tree

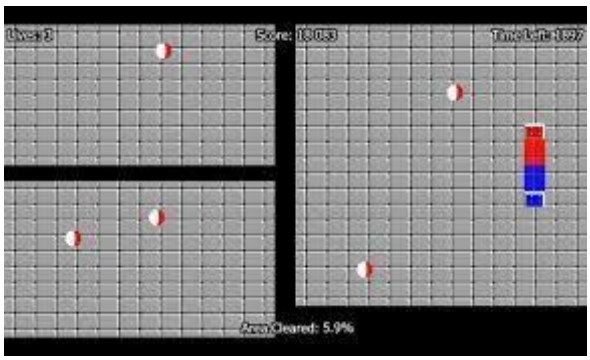
This is one of my favorite algorithm and I use it quite frequently. It is a type of supervised learning algorithm that is mostly used for classification problems. Surprisingly, it works for both categorical and continuous dependent variables. In this algorithm, we split the population into two or more homogeneous sets. This is done based on most significant attributes/ independent variables to make as distinct groups as possible. For more details, you can read: Decision Tree Simplified.



source: statsexchange

In the image above, you can see that population is classified into four different groups based on multiple attributes to identify 'if they will play or not'. To split the population into different heterogeneous groups, it uses various techniques like Gini, Information Gain, Chi-square, entropy.

The best way to understand how decision tree works, is to play Jezzball – a classic game from Microsoft (image below). Essentially, you have a room with moving walls and you need to create walls such that maximum area gets cleared off with out the balls.



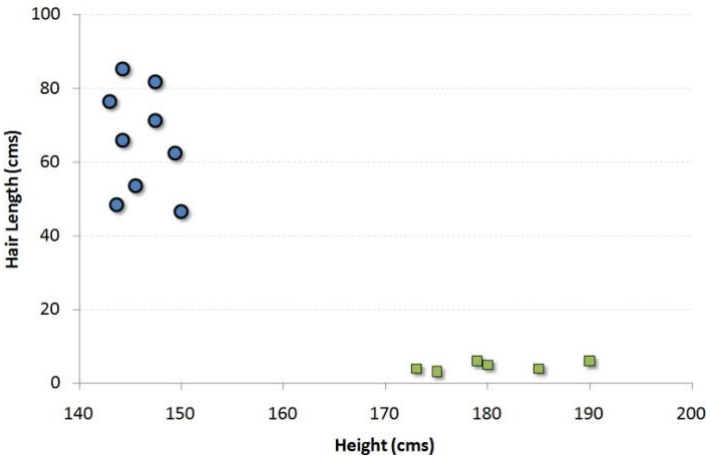
So, every time you split the room with a wall, you are trying to create 2 different populations with in the same room. Decision trees work in very similar fashion by dividing a population in as different groups as possible.

4. SVM (Support Vector Machine)

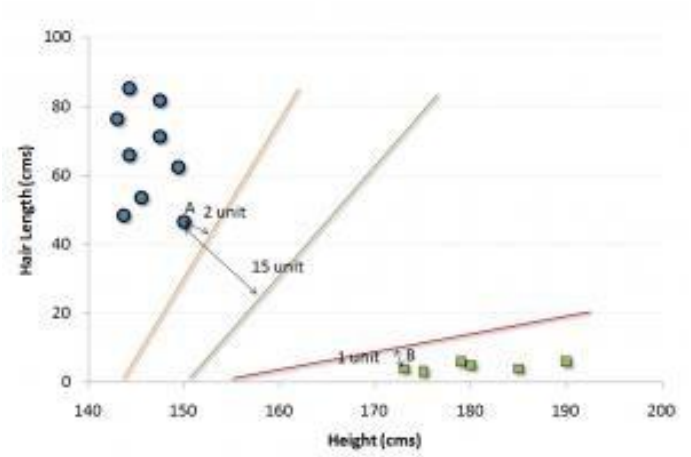
It is a classification method. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.

For example, if we only had two features like Height and Hair length of an individual, we'd first plot these two variables in two dimensional space where each point has two co-ordinates (these co-ordinates are known as Support Vectors)

.



Now, we will find some LINE that splits the data between the two differently classified groups of data. This will be the line such that the distances from the closest point in each of the two groups will be farthest away.



In the example shown above, the line which splits the data into two differently classified groups is the BLACK line, since the two closest points are the farthest apart from the line. This line is our classifier. Then, depending on where the testing data lands on either side of the line, that's what class we can classify the new data as.

More: Simplified Version of Support Vector Machine

Think of this algorithm as playing JezzBall in n-dimensional space. The tweaks in the game are:

- You can draw lines/planes at any angles (rather than just horizontal or vertical as in the classic game)

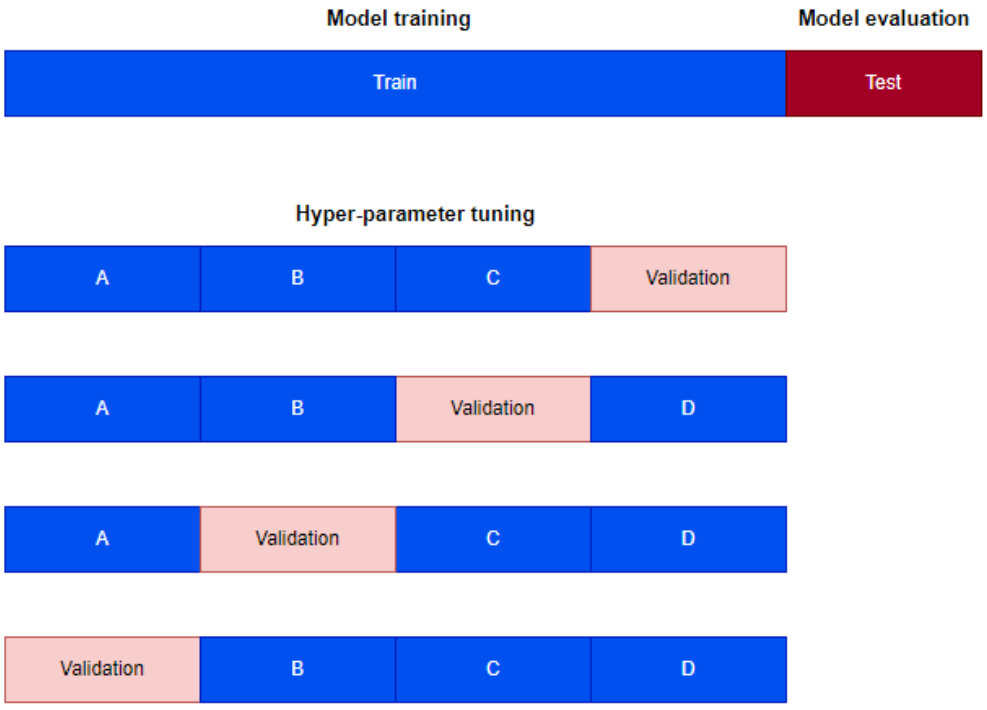
- The objective of the game is to segregate balls of different colors in different rooms.
- And the balls are not moving.

Logistic Regression

Logistic regression works with sigmoid function because the sigmoid function can be used to classify the output that is dependent feature and it uses the probability for classification of the dependent feature. This algorithm works well with less amount of data set because of the use of sigmoid function if value the of sigmoid function is greater than 0.5 the output will 1 if the output the sigmoid function is less than 0.5 then the output is considered as the 0. But this sigmoid function is not suitable for deep learning because the if deep learning when we back tracking from the output to input we have to update the weights to minimize the error in weight update. we have to do differentiation of sigmoid activation function in middle layer neuron then results in the value of 0.25 this will affect the accuracy of the module in deep learning.

Model Stacking

For model stacking process we take four models of machine learning. KNN, DNN, RF, LGBM is combined using stacking in sckit-learn library.
A simple linear KNN regression model was used to meta-student and prepared on 4-overlay cross-approved expectations of the first information highlights alongside the base model.
The stacking register utilizes the cross_val_predict work which returns, for every model in the preparation information, the forecast that was gotten for that model when it was in the approval set. In different premise models these forecasts are utilized as contributions to the meta-student (see Sklearn User Guide). This approach limits the gamble of overfiting.



REQUIREMENTS

Hardware Requirements:

RAM: 4GB and Higher
Processor: Intel i3 and above
Hard Disk: 500GB: Minimum

Software Requirements:

OS: Windows or Linux
Python IDE : python 2.7.x and above
Jupyter IDE
Anaconda 3.x

Setup tools and pip to be installed for

3.6 and above Python Scripting

DIAGRAMS

. UML DIAGRAMS

The Unified Modeling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development. UML offers a standard way to visualize a system's architectural blueprints, including elements such as:

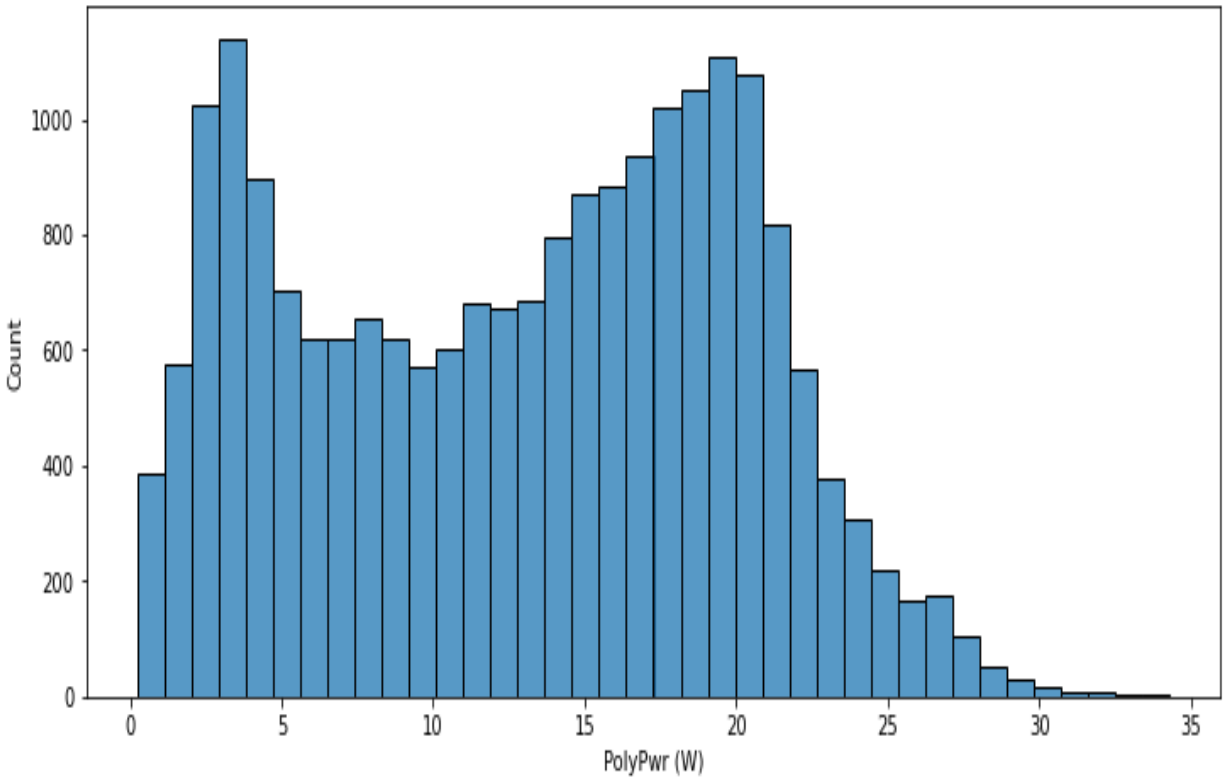
- actors
- business processes
- (logical) components
- activities
- programming language statements
- database schemas, and

Reusable software components

Use Case Diagram

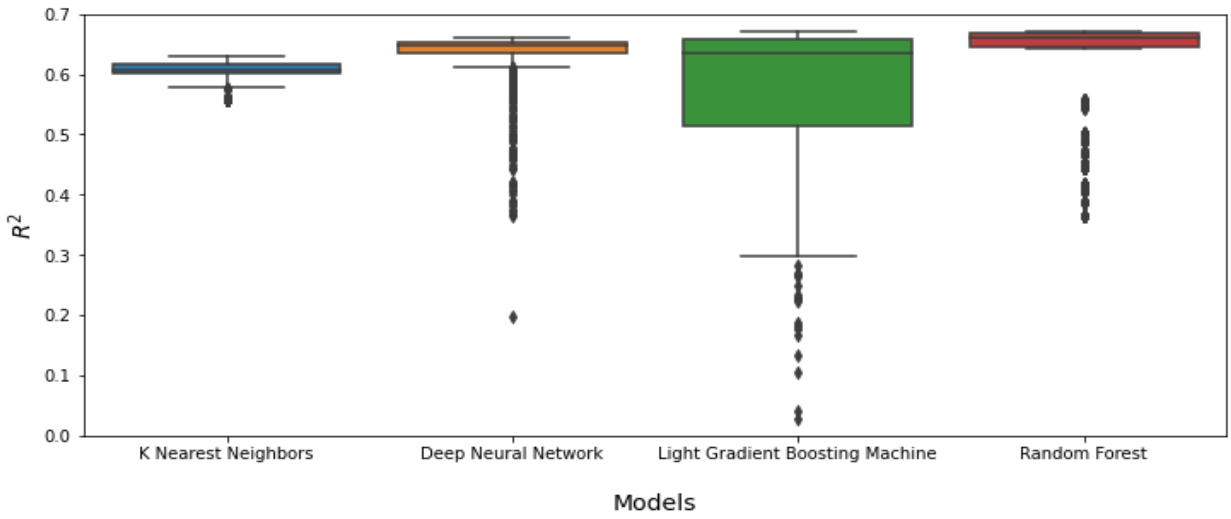
Feature engineering

Feature engineering is the process of select,manipulate and transfarm that raw data into features that can be used for supervised learning. If we want to make our machine learning model good on new tasks, than design and train better features in our dataset is necessary.

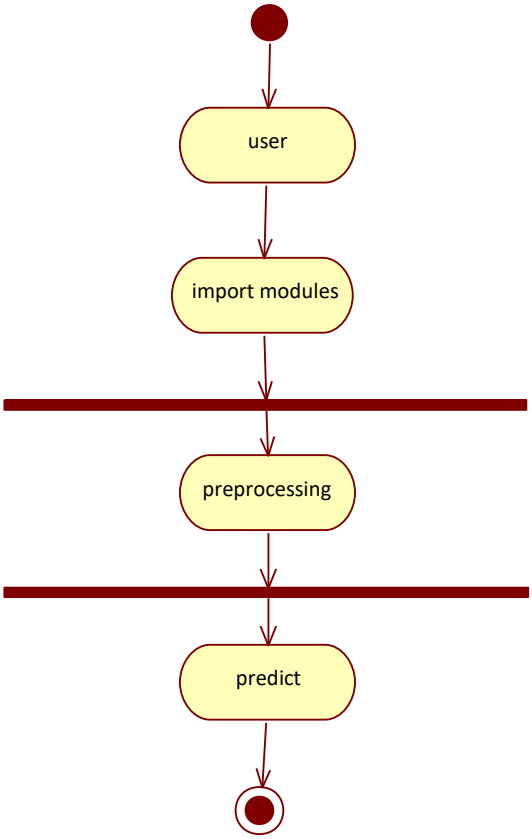


. RESULT

The cv score of random testing of our data is performed of hyper parameters for four different type of algorithums.



Activity diagram



IMPLEMENTATION

1 Import relevant libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

2 Load dataset

 # Load the dataset and read the dataset

```
data = pd.read_csv("SolarPower.csv")
```

Out[3]:

	Location	Date	Time	Latitude	Longitude	Altitude	YRMODAHRMI	Month	Hour	Season	Humidity	AmbientTemp	PolyPwr	Wind.Speed	Visibility	Pressure	Cloud.Ceiling
0	Camp Murray	20171203	1145	47.11	-122.57	84	2.020000e+11	12	11	Winter	81.71997	12.86919	2.42769	5	10.0	1010.6	722
1	Camp Murray	20171203	1315	47.11	-122.57	84	2.020000e+11	12	13	Winter	96.64917	9.66415	2.46273	0	10.0	1011.3	23
2	Camp Murray	20171203	1330	47.11	-122.57	84	2.020000e+11	12	13	Winter	93.61572	15.44983	4.46836	5	10.0	1011.6	32
3	Camp Murray	20171204	1230	47.11	-122.57	84	2.020000e+11	12	12	Winter	77.21558	10.36659	1.65364	5	2.0	1024.4	6
4	Camp Murray	20171204	1415	47.11	-122.57	84	2.020000e+11	12	14	Winter	54.80347	16.85471	6.57939	3	3.0	1023.7	9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
21040	USAFA	20180928	1530	38.95	-104.83	1947	2.020000e+11	9	15	Fall	11.66992	43.22510	9.79611	14	10.0	802.3	722
21041	USAFA	20180929	1300	38.95	-104.83	1947	2.020000e+11	9	13	Fall	18.22510	28.98247	10.88992	13	10.0	799.2	722
21042	USAFA	20180929	1400	38.95	-104.83	1947	2.020000e+11	9	14	Fall	15.52124	33.49167	8.24479	10	10.0	798.4	722
21043	USAFA	20180929	1500	38.95	-104.83	1947	2.020000e+11	9	15	Fall	6.63452	51.62163	12.47328	10	10.0	797.8	722
21044	USAFA	20181001	1400	38.95	-104.83	1947	2.020000e+11	10	14	Fall	22.58301	32.83958	6.39732	15	10.0	801.2	110

21045 rows x 17 columns

4.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21045 entries, 0 to 21044
Data columns (total 17 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Location      21045 non-null  object
 1   Date          21045 non-null  int64
 2   Time          21045 non-null  int64
 3   Latitude      21045 non-null  float64
 4   Longitude     21045 non-null  float64
 5   Altitude      21045 non-null  int64
 6   YRMODAHRMI    21045 non-null  float64
 7   Month         21045 non-null  int64
 8   Hour          21045 non-null  int64
 9   Season        21045 non-null  object
 10  Humidity      21045 non-null  float64
 11  AmbientTemp   21045 non-null  float64
 12  PolyPwr       21045 non-null  float64
 13  Wind.Speed    21045 non-null  int64
 14  Visibility    21045 non-null  float64
 15  Pressure      21026 non-null  float64
 16  Cloud.Ceiling 21045 non-null  int64
dtypes: float64(8), int64(7), object(2)
memory usage: 2.7+ MB
```

```
data.isna().any()
```

```
Location        False
Date            False
Time            False
Latitude        False
Longitude       False
Altitude        False
YRMODAHRMI      False
Month           False
Hour            False
Season          False
Humidity        False
AmbientTemp     False
PolyPwr         False
Wind.Speed      False
Visibility      False
Pressure         True
Cloud.Ceiling   False
dtype: bool
```

3 Explore the data

3-i Identify available columns in the data

```
data.columns
```

```
Index(['Location', 'Date', 'Time', 'Latitude', 'Longitude', 'Altitude',
       'YRMODAHRMI', 'Month', 'Hour', 'Season', 'Humidity', 'AmbientTemp',
       'PolyPwr', 'Wind.Speed', 'Visibility', 'Pressure', 'Cloud.Ceiling'],
      dtype='object')
```

3-ii Check for missing values, Impute missing values if present

```
data.isnull().sum()
```

```
Location        0
Date            0
Time            0
Latitude        0
Longitude       0
Altitude        0
YRMODAHRMI      0
Month           0
Hour            0
Season          0
Humidity        0
AmbientTemp     0
PolyPwr         0
Wind.Speed      0
Visibility      0
Pressure       19
Cloud.Ceiling   0
dtype: int64
```

```
data.fillna(data['Pressure'].mean(), inplace =True)
```

```
data
```

Out[8]:

	Location	Date	Time	Latitude	Longitude	Altitude	YRMODAHRMI	Month	Hour	Season	Humidity	AmbientTemp	PolyPwr	Wind.Speed	Visibility	Pressure	Cloud.Ceiling
0	Camp Murray	20171203	1145	47.11	-122.57	84	2.020000e+11	12	11	Winter	81.71997	12.86919	2.42769	5	10.0	1010.6	722
1	Camp Murray	20171203	1315	47.11	-122.57	84	2.020000e+11	12	13	Winter	96.64917	9.66415	2.46273	0	10.0	1011.3	23
2	Camp Murray	20171203	1330	47.11	-122.57	84	2.020000e+11	12	13	Winter	93.61572	15.44983	4.46836	5	10.0	1011.6	32
3	Camp Murray	20171204	1230	47.11	-122.57	84	2.020000e+11	12	12	Winter	77.21558	10.36659	1.65364	5	2.0	1024.4	6
4	Camp Murray	20171204	1415	47.11	-122.57	84	2.020000e+11	12	14	Winter	54.80347	16.85471	6.57939	3	3.0	1023.7	9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
21040	USAFA	20180928	1530	38.95	-104.83	1947	2.020000e+11	9	15	Fall	11.66992	43.22510	9.79611	14	10.0	802.3	722
21041	USAFA	20180929	1300	38.95	-104.83	1947	2.020000e+11	9	13	Fall	18.22510	28.98247	10.88992	13	10.0	799.2	722
21042	USAFA	20180929	1400	38.95	-104.83	1947	2.020000e+11	9	14	Fall	15.52124	33.49167	8.24479	10	10.0	798.4	722
21043	USAFA	20180929	1500	38.95	-104.83	1947	2.020000e+11	9	15	Fall	6.63452	51.62163	12.47328	10	10.0	797.8	722
21044	USAFA	20181001	1400	38.95	-104.83	1947	2.020000e+11	10	14	Fall	22.58301	32.83958	6.39732	15	10.0	801.2	110

data.isnull().sum() Out[9]:

Location 0
Date 0
Time 0
Latitude 0
Longitude 0
Altitude 0
YRMODAHRMI 0
Month 0
Hour 0
Season 0
Humidity 0
AmbientTemp 0
PolyPwr 0
Wind.Speed 0
Visibility 0
Pressure 0
Cloud.Ceiling 0
dtype: int64

3-iii Calculate summary statistics of numerical columns In [10]:

data.describe()

	Date	Time	Latitude	Longitude	Altitude	YRMODAHRMI	Month	Hour	Humidity	AmbientTemp	PolyPwr	Wind.Speed	
count	2.104500e+04	21045.000000	21045.000000	21045.000000	21045.000000	2.104500e+04	21045.000000	21045.000000	21045.000000	21045.000000	21045.000000	21045.000000	2
mean	2.017720e+07	1267.483725	38.213823	-108.593678	798.843668	2.020000e+11	6.565883	12.627845	37.121941	29.285117	12.978583	10.318318	
std	4.579585e+03	167.602767	6.323761	16.364130	770.681794	0.000000e+00	2.983958	1.672952	23.823011	12.366820	7.123255	6.385030	
min	2.017052e+07	1000.000000	20.890000	-156.440000	1.000000	2.020000e+11	1.000000	10.000000	0.000000	-19.981770	0.257330	0.000000	
25%	2.017111e+07	1100.000000	38.160000	-117.260000	2.000000	2.020000e+11	4.000000	11.000000	17.529300	21.915280	6.404570	6.000000	
50%	2.018032e+07	1300.000000	38.950000	-111.180000	458.000000	2.020000e+11	7.000000	13.000000	33.123780	30.289150	13.798700	9.000000	
75%	2.018062e+07	1400.000000	41.150000	-104.710000	1370.000000	2.020000e+11	9.000000	14.000000	52.593990	37.474670	18.863650	14.000000	
max	2.018100e+07	1545.000000	47.520000	-80.110000	1947.000000	2.020000e+11	12.000000	15.000000	99.987790	65.738370	34.285020	49.000000	

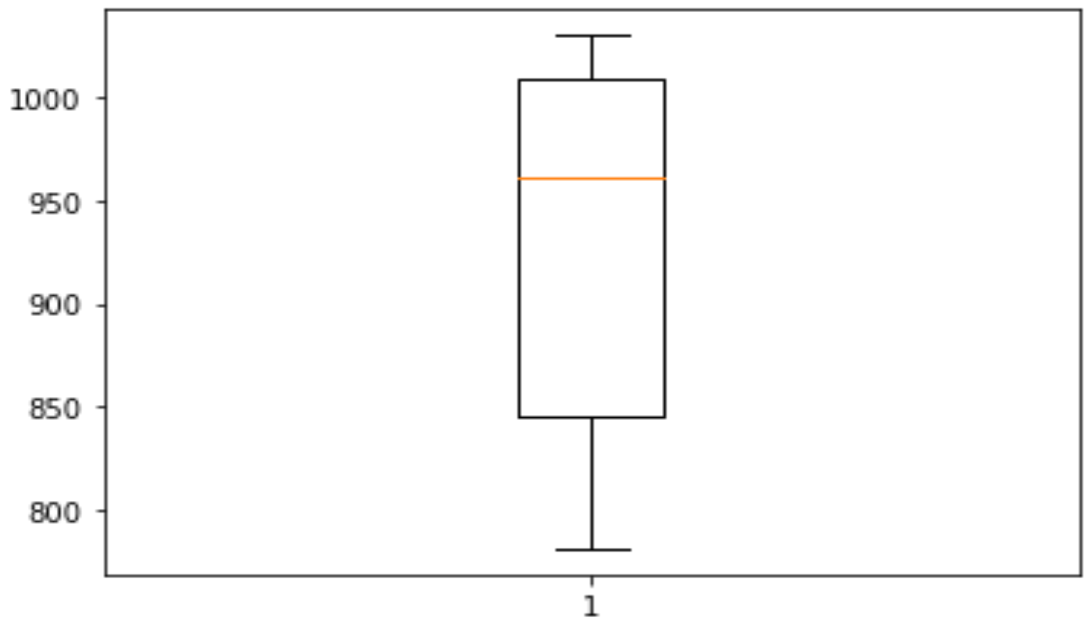
3-iv Visualize data distribution

In [11]:

```
plt.boxplot(data['Pressure'])
```

Out[11]:

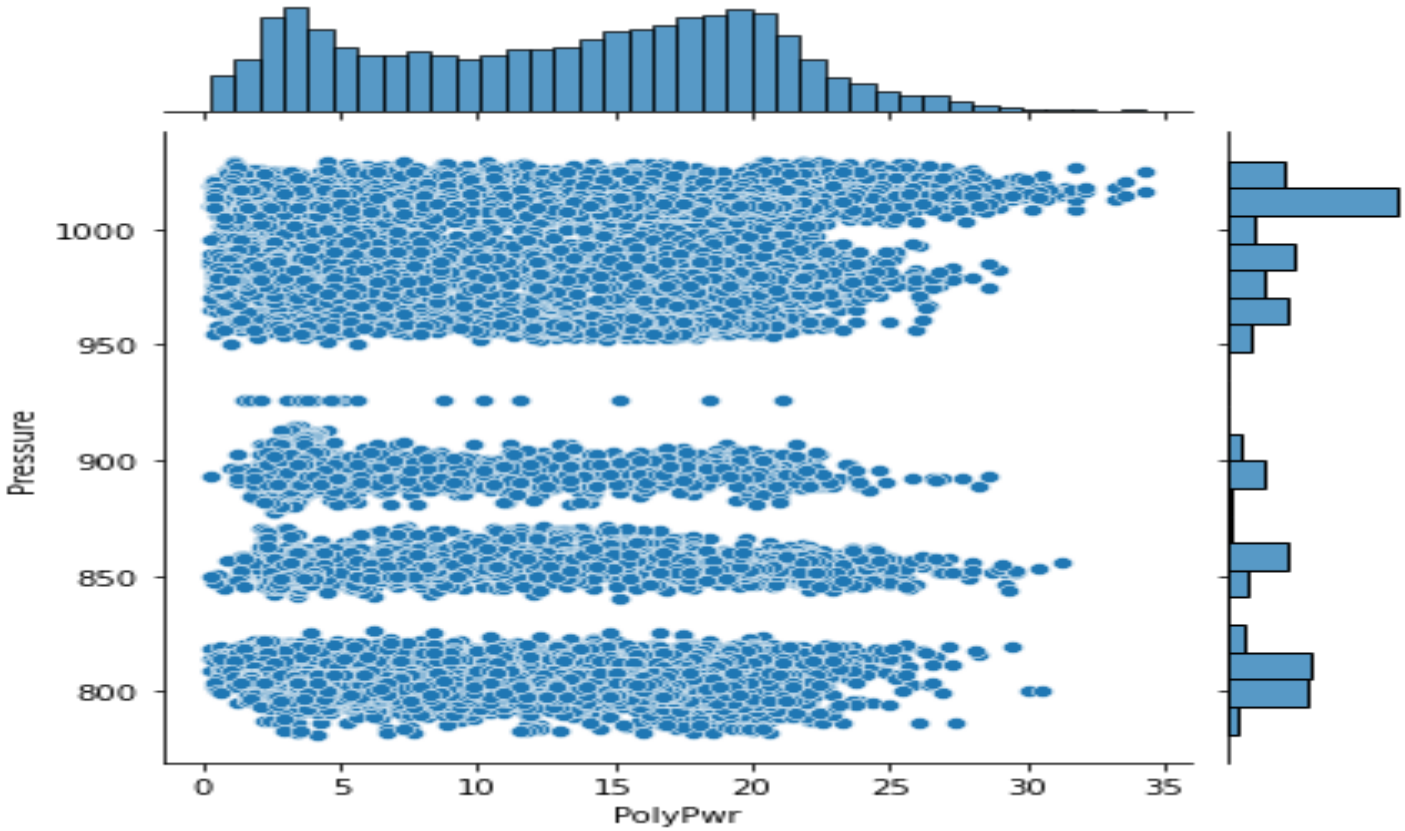
```
{'whiskers': [<matplotlib.lines.Line2D at 0x23705a19880>,
  <matplotlib.lines.Line2D at 0x23705a19b50>],
 'caps': [<matplotlib.lines.Line2D at 0x23705a19ee0>,
  <matplotlib.lines.Line2D at 0x23705da12b0>],
 'boxes': [<matplotlib.lines.Line2D at 0x23705a19550>],
 'medians': [<matplotlib.lines.Line2D at 0x23705da1640>],
 'fliers': [<matplotlib.lines.Line2D at 0x23705da19d0>],
 'means': []}
```



```
sns.jointplot(x = data['PolyPwr'],y = data['Pressure'])
```

Out[12]:

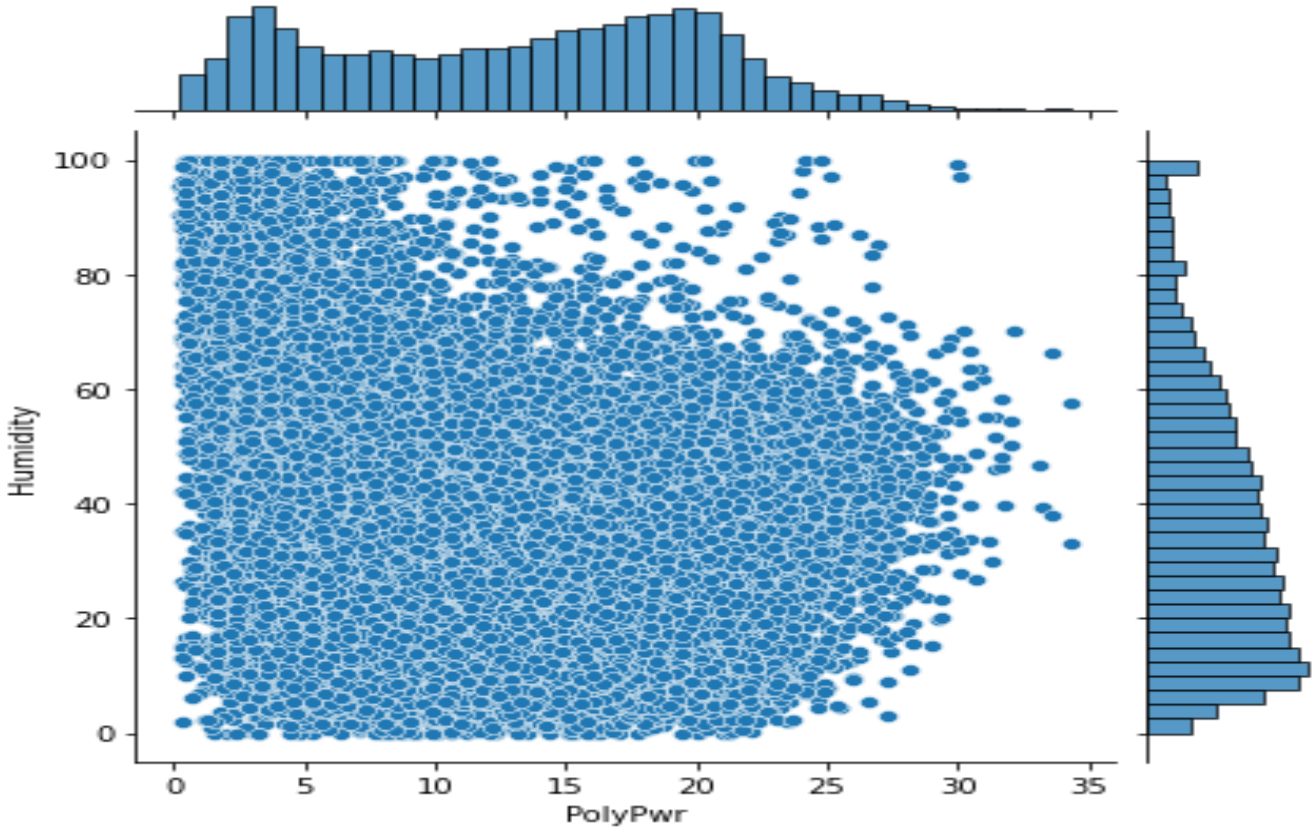
```
<seaborn.axisgrid.JointGrid at 0x23705dc4df0>
```




```
sns.jointplot(x = data['PolyPwr'],y = data['Humidity'])
```

Out[13]:

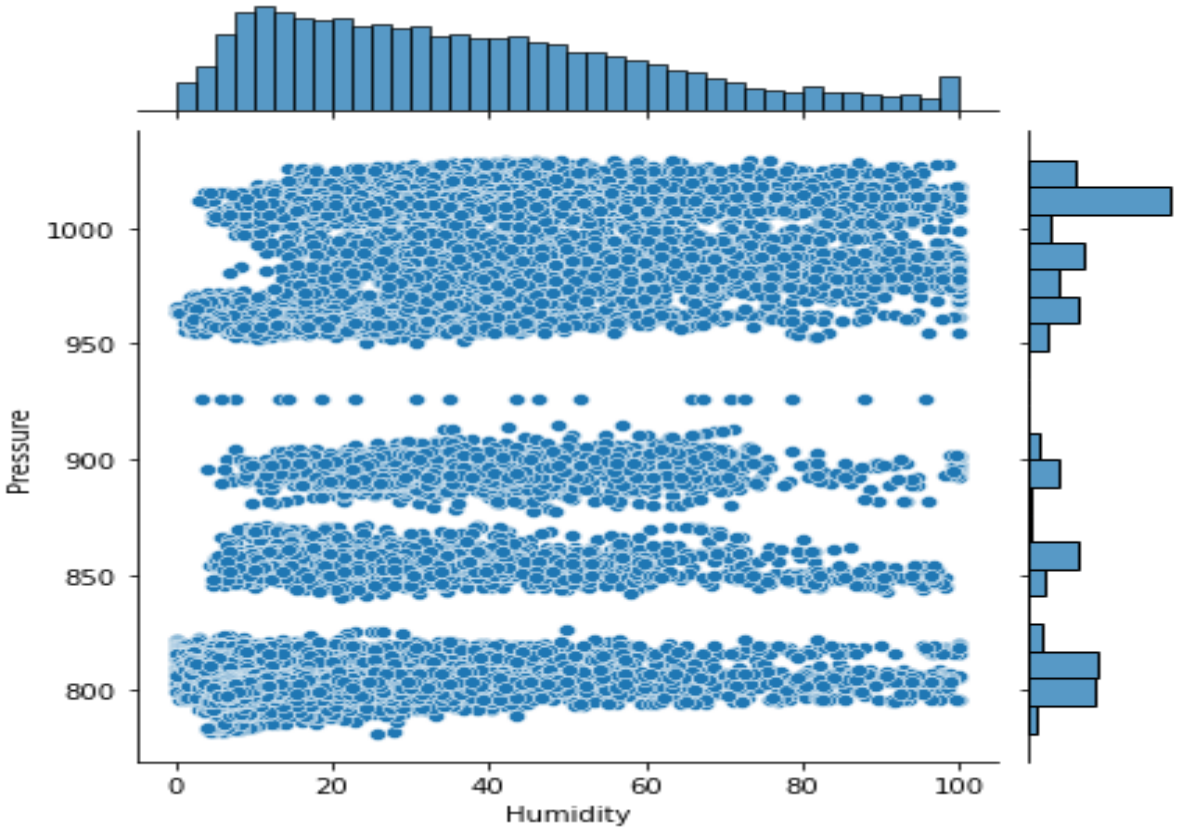
```
<seaborn.axisgrid.JointGrid at 0x237065067c0>
```




```
sns.jointplot(x = data['Humidity'],y = data['Pressure'])
```

Out[14]:

```
<seaborn.axisgrid.JointGrid at 0x237066ae490>
```



3-v Correlation analysis

In [15]:

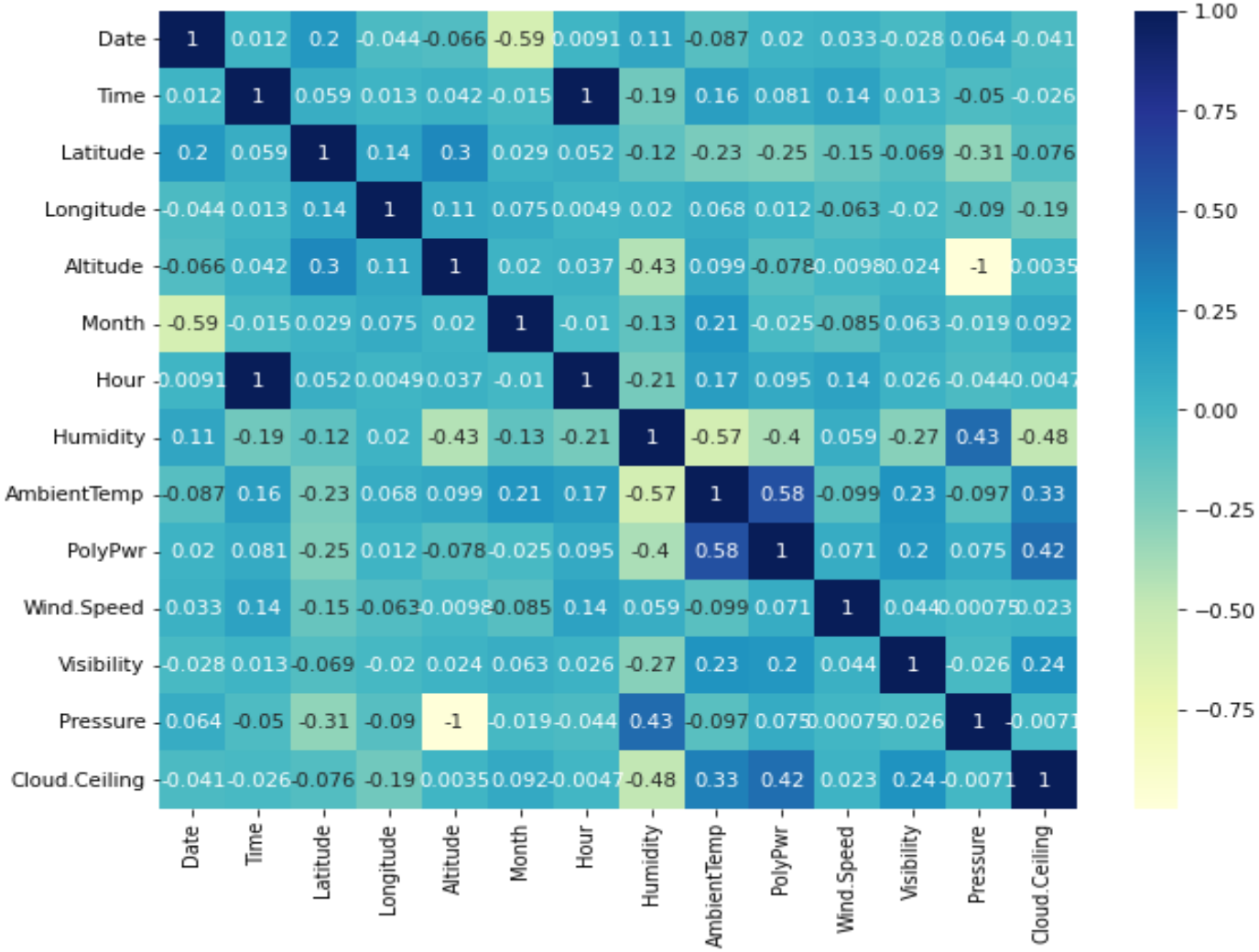
```
data.drop('YRMODAHRMI',axis='columns',inplace=True)
```

In [16]:

 # "YRMODAHRMI" column was dropped because it is not intutive and no description is provided.

In [17]:

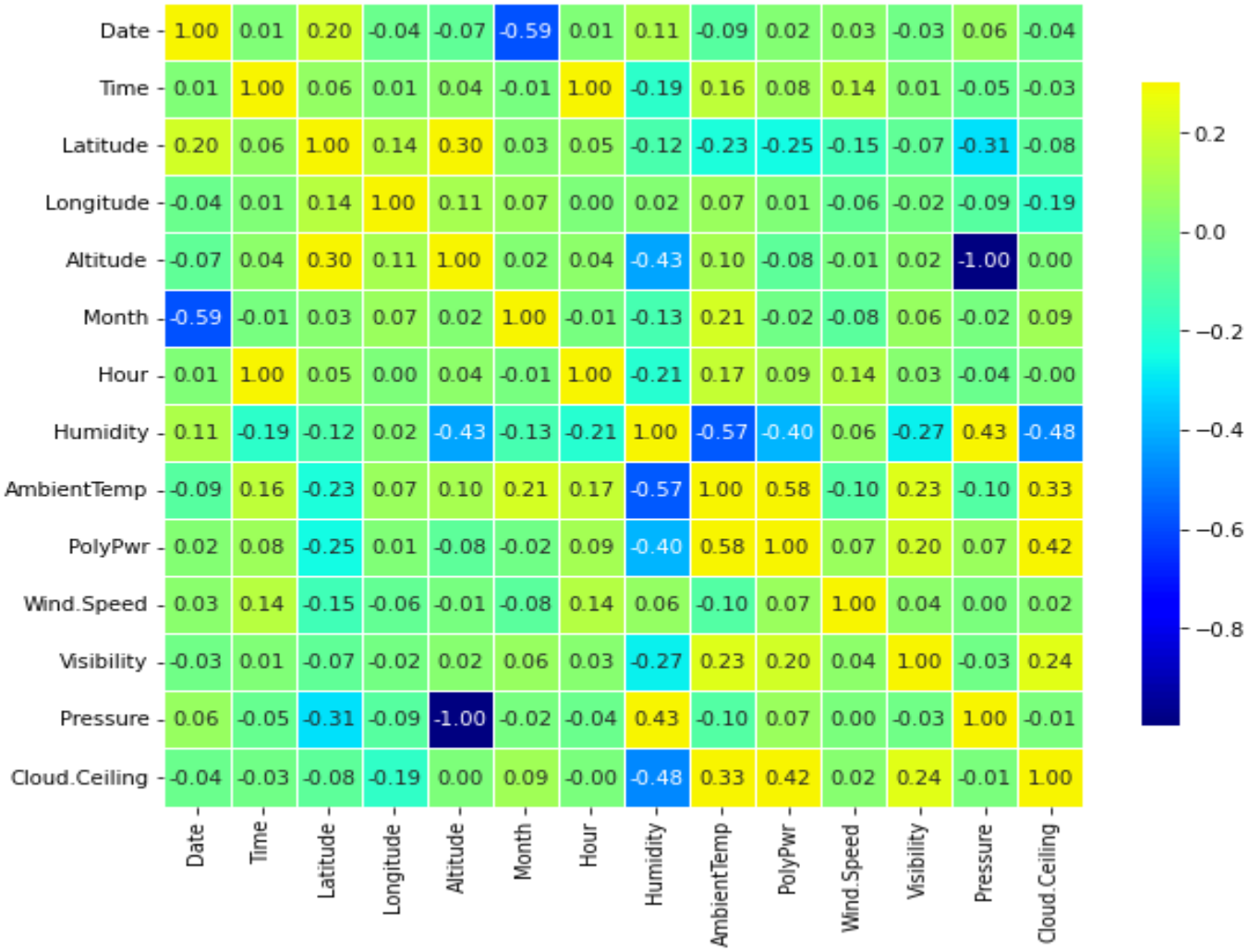
```
data.corr()
```



plt.figure(figsize=(10,8))
sns.heatmap(data.corr(), annot=True , cmap ='YlGnBu')

Out[18]:

<AxesSubplot:>



Average of PolyPwr with respective location

In [21]:

```
data1 =pd.pivot_table(data,index= 'Location',values = 'PolyPwr',aggfunc='mean')
```

In [22]:

```
data1
```

Out[22]:

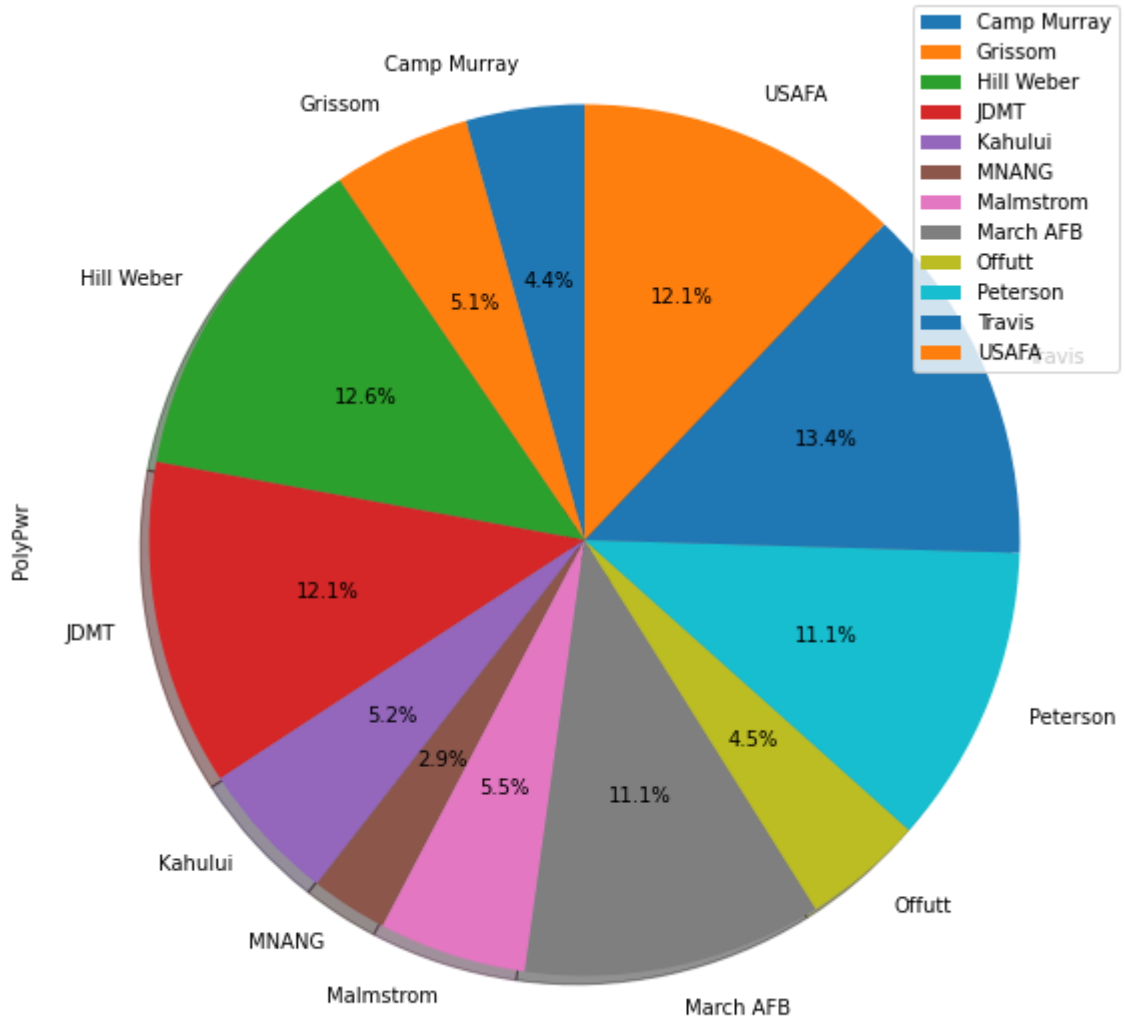
Location	PolyPwr
Camp Murray	10.777723
Grissom	9.426174
Hill Weber	14.437151
JDMT	18.574124
Kahului	15.222645
MNANG	9.996263
Malmstrom	9.886024

Location	PolyPwr
March AFB	13.761134
Offutt	13.895667
Peterson	11.517351
Travis	13.296735
USAFA	12.845495

Pie chart draw in respective columns

In [23]:

```
plt=data.groupby(data['Location']).sum().plot(kind='pie',y='PolyPwr', subplots=True, shadow = True,startangle=90,
figsize=(15,10), autopct='%1.1f%%')
```



MODEL TRAINING:

KNN REGRESSION ALGORITHM

 # Define time bounds in data

```
min_hour_of_interest = 10
max_hour_of_interest = 15
# Calculate time lapse since onset of power generation
df_with_loc_season_en['delta_hr']= df_with_loc_season_en.Hour - min_hour_of_interest
df_with_loc_season_en['delta_hr']
```

Out[27]:

```
0      1
1      3
2      3
3      2
4      4
          ..
21040  5
21041  3
21042  4
21043  5
21044  4
Name: delta_hr, Length: 21045, dtype: int64
```

In [28]:

```
# Create cyclic month features
df_with_loc_season_en['sine_mon']= np.sin((df_with_loc_season_en.Month - 1)*np.pi/11)
df_with_loc_season_en['sine_mon']
```

Out[28]:

```
0      5.665539e-16
1      5.665539e-16
2      5.665539e-16
3      5.665539e-16
4      5.665539e-16
                  ...
21040  7.557496e-01
21041  7.557496e-01
21042  7.557496e-01
21043  7.557496e-01
21044  5.406408e-01
Name: sine_mon, Length: 21045, dtype: float64
```

In [29]:

```
df_with_loc_season_en['cos_mon']= np.cos((df_with_loc_season_en.Month - 1)*np.pi/11)
df_with_loc_season_en['cos_mon']
```

Out[29]:

```
0     -1.000000
1     -1.000000
2     -1.000000
3     -1.000000
4     -1.000000
              ...
21040 -0.654861
21041 -0.654861
21042 -0.654861
21043 -0.654861
21044 -0.841254
Name: cos_mon, Length: 21045, dtype: float64
```

In [30]:

```
# Create cyclic hour features
df_with_loc_season_en['sine_hr']= np.sin((df_with_loc_season_en.delta_hr*np.pi/(max_hour_of_interest -
min_hour_of_interest)))
df_with_loc_season_en['sine_hr']
```

Out[30]:

```
0        5.877853e-01
1        9.510565e-01
2        9.510565e-01
3        9.510565e-01
4        5.877853e-01
                ...
21040    1.224647e-16
21041    9.510565e-01
21042    5.877853e-01
21043    1.224647e-16
21044    5.877853e-01
Name: sine_hr, Length: 21045, dtype: float64
```

In [ ]:

```
df_with_loc_season_en['cos_hr']= np.cos((df_with_loc_season_en.delta_hr*np.pi/(max_hour_of_interest - min_hour_of_interest)))
df_with_loc_season_en['cos_hr']
```

adding columns

In [ ]:

```
data[['delta_hr', 'sin_mon', 'cos_mon','sine_hr','cos_hr']] = pd.DataFrame([[df_with_loc_season_en['delta_hr'],df_with_loc_season_en['sine_mon'],df_with_loc_season_en['cos_mon'],df_with_loc_season_en['sine_hr'],df_with_loc_season_en['cos_hr']]], index=data.index)
data
```

Linear Regression

In [ ]:

```
from sklearn.model_selection import train_test_split
predictors=["Humidity"]
target=["PolyPwr"]
x=data[predictors]
y=data[target]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33)
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x,y)
print(model.intercept_)
model.coef_
rsq=model.score(x_test,y_test)
print(rsq)
rsq1=model.score(x_train,y_train)
rsq1
```

KNN Regression

In [ ]:

```
predictors=["Humidity"]
target=["PolyPwr"]
X=data[predictors]
Y=data[target]
```

In [ ]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, Y, test_size=0.2, random_state=0)
```

In [ ]:

```
from sklearn.neighbors import KNeighborsRegressor
knn_model = KNeighborsRegressor(n_neighbors=3)
```

In [ ]:

```
knn_model.fit(X_train, y_train)
```

In [ ]:

```
from sklearn.metrics import mean_squared_error
from math import sqrt
# x_Train
train_preds = knn_model.predict(X_train)
mse = mean_squared_error(y_train, train_preds)
rmse = sqrt(mse)
rmse
```

In [ ]:

```
# X_Test
test_preds = knn_model.predict(X_test)
mse = mean_squared_error(y_test, test_preds)
rmse = sqrt(mse)
rmse
```

KNN Classifier

In [ ]:

```
dummies = pd.get_dummies(data.Season)
dummies
```

In [ ]:

```
data['PolyPwr']=data['PolyPwr'].astype(int)
```

In [ ]:

```
predictors=["Fall","Spring","Summer","Winter"]
target=["PolyPwr"]
X=dummies[predictors]
y=data[target]
```

In [ ]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=0)
```

In [ ]:

```
#knn classifier
from sklearn.neighbors import KNeighborsClassifier
model=KNeighborsClassifier()
```

In [ ]:

```
model.fit(X_train,y_train)
```

In [ ]:

```
y_pred=model.predict(X_test)
```

In [ ]:

```
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,y_pred))
confusion_matrix(y_test,y_pred)
```

In [ ]:

```
from sklearn.model_selection import cross_val_score
print('cross val',cross_val_score(model,y_test,y_pred))
print
```

In [ ]:

```
model.score(X_test,y_test)
```

Random Forest

In [ ]:

```
data['PolyPwr']=data['PolyPwr'].astype(int)
```

In [ ]:

```
from sklearn.model_selection import train_test_split
```

```
X=data[['Altitude']]  # Features
y=data['PolyPwr']  # Labels

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% training and 30% test
```

In [ ]:

```
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)

y_pred=clf.predict(X_test)
```

In [ ]:

```
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

In [ ]:

```
import matplotlib.pyplot as plt
import seaborn as sns
# Creating a bar plot
# Model Accuracy, how often is the classifier correct?
sns.barplot(y_test, y_pred)
# Add labels to your graph
plt.xlabel('Feature Importance Score')
plt.ylabel('polypower')
plt.title("Visualizing Important Features")
plt.legend()
plt.show()
Cross-validation
```

In [ ]:

```
# Predicting th input features , x=Predictotrs
x = np.array(data[['AmbientTemp','Humidity','Pressure','Cloud.Ceiling']])
x.shape
```

In [ ]:

```
# proving the output feature, y = regressand
y =np.array(data['PolyPwr'])
y.shape
```

In [ ]:

```
# splitting the dataset into train and test datasets
# importing the train_test_split method
from sklearn.model_selection import train_test_split
X_train, X_test,y_train, y_test = train_test_split(x, y, test_size=0.30)
y_train.shape
```

In [ ]:

```
# importig the linear regerssion class
from sklearn.linear_model import LinearRegression
# Instantiation
model =  LinearRegression()
# Fitting the model
mymodel = model.fit(X_train,y_train)
```

In [ ]:

```
# k- fold cv
```

```
# Importing cross_val_score function
from sklearn.model_selection import cross_val_score
# 10-fold cv
score = cross_val_score(mymodel,X_train,y_train,scoring ='r2',cv = 10) # scoring ='r2' random cross validation
score
```

In [ ]:

```
#printing the average score
print(np.mean(score))
```

In [ ]:

```
# printing the score on the test dataset
# first:prediction
# importing cross_val_predict function
from sklearn.model_selection import cross_val_predict
# The predictions
pred = cross_val_predict(model,X_test,y_test)
pred
```

In [ ]:

```
# 10-fold cv on test data
score_test = cross_val_score(model,X_test,y_test,cv = 10)
score_test
```

In [ ]:

```
# The average score
print(np.mean(score_test))
```

REFERENCES

1. P. A. G. M. Amarasinghef and S. K. Abeygffunawardane, "Application of Machine Learning Algorithms for Solar Power Forecafsting in Sri Lafnka"

2. M. Z. Hasdsdfsan, M. E. K. Ali, A. B. M. S. Ali and J. Kumar, "Forecasting Day-Adhead Solar Radiation Using Machine Learnin

3. A. Bajpai and M. Duchon, "Hybrid Approach of Solar Power Forecasting Using Machine Learning

4. A. Khan, R. Bhatnagar, V. Masrani and V. B. Lobo, "A Comparative Study on Solar Power Forecasting using Ensemble Learning,"

5. Khan, P.W.; Byun, Y.-C.; Lee, S.-J.; Kang, D.-H.; Kang, J.-Y.; Park, H.-S. Energies, 13, 4870 (2020).

6 Tao Hong, Pierre Pinson, Shu Fan, Hamidrez Zareipour, Alberto Troccoli, and Rob J. Hyndman. Probabilistic energy forecasting: Global energy forecasting competiton 2014 and beyond. Internatinal Journal of Forecasting, 32(3):896 – 913, 2016.

7. Gordon Reikard. Predicting solar radiation at high resolutins: A comparison of time series forecasts. Solar Energy, 83 – 349, 2009.

8 Peder Bacher, Henrik Masen, and Henrik Aalborg Nielsen. Online short-term solar power forecasting. Solar Energy, 3(10):1772 – 1783, 2009.

9. Hugo T.C. Pedro and Carlos F.M. Coimbra. Assesssment of forecasting techniques for solar peower production with no exogenous inputs. Solar Energy, 86(7):2017 – 2028, 2012.