Appliance Energy Prediction Using Machine Learning – A data Science Approach

A project work report

Submitted in partial fulfilment of the

Requirements of the degree of

Master of Science (Tech.) Engineering Physics

In

Electronics

By

NUTAN TRIPATHY

Roll No. 195927

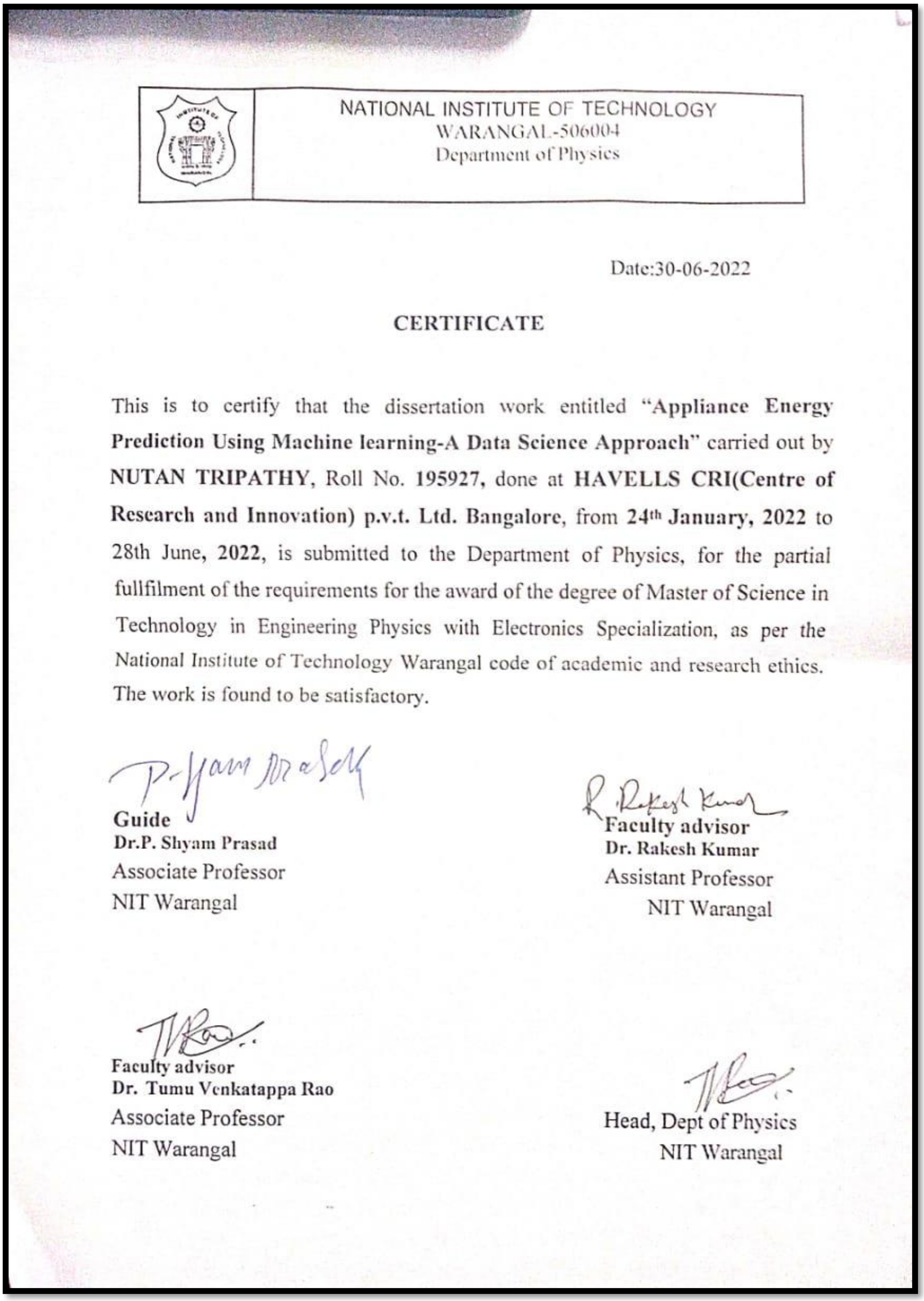
Under Supervision of

Dr. P. SYAM PRASAD P	**Dr. MARIMUTHU C**
ASSOCIATE PROFESSOR	**RESEARCH ENGINEEER**
DEPARTMENT OF PHYSICS	**CRI HAVELLS**
NIT WARANGAL	**HAVELLS, BANGALORE**

DEPARTMENT OF PHYSICS

NATIONAL INSTITUTE OF TECHNOLOGY, WARANGAL

JUNE, 2022



NATIONAL INSTITUTE OF TECHNOLOGY
WARANGAL-506004
Department of Physics

Date:30-06-2022

CERTIFICATE

This is to certify that the dissertation work entitled **"Appliance Energy Prediction Using Machine learning-A Data Science Approach"** carried out by **NUTAN TRIPATHY**, Roll No. **195927**, done at **HAVELLS CRI(Centre of Research and Innovation) p.v.t. Ltd. Bangalore**, from **24th January, 2022** to 28th June, **2022**, is submitted to the Department of Physics, for the partial fullfilment of the requirements for the award of the degree of Master of Science in Technology in Engineering Physics with Electronics Specialization, as per the National Institute of Technology Warangal code of academic and research ethics. The work is found to be satisfactory.

[illegible]
Guide
Dr.P. Shyam Prasad
Associate Professor
NIT Warangal

[illegible]
Faculty advisor
Dr. Rakesh Kumar
Assistant Professor
NIT Warangal

[illegible]
Faculty advisor
Dr. Tumu Venkatappa Rao
Associate Professor
NIT Warangal

[illegible]
Head, Dept of Physics
NIT Warangal

Certificate of Approval

It is to certify that final year project report entitled "**Appliance Energy Prediction Using Machine Learning – A data Science Approach**" submitted by **NUTAN TRIPATHY** has been carried out under the guidance of **Dr. P SYAM PRASAD (ASSOCIATE PROF. NITW) AND Dr. MARIMUTHU C (RESEARCH ENGINEER, CRI HAVELLS, BANGALORE).** The project report is approved for submission requirement for final year project in 6th semester in **Engineering Physics (Electronics)** from **NATIONAL INSTITUTE OF TECHNOLOGY, WARANGAL, Telangana.**

Dr. MARIMUTHU C

RESEARCH ENGINEER,

CRI Havells, Bangalore

[illegible]

ACKNOWLEDGEMENT

Thank you to my guide, **Dr.P. Syam Prasad**, for your patience, guidance, and support. I have benefited greatly from your wealth of knowledge and meticulous editing. I am extremely grateful that you took me on as a student and continued to have faith in me over the years.

I express my deep sense of gratitude and respect to my mentor **Dr. Marimuthu**. He has been an ideal teacher, mentor, and thesis supervisor, offering advice and encouragement with a perfect blend of insight and humor. I'm proud of, and grateful for, my time working with him.

I am also grateful to respected **Dr. DINAKAR D. (HOD-Physics dept. NITW)** and to our respected director **Prof. N.V. RAMANA RAO(NITW)** for permitting me to utilize all the necessary facilities of the institution.

Lastly, I would like to thank all staff members of Havells **CRI, Bangalore** for providing the moral support and encouragement.

AUTHOR'S DECLARATION

I hereby declare that the project entitled "**Appliance Energy Prediction Using Machine Learning – A data Science Approach**" submitted by me to **NATIONAL INSTITUTE OF TECHNOLOGY**, **Warangal**, was carried out in accordance with the requirements of the University's regulations and code for practice for Master Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the authors.

Date: 15/06/2021 **NUTAN TRIPATHY**

Place: NIT Warangal

Table of Contents

List Of Figures

Introduction to Data Analytics and Machine Learning

Data Analytics

Data Analytics usually refers to the techniques used to analyze data and to enhance productivity. Data is extracted from various sources and is cleaned and categorized to analyze various behavioural patterns. In this project we are using different machine learning models to do the analysis of data and to predict further results

Introduction to Machine Learning

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people. Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs. Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type. Recommendation engines, powered by machine learning, suggest what movies or television shows to watch next based on user preferences. Self-driving cars that rely on machine learning to navigate may soon be available to consumers. Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyze the impact of machine learning processes.[1]

It is used in various sectors such as: -

- Financial Services
- Healthcare
- For Government Programs

- For Transportation
- In Retail Services

Types Of Machine Learning

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

Supervised Machine Learning

In supervised learning, you train your model on a labelled dataset that means we have both raw input data as well as its results. We split our data into a training dataset and test dataset where the training dataset is used to train our network whereas the test dataset acts as new data for predicting results or to see the accuracy of our model. Hence, in supervised learning, our model learns from seen results the same as a teacher teaches his students because the teacher already knows the results. Accuracy is what we achieve in supervised learning as model perfection is usually high.

Unsupervised Learning

In unsupervised learning, the information used to train is neither classified nor labelled in the dataset. Unsupervised learning studies on how systems can infer a function to describe a hidden structure from unlabelled data. The main task of unsupervised learning is to find patterns in the data. Once a model learns to develop patterns, it can easily predict patterns for any new dataset in the form of clusters. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from un-labelled data

Reinforcement Learning

It is a Machine Learning algorithm that allows software agents and machines to automatically determine the ideal behaviour within a specific context to maximize its performance. It does not have a labelled dataset or results associated with data so the only way to perform a given task is to learn from experience. For every correct action or decision of algorithm, it is rewarded with positive reinforcement whereas, for every incorrect action, it is rewarded with negative reinforcement. In this way, it learns which actions are needed to perform and which are not.[2] In our project we are using already labelled dataset therefore we will train according to the rules of supervised Learning

Supervised Learning And its Types

Supervised learning uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.[3] Supervised learning can be separated into two types of problems when data mining— classification and regression:

- Classification uses an algorithm to accurately assign test data into specific categories. It recognizes specific entities within the dataset and attempts to draw some conclusions on how those entities should be labelled or defined. Common classification algorithms are linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbour, and random forest, which are described in more detail below.

- Regression is used to understand the relationship between dependent and independent variables. It is commonly used to make projections, such as for sales revenue for a given business. Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

SOME BASIC ALGORITHMS RELATED TO OUR STUDY

K-Nearest Neighbour (K-NN)

It is a simple algorithm that stores all the available cases and classifies the new data or case based on a similarity measure. In K-*NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbours, with the object being assigned to the class most common among its *k* nearest neighbours (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbour.To determine which of the K instances in the training dataset are most similar to a new input, a distance measure is used. [5]

Linear regression

Linear regression is used to identify the relationship between a dependent variable and one or more independent variables and is typically leveraged to make predictions about future outcomes. When there is only one independent variable and one dependent variable, it is known as simple linear regression. As the number of independent variables increases, it is referred to as multiple linear regression. For each type of linear regression, it seeks to plot a line of best fit, which is calculated through the method of Comparing different models to Identify heart disease from 7 least squares. However, unlike other regression models, this line is straight when plotted on a graph.

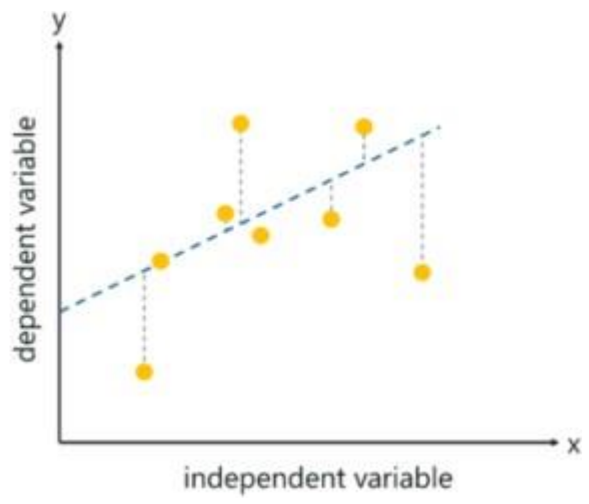


Fig 1 − Graph of linear regression[Supervised Learning | What is, Types, Applications and Example | Edureka]

Logistic regression

While linear regression is leveraged when dependent variables are continuous, logistical regression is selected when the dependent variable is categorical, meaning they have binary outputs, such as "true" and "false" or "yes" and "no." While both regression models seek to understand relationships between data inputs, logistic regression is mainly used to solve binary classification problems, such as spam identification.

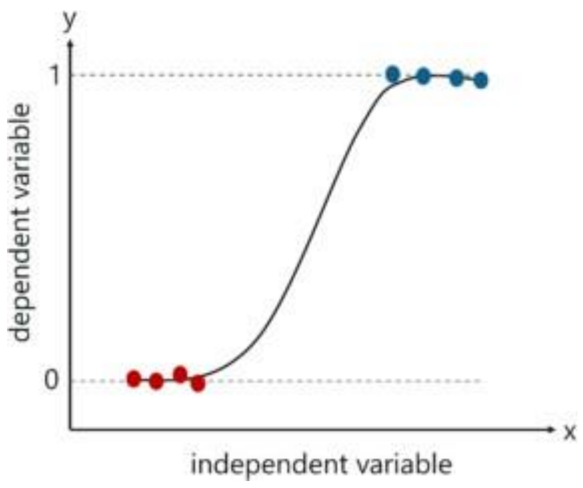


Fig 2 – Graph of logistic regression [Supervised Learning | What is, Types, Applications and Example | Edureka]

Support vector machine (SVM)

A support vector machine is a popular supervised learning model developed by Vladimir Vapnik, used for both data classification and regression. They use Kernal functions which are a central concept for most of the learning tasks. That said, it is typically leveraged for classification problems, constructing a hyperplane where the distance between two classes of data points is at its maximum. This hyperplane is known as the decision boundary, separating the classes of data points (e.g., oranges vs. apples) on either side of the plane.

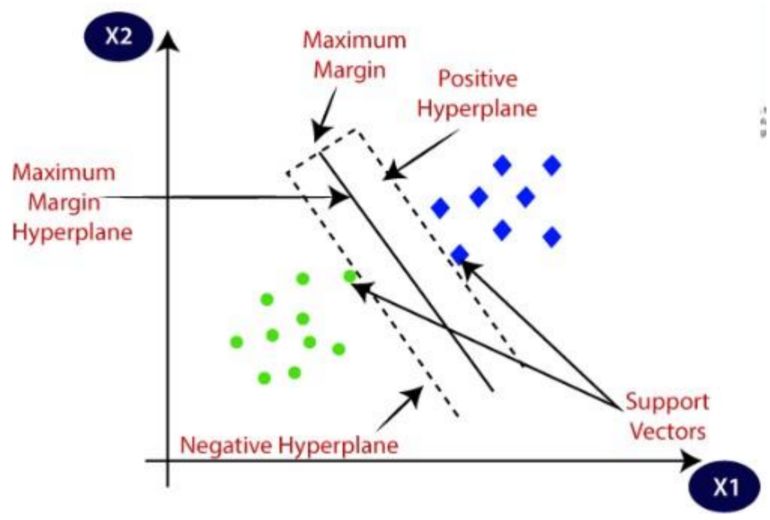
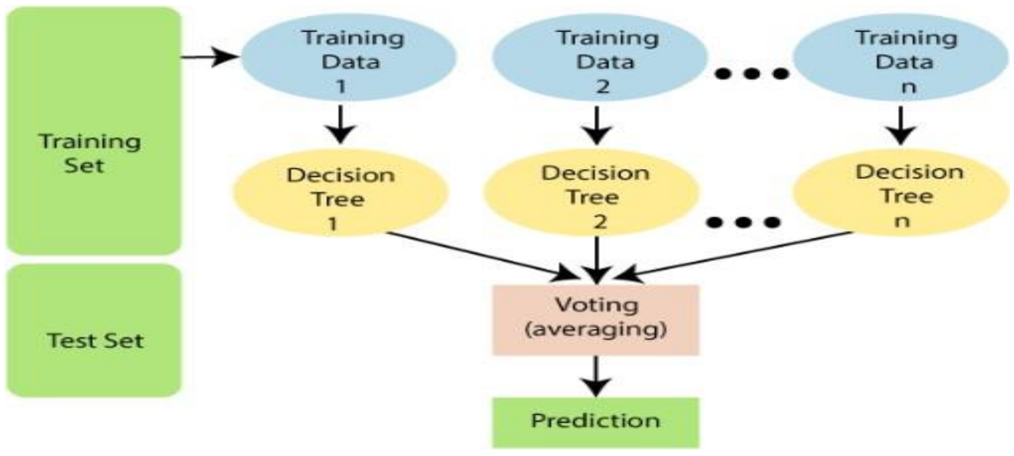


Fig 3 – graph of hyperplane for SVM algorithm [Support Vector Machine (SVM) Algorithm - Javatpoint]

Random forest

Random forest is another flexible supervised machine learning algorithm used for both classification and regression purposes. The "forest" references a collection of uncorrelated decision trees, which are then merged together to reduce variance and create more accurate data predictions. It is a classifier which 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.[4]



[Fig - 4 Working of random forest algorithm [Machine Learning Random Forest Algorithm - Javatpoint]

Understanding Energy Consumption for Appliances

In this time of global uncertainty world needs energy and in increasing quantities to support economic and social progress and build a better quality of life, in particular in developing countries. But even in today’s time there are many places especially in developing world where there are outages. Power outages pose significant economic and social impacts on communities around the world. The increasing reliance of the society on electricity reduces the tolerance for power outages, and consequently highlights the need for enhancing the power grid resilience against natural hazards. Although these hazards are low-probability events, even one occurrence of an extreme event can be catastrophic for infrastructure systems. These outages are primary because of excess load consumed by appliances at home. Heating and cooling appliances takes most power in house. In this project we will be analysing the appliance usage in the house gathered via home sensors. All readings are taken at 10 mins intervals for 4.5 months. The goal is to predict energy consumption by appliances.[6]

In the age of smart homes, ability to predict energy consumption can not only save money for end user but can also help in generating money for user by giving excess energy back to Grid (in case of solar panels usage). In this case regression analysis will be used to predict Appliance energy usage based on data collected from various sensors.

The Problem in Front of Us

We should predict Appliance energy consumption for a house based on factors like temperature, humidity & pressure. In order to achieve this, we need to develop a supervised learning model using regression algorithms. Regression algorithms are used as data consist of continuous features and there are no identification of appliances in dataset

The Dataset Description

The date sets can be downloaded from Kaggle .

There are 29 features to describe appliances energy use :

1. date : time year-month-day hour:minute:second

2. lights : energy use of light fixtures in the house in Wh

3. T1 : Temperature in kitchen area, in Celsius

4. T2 : Temperature in living room area, in Celsius

5. T3 : Temperature in laundry room area

6. T4 : Temperature in office room, in Celsius

7. T5 : Temperature in bathroom, in Celsius

8. T6 : Temperature outside the building (north side), in Celsius

9. T7 : Temperature in ironing room, in Celsius

10.T8 : Temperature in teenager room 2, in Celsius

11. T9 : Temperature in parents' room, in Celsius

12. T_out : Temperature outside (from Chievres weather station), in Celsius

13. Tdewpoint : (from Chievres weather station), Â°C

14. RH_1 : Humidity in kitchen area, in %

15. RH_2 : Humidity in living room area, in %

16. RH_3 : Humidity in laundry room area, in %

17. RH_4 : Humidity in office room, in %

18. RH_5 : Humidity in bathroom, in %

19. RH_6 : Humidity outside the building (north side), in %

20. RH_7 : Humidity in ironing room, in %

21. RH_8 : Humidity in teenager room 2, in %

22. RH_9 : Humidity in parents' room, in %

23. RH_out :Humidity outside (from Chievres weather station), in %

24. Pressure : (from Chievres weather station), in mm Hg

25. Wind speed: (from Chievres weather station), in m/s

26. Visibility :(from Chievres weather station), in km

27. Rv1 :Random variable 1, non-dimensional

28. Rv2 :Random variable 2, non-dimensional

29. Appliances : Total energy used by appliances, in Wh

The dataset was collected by sensors placed inside the house and outside readings came from the nearby weather station. The main attributes are temperature, humidity and pressure readings. Each observation measures electricity in a 10-minute interval. The temperatures and humidity have been averaged for 10-minute intervals.

The Process that we are going to follow

- Looking for Energy related dataset on UCI Machine Learning repository and Kaggle where in benchmark numbers are available.
- Deciding between Classification and Regression problems.
- Visualized the data, do the pre-processing and EDA by learning from other regression contests from Kaggle.
- Pre-processing the data and feature selection. Look for correlation between features
- Deciding the regression algorithms to be used to solve the problem.
- Using Time series split Cross validation method to create benchmark model.
- Applying selected algorithms and visualizing the results.
- Comparing all the models to consider the best model for the problem

Clearly looking at the data from different sites and after understanding how the values are changing, we can clearly say the this is a regression problem so our further approach will be according to that

<u>Key Observations from the First Look at The Dataset:</u>

- No. of Independent variables are 28(11 temperature, 10 humidity, 1 pressure, 2 randoms) and only one dependent variable(Appliances)
- Date column is only used for understanding the consumption vs date time behaviour and given this is not a time series problem it can be removed. We can add one more column temporarily (WEEKDAY)which focuses on if a day was weekday or weekend in order to check the difference in appliance consumption
- Light column can also be removed as there are readings of submeter and we are not focusing on appliance specific reading
- Number of Independent variables at this stage — 26
- Number of Dependent variables at this stage — 1
- Total number of rows — 19735
- The data set will be split 75−25 % between train & test.
- Total no. of rows in training set — 14801
- Total no, of rows in test set — 4934
- All the features have numerical values. There are no categorical or ordinal features.

EDA or Exploratory Data Analysis For Appliance Energy Prediction Data

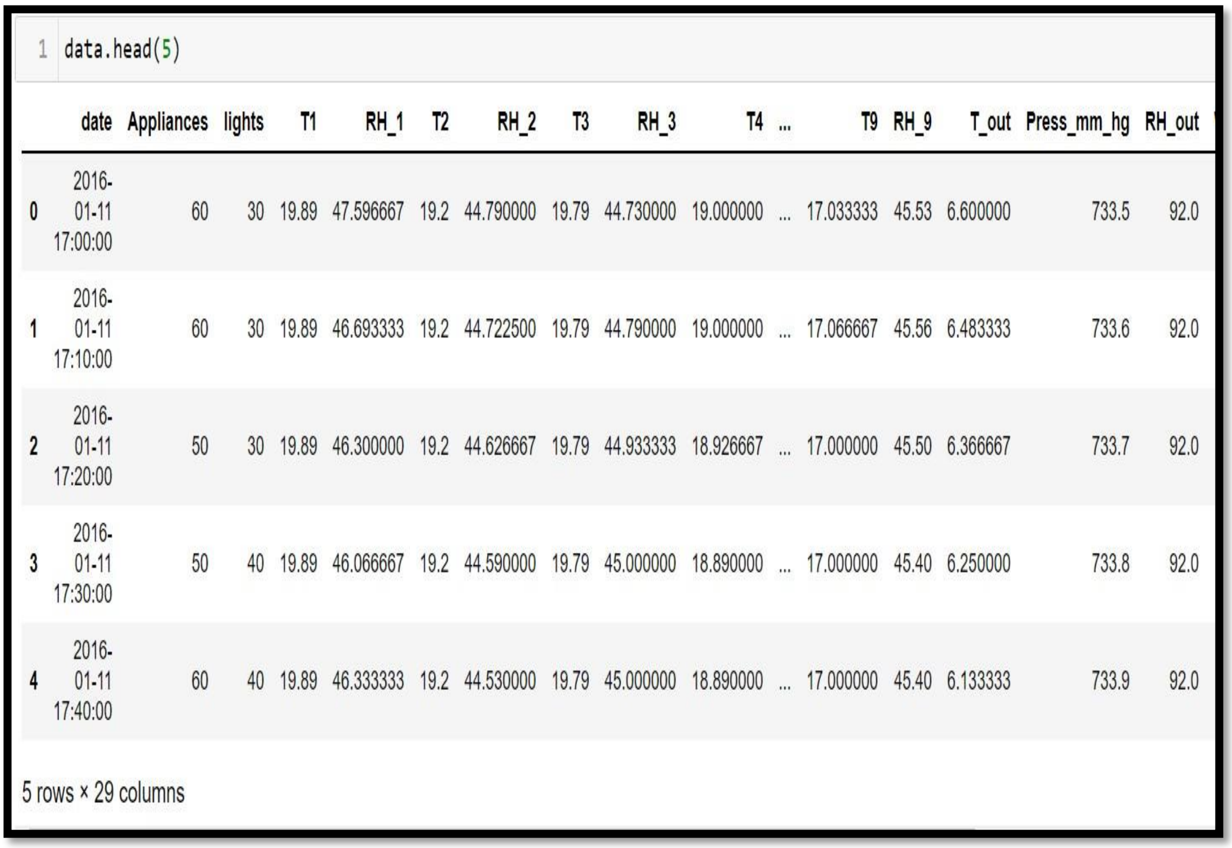
Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns to spot anomalies to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. It is a good practice to understand the data first and try to gather as many insights from it.[7] EDA is primarily used to see what data can reveal beyond the formal modelling or hypothesis testing task and provides a provides a better understanding of data set variables and the relationships between them. It can also help determine if the statistical techniques you are considering for data analysis are appropriate.

Why is exploratory data analysis important in data science?

The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, find interesting relations among the variables.

Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals. EDA also helps stakeholders by confirming they are asking the right questions. EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is complete and insights are drawn, its features can then be used for more sophisticated data analysis or modelling, including machine learning.[8]

So, we have also done EDA of our test and train data that we got directly form Kaggle. To have a look at the whole data set, we have used data.head() function from the pandas library. This will show us the first five rows of the dataset.

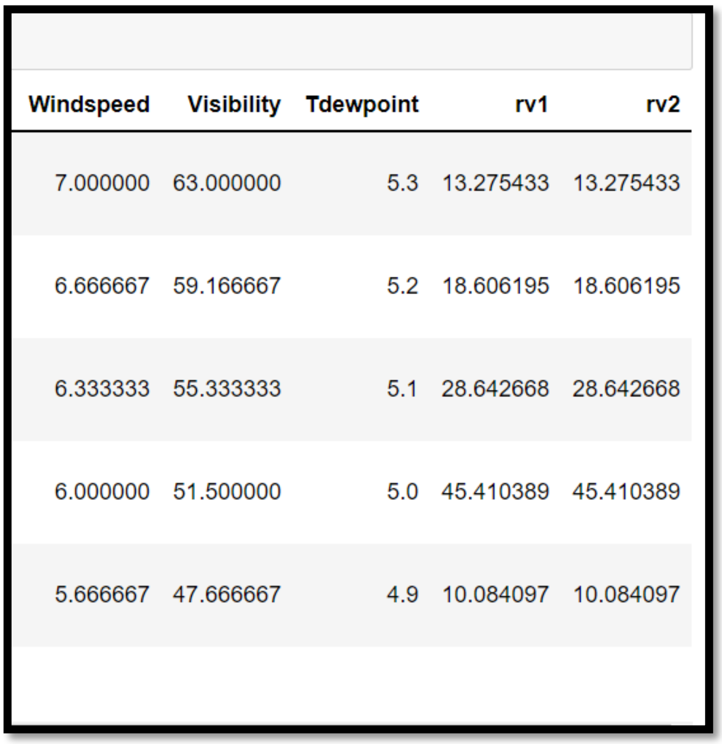


```
data.head(5)
```

	date	Appliances	lights	T1	RH_1	T2	RH_2	T3	RH_3	T4	...	T9	RH_9	T_out	Press_mm_hg	RH_out
0	2016-01-11 17:00:00	60	30	19.89	47.596667	19.2	44.790000	19.79	44.730000	19.000000	...	17.033333	45.53	6.600000	733.5	92.0
1	2016-01-11 17:10:00	60	30	19.89	46.693333	19.2	44.722500	19.79	44.790000	19.000000	...	17.066667	45.56	6.483333	733.6	92.0
2	2016-01-11 17:20:00	50	30	19.89	46.300000	19.2	44.626667	19.79	44.933333	18.926667	...	17.000000	45.50	6.366667	733.7	92.0
3	2016-01-11 17:30:00	50	40	19.89	46.066667	19.2	44.590000	19.79	45.000000	18.890000	...	17.000000	45.40	6.250000	733.8	92.0
4	2016-01-11 17:40:00	60	40	19.89	46.333333	19.2	44.530000	19.79	45.000000	18.890000	...	17.000000	45.40	6.133333	733.9	92.0

5 rows × 29 columns

Fig-5 (a)

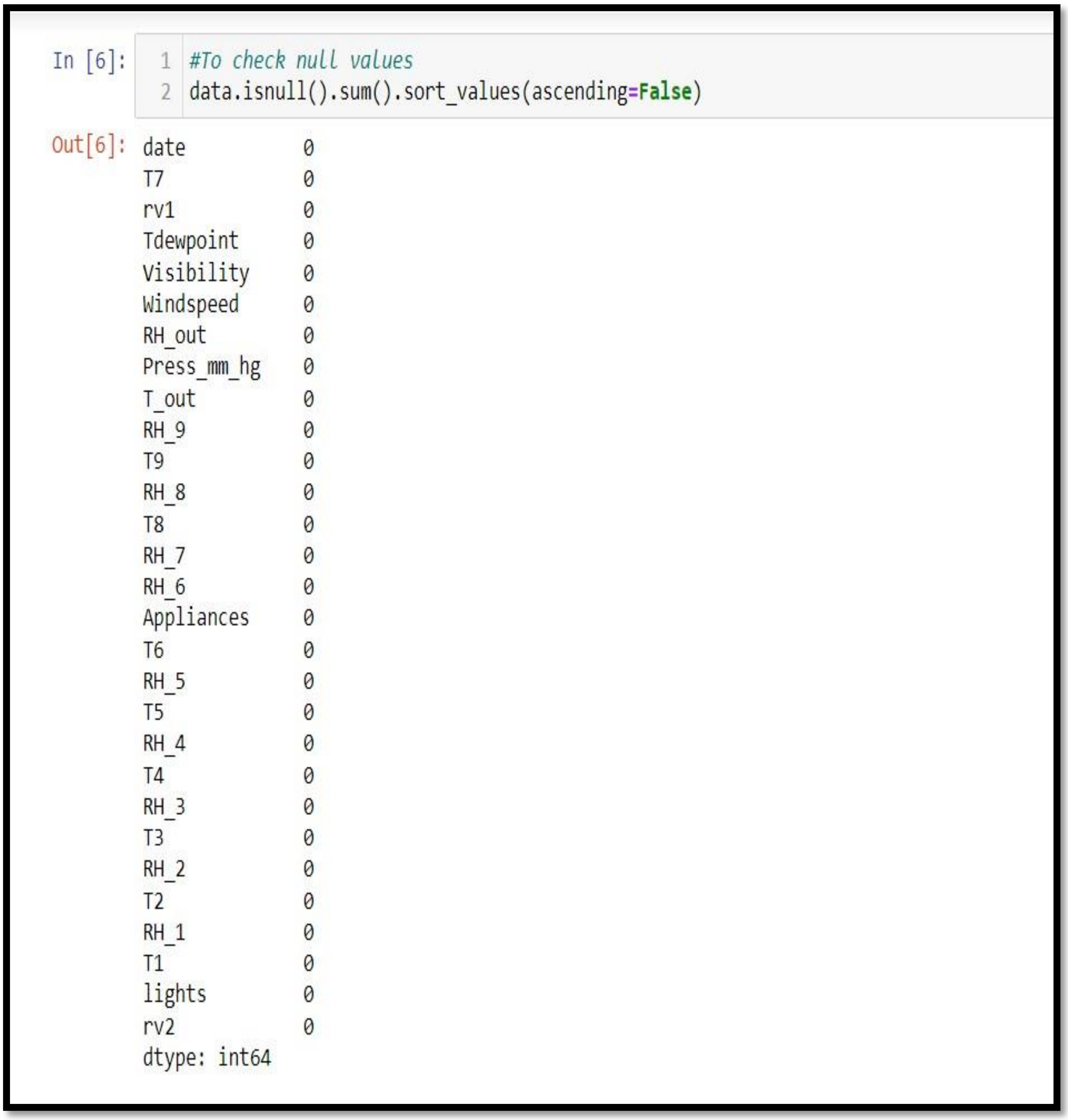


Windspeed	Visibility	Tdewpoint	rv1	rv2
7.000000	63.000000	5.3	13.275433	13.275433
6.666667	59.166667	5.2	18.606195	18.606195
6.333333	55.333333	5.1	28.642668	28.642668
6.000000	51.500000	5.0	45.410389	45.410389
5.666667	47.666667	4.9	10.084097	10.084097

Fig -5(b)

[Fig 5a and 5b show the result of head function applied on dataset]

As handling missing values is an essential part of data cleaning and preparation process because almost all data in real life comes with some missing values. So, our Next step will be check for null values so that we can get more clear picture of the dataset

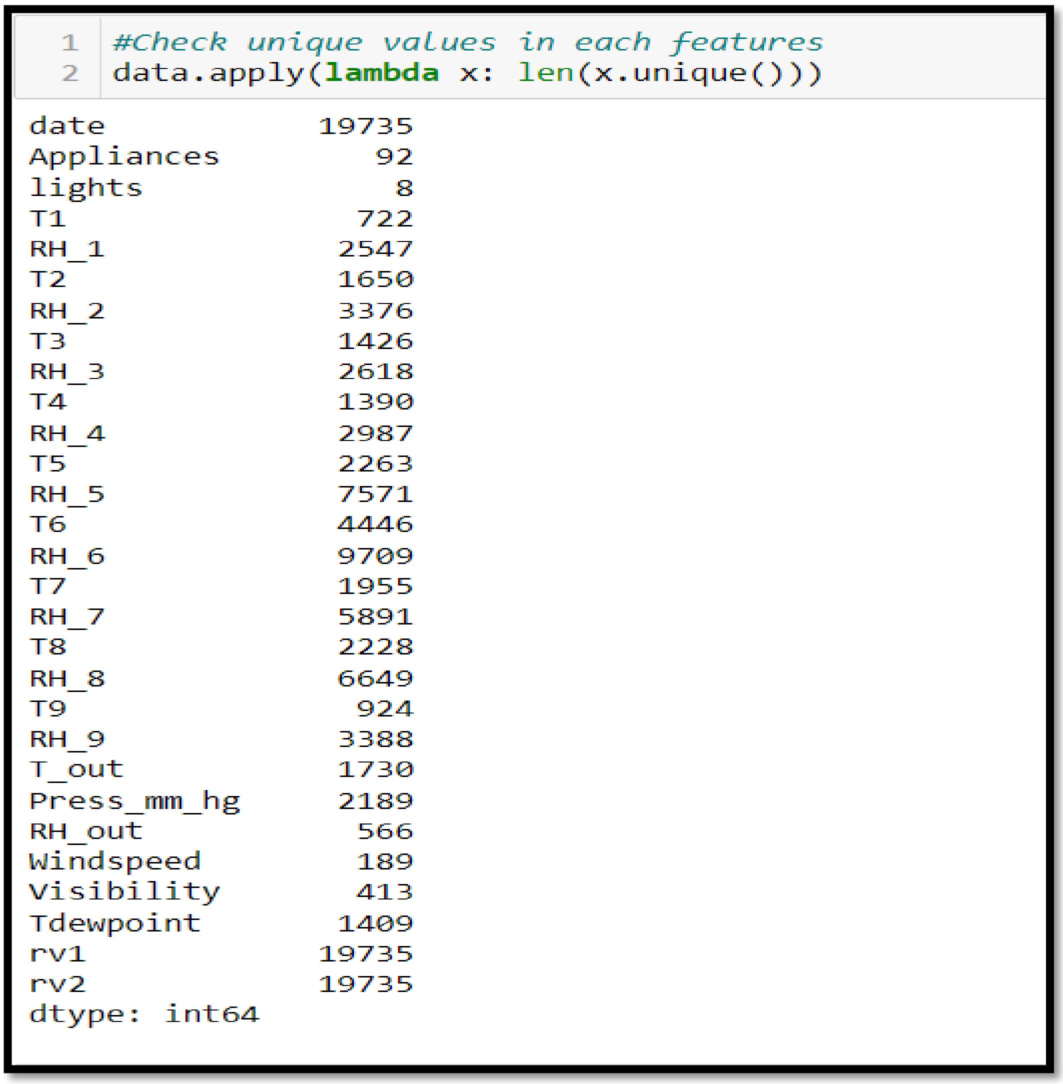


```
In [6]:  #To check null values
         data.isnull().sum().sort_values(ascending=False)

Out[6]:  date           0
         T7             0
         rv1            0
         Tdewpoint      0
         Visibility     0
         Windspeed      0
         RH_out         0
         Press_mm_hg    0
         T_out          0
         RH_9           0
         T9             0
         RH_8           0
         T8             0
         RH_7           0
         RH_6           0
         Appliances     0
         T6             0
         RH_5           0
         T5             0
         RH_4           0
         T4             0
         RH_3           0
         T3             0
         RH_2           0
         T2             0
         RH_1           0
         T1             0
         lights         0
         rv2            0
         dtype: int64
```

[Fig 6 - Result of .isnull() function to check null values]

This result is clearly showing that we have no null values for the dataset which is a good sign. If there are missing values in the dataset then generally according to our requirement, we can either drop them or replace them using mode or mean of that feature. But as in our case it is not needed.

Now we will check the unique values or we can say the repeated values of each features using the unique function. This will help in identifying how scattered will be the range of variable.



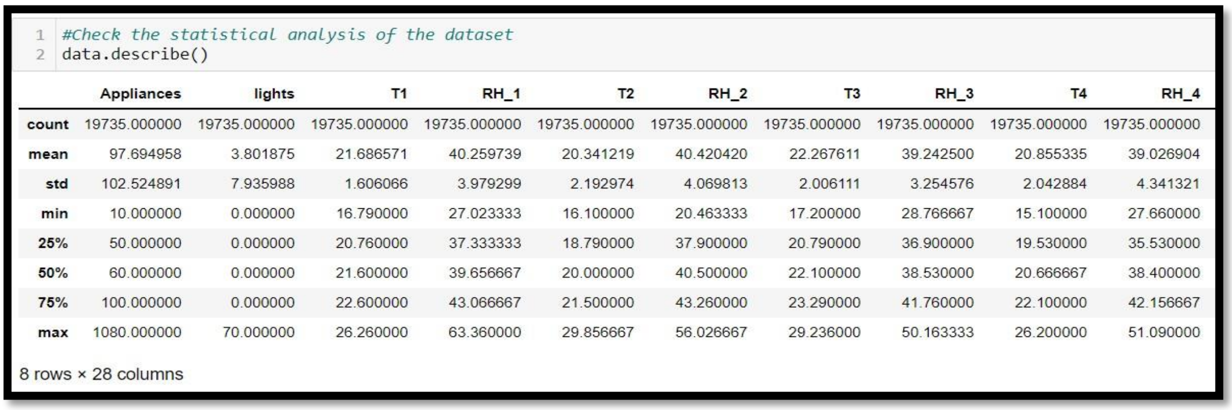
```
#Check unique values in each features
data.apply(lambda x: len(x.unique()))
```

```
date            19735
Appliances         92
lights              8
T1                722
RH_1             2547
T2               1650
RH_2             3376
T3               1426
RH_3             2618
T4               1390
RH_4             2987
T5               2263
RH_5             7571
T6               4446
RH_6             9709
T7               1955
RH_7             5891
T8               2228
RH_8             6649
T9                924
RH_9             3388
T_out            1730
Press_mm_hg      2189
RH_out            566
Windspeed         189
Visibility        413
Tdewpoint        1409
rv1             19735
rv2             19735
dtype: int64
```

[Fig 7 – Result of .unique function on the dataset to check for unique values]

All the unique values are listed in front of each feature. Remaining values are repeated. As we can see lights column has least unique values

Now to get the statistical overview of the dataset we will be using describe function. The describe() method is used for calculating some statistical data like **percentile, mean** and **std** of the numerical values of the Series or DataFrame. It analyzes both numeric and object series and also the DataFrame column sets of mixed data types. It returns the statistical summary of the Series and DataFrame.[9]

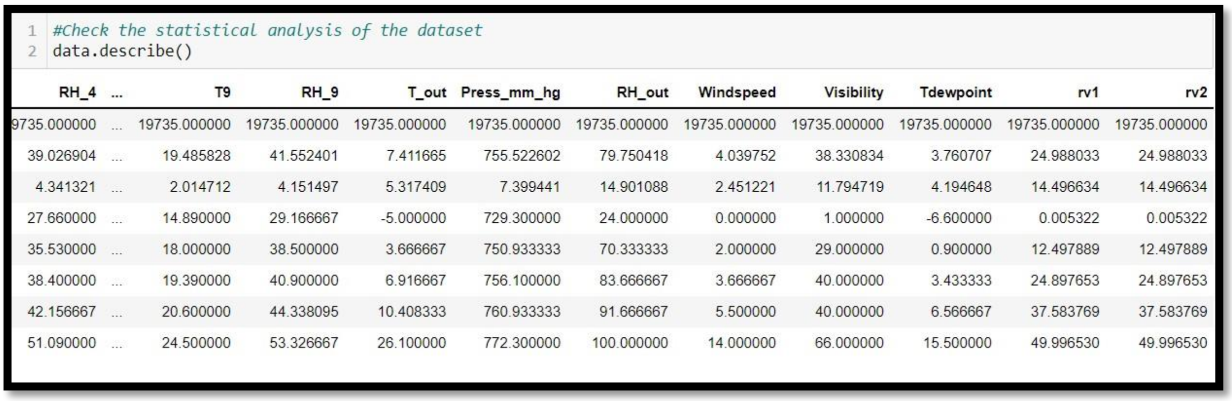


```
#Check the statistical analysis of the dataset
data.describe()
```

	Appliances	lights	T1	RH_1	T2	RH_2	T3	RH_3	T4	RH_4
count	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000
mean	97.694958	3.801875	21.686571	40.259739	20.341219	40.420420	22.267611	39.242500	20.855335	39.026904
std	102.524891	7.935988	1.606066	3.979299	2.192974	4.069813	2.006111	3.254576	2.042884	4.341321
min	10.000000	0.000000	16.790000	27.023333	16.100000	20.463333	17.200000	28.766667	15.100000	27.660000
25%	50.000000	0.000000	20.760000	37.333333	18.790000	37.900000	20.790000	36.900000	19.530000	35.530000
50%	60.000000	0.000000	21.600000	39.656667	20.000000	40.500000	22.100000	38.530000	20.666667	38.400000
75%	100.000000	0.000000	22.600000	43.066667	21.500000	43.260000	23.290000	41.760000	22.100000	42.156667
max	1080.000000	70.000000	26.260000	63.360000	29.856667	56.026667	29.236000	50.163333	26.200000	51.090000

8 rows × 28 columns

Fig 8(a)



```
#Check the statistical analysis of the dataset
data.describe()
```

RH_4	...	T9	RH_9	T_out	Press_mm_hg	RH_out	Windspeed	Visibility	Tdewpoint	rv1	rv2
9735.000000	...	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000
39.026904	...	19.485828	41.552401	7.411665	755.522602	79.750418	4.039752	38.330834	3.760707	24.988033	24.988033
4.341321	...	2.014712	4.151497	5.317409	7.399441	14.901088	2.451221	11.794719	4.194648	14.496634	14.496634
27.660000	...	14.890000	29.166667	-5.000000	729.300000	24.000000	0.000000	1.000000	-6.600000	0.005322	0.005322
35.530000	...	18.000000	38.500000	3.666667	750.933333	70.333333	2.000000	29.000000	0.900000	12.497889	12.497889
38.400000	...	19.390000	40.900000	6.916667	756.100000	83.666667	3.666667	40.000000	3.433333	24.897653	24.897653
42.156667	...	20.600000	44.338095	10.408333	760.933333	91.666667	5.500000	40.000000	6.566667	37.583769	37.583769
51.090000	...	24.500000	53.326667	26.100000	772.300000	100.000000	14.000000	66.000000	15.500000	49.996530	49.996530

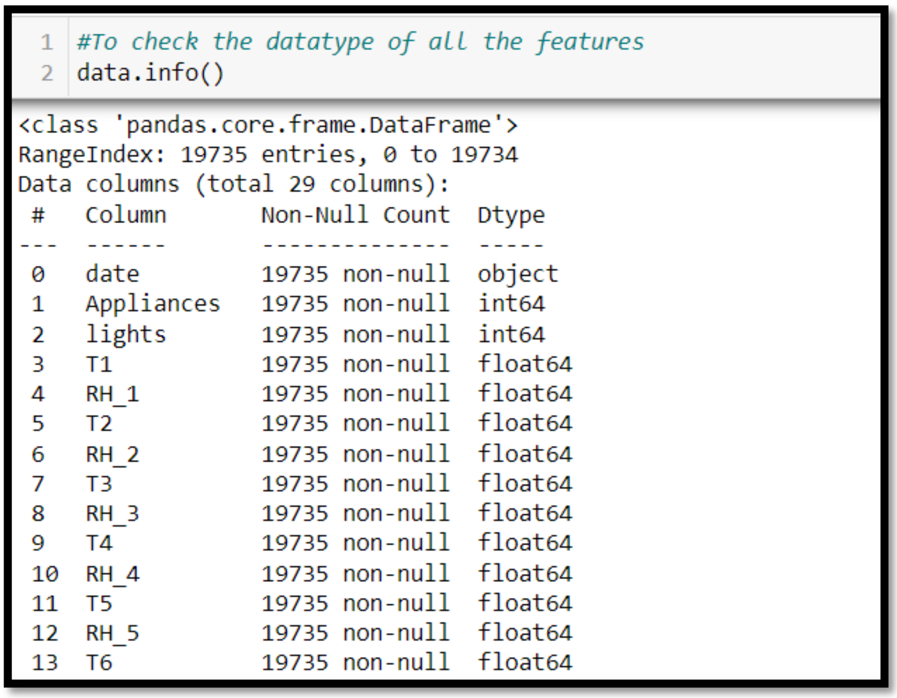
Fig8(b)

[Fig 8(a) and 8(b) shows the result of describe function applied on the complete dataset]

Feature ranges that we can tell from the above describe function are as follows

- Temperature : -6 to 30 deg
- Humidity : 1 to 100 %
- Windspeed : 0 to 14 m/s
- Visibility : 1 to 66 km
- Pressure : 729 to 772 mm Hg
- Appliance Energy Usage : 10 to 1080 Wh

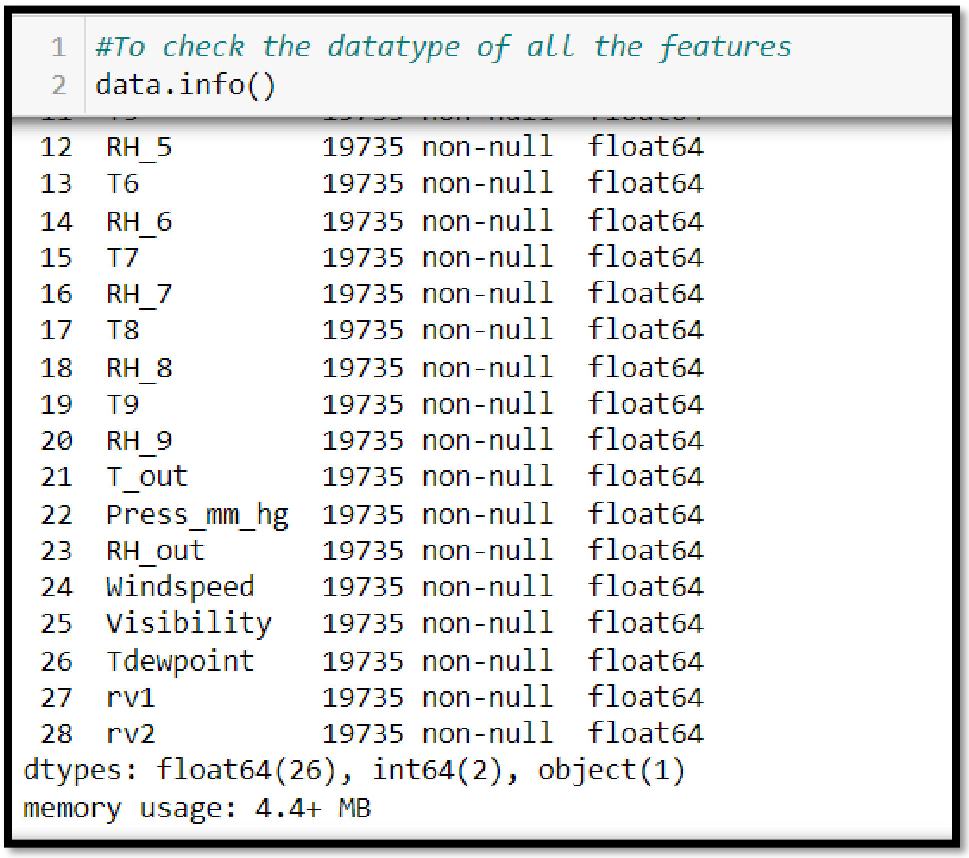
Now to check the information about the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values), we have used data.info method and the results are as follows.



```
#To check the datatype of all the features
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19735 entries, 0 to 19734
Data columns (total 29 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   date         19735 non-null  object
 1   Appliances   19735 non-null  int64
 2   lights       19735 non-null  int64
 3   T1           19735 non-null  float64
 4   RH_1         19735 non-null  float64
 5   T2           19735 non-null  float64
 6   RH_2         19735 non-null  float64
 7   T3           19735 non-null  float64
 8   RH_3         19735 non-null  float64
 9   T4           19735 non-null  float64
 10  RH_4         19735 non-null  float64
 11  T5           19735 non-null  float64
 12  RH_5         19735 non-null  float64
 13  T6           19735 non-null  float64
```

Fig 9a



```
#To check the datatype of all the features
data.info()
```

```
 12  RH_5         19735 non-null  float64
 13  T6           19735 non-null  float64
 14  RH_6         19735 non-null  float64
 15  T7           19735 non-null  float64
 16  RH_7         19735 non-null  float64
 17  T8           19735 non-null  float64
 18  RH_8         19735 non-null  float64
 19  T9           19735 non-null  float64
 20  RH_9         19735 non-null  float64
 21  T_out        19735 non-null  float64
 22  Press_mm_hg  19735 non-null  float64
 23  RH_out       19735 non-null  float64
 24  Windspeed    19735 non-null  float64
 25  Visibility   19735 non-null  float64
 26  Tdewpoint    19735 non-null  float64
 27  rv1          19735 non-null  float64
 28  rv2          19735 non-null  float64
dtypes: float64(26), int64(2), object(1)
memory usage: 4.4+ MB
```

Fig 9(b)

[Fig 9(a) and 9(b)- Result of data.info() applied on the whole dataset]

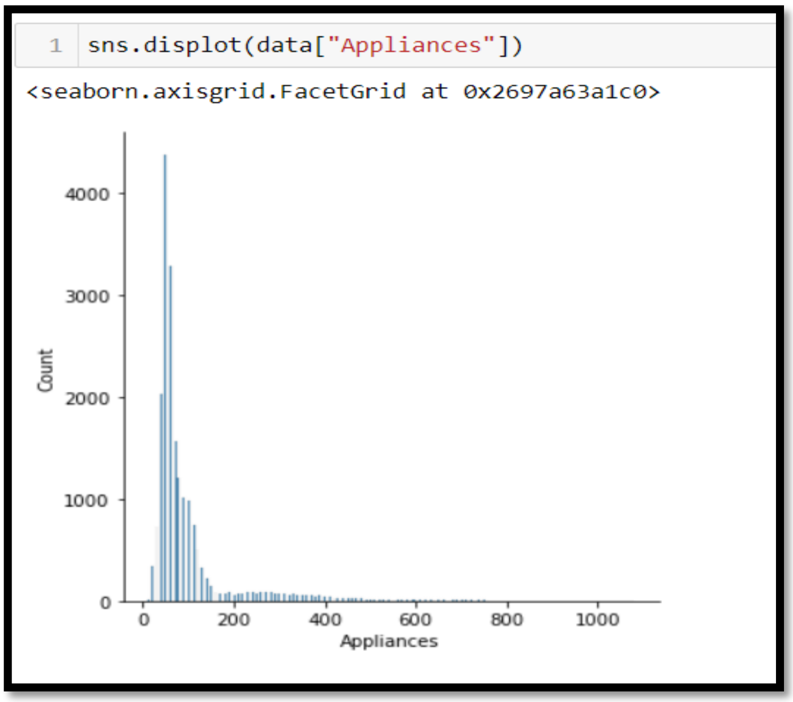
So, from above we conclude that there is only one feature which has an object data type and that is date. Apart from that Appliance and Lights are of int type and rest are float type. Total memory usage is also given for the dataset and moreover we can also see that these all features are not null.

Data Visualization

Now moving to data visualization part so Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data. In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.[10]

It is basically another form of visual art that grabs our interest and keeps our eyes on the message. When we see a chart, we quickly see trends and outliers. If we can see something, we internalize it quickly. It's storytelling with a purpose. If you've ever stared at a massive spreadsheet of data and couldn't see a trend, you know how much more effective a visualization can be.

So, to start with we will see the distribution of our target variable using displot. It is a function in seaborn library which provides access to several approaches for visualizing the univariate or bivariate distribution of data, including subsets of data



```
sns.displot(data["Appliances"])
```

```
<seaborn.axisgrid.FacetGrid at 0x2697a63a1c0>
```

Fig 10 – Displot of the energy consumption in the appliances

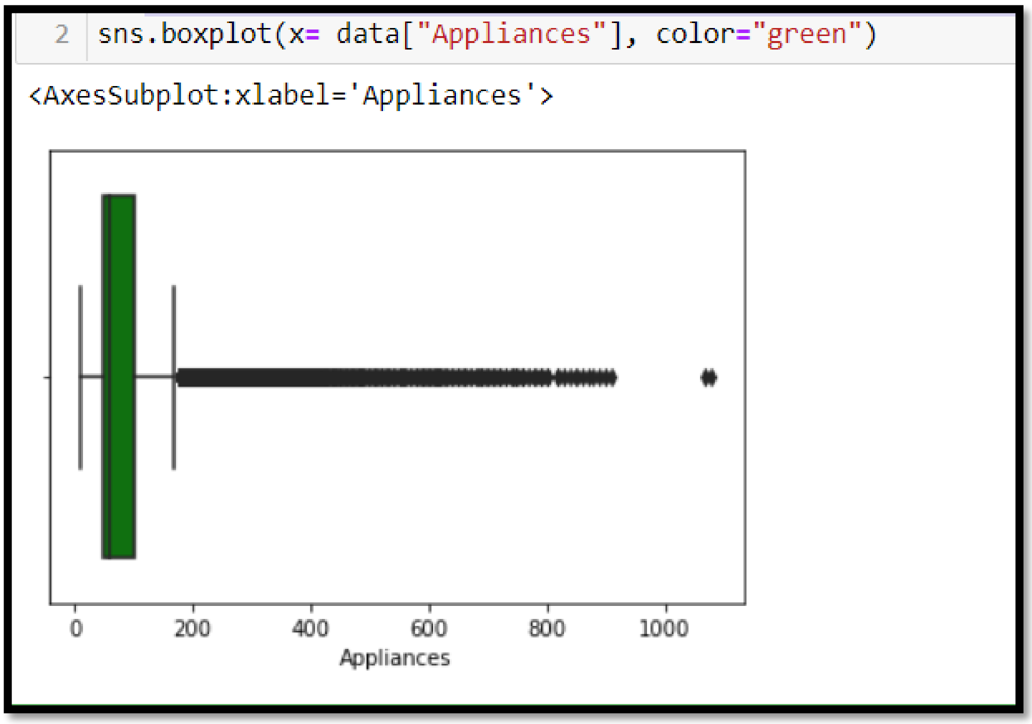
As we can see the plot of appliances is skew in nature so clearly there will be outliers in appliances part so we will use box plot to visualize that. A box plot (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables. The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution. The box plot (a.k.a. box and whisker diagram) is a standardized way of displaying the distribution of data based on the five number summary:

- Minimum
- First quartile
- Median
- Third quartile
- Maximum.

In the simplest box plot the central rectangle spans the first quartile to the third quartile (the interquartile range or IQR).

A segment inside the rectangle shows the median and “whiskers” above and below the box show the locations of the minimum and maximum.

Outliers are either 1.5×IQR or more above the third quartile or 1.5×IQR or more below the first quartile.



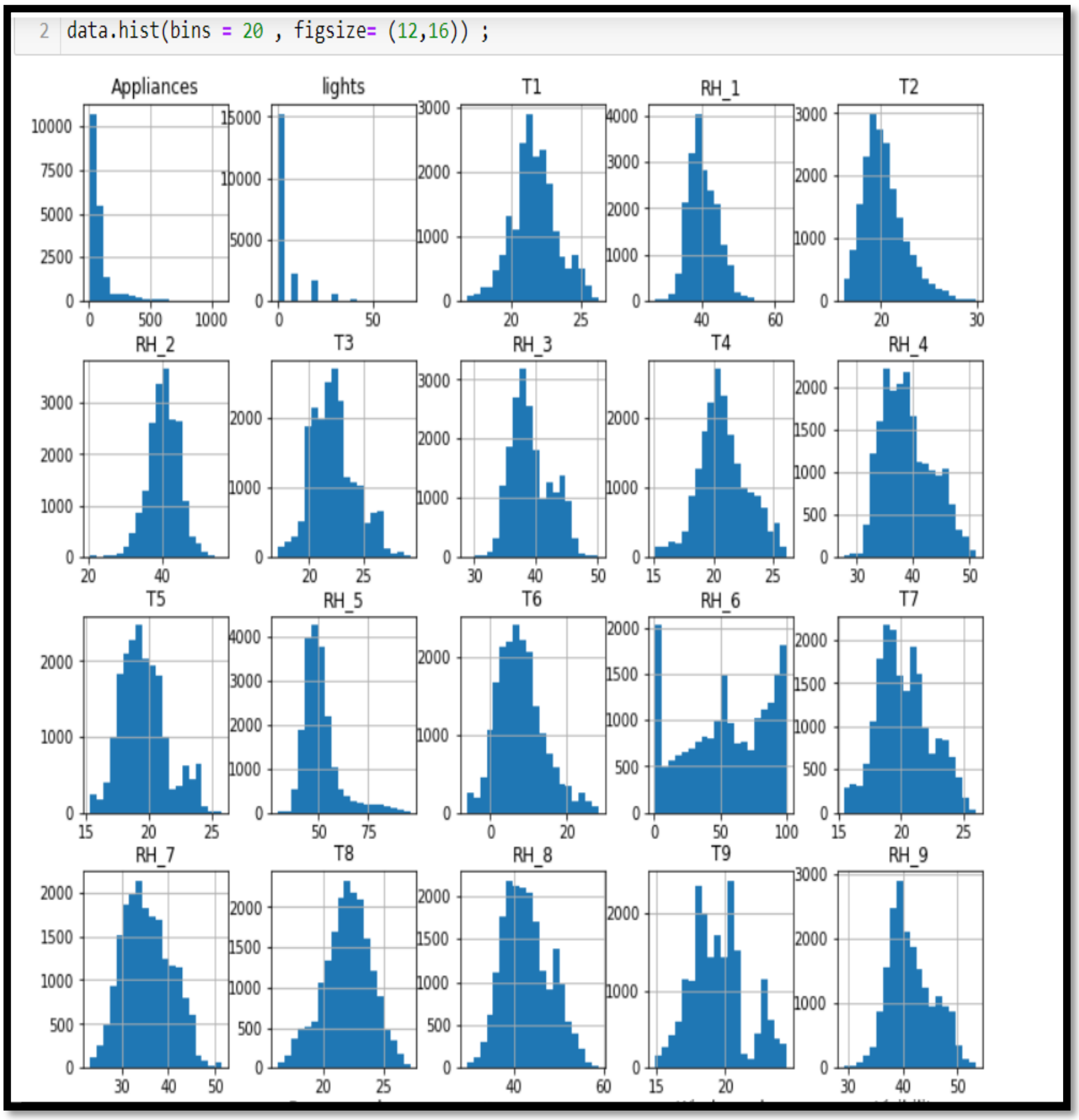
```
sns.boxplot(x= data["Appliances"], color="green")
```

```
<AxesSubplot:xlabel='Appliances'>
```

[Fig – 11 Result of box plot for the Appliances column]

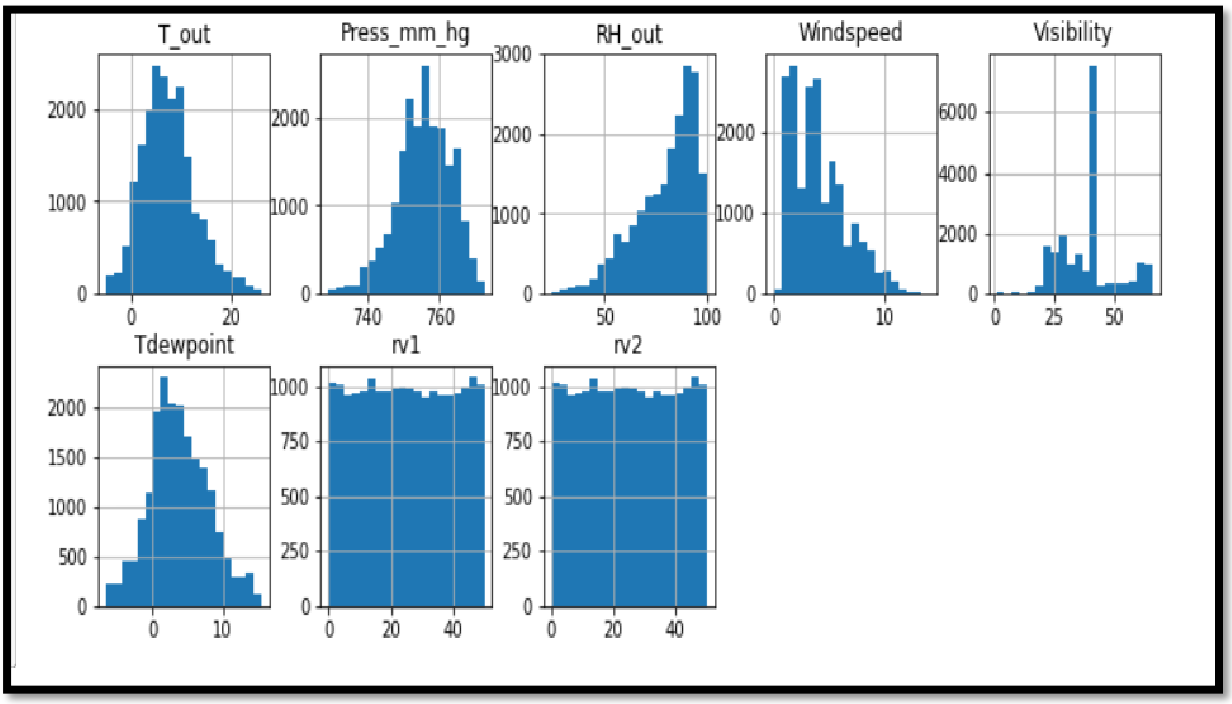
Clearly, we can see that there are lot of outliers in the dataset but we will also have to figure out that weather these outliers are just random or have some meaning. We will not simply remove these outliers without understanding the reason for their existence in this graph.

Now we will see the distribution of all the feature variables of the dataset using hist plot which is quite an important part of data visualization. As we know A histogram is a graph showing frequency distributions. It is a graph showing the number of observations within each given interval. In Matplotlib, we use the hist() function to create histograms. The hist() function will use an array of numbers to create a histogram, the array is sent into the function as an argument.



```
data.hist(bins = 20 , figsize= (12,16)) ;
```

[Fig – 12a Result of hist plot for all the column]



[Fig – 12a Result of hist plot for all the column]

Observations based on distribution plot

1. All humidity values except RH_6 and RHout follow a normal distribution.

2. All temperature readings follow a normal distribution except for T9.

3. From the columns which are remaining, we can observed that Visibility, Windspeed and Appliances are skewed.

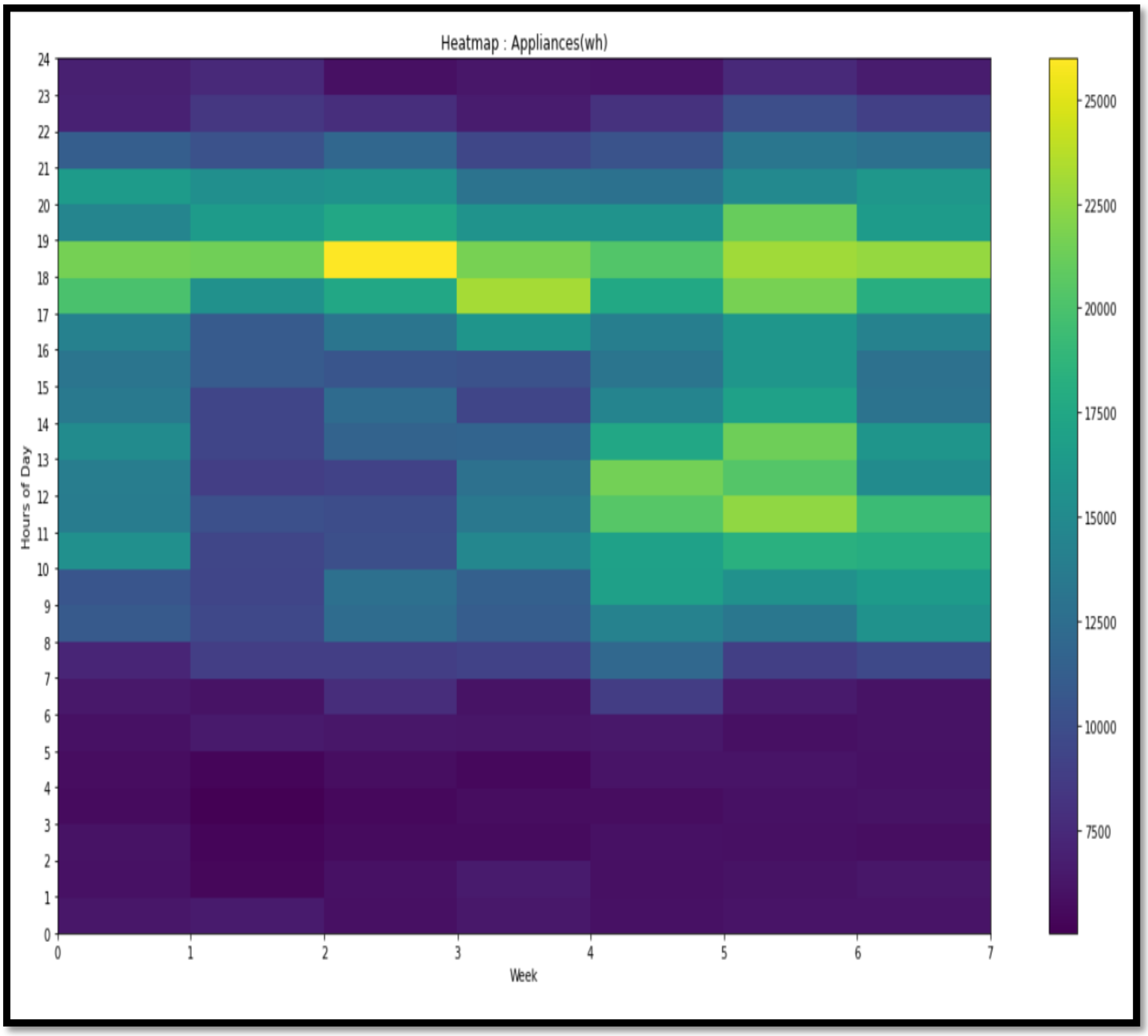
4. The random variables rv1 and rv2 have more or less the same values for all the recordings.

5. The target variable Appliances has most values less than 200Wh, showing that high energy consumption cases are very low.

6. No column has a distribution like the target variable Appliances.

Hence, we can clearly say that there are no independent feature with a linear relationship with the target.

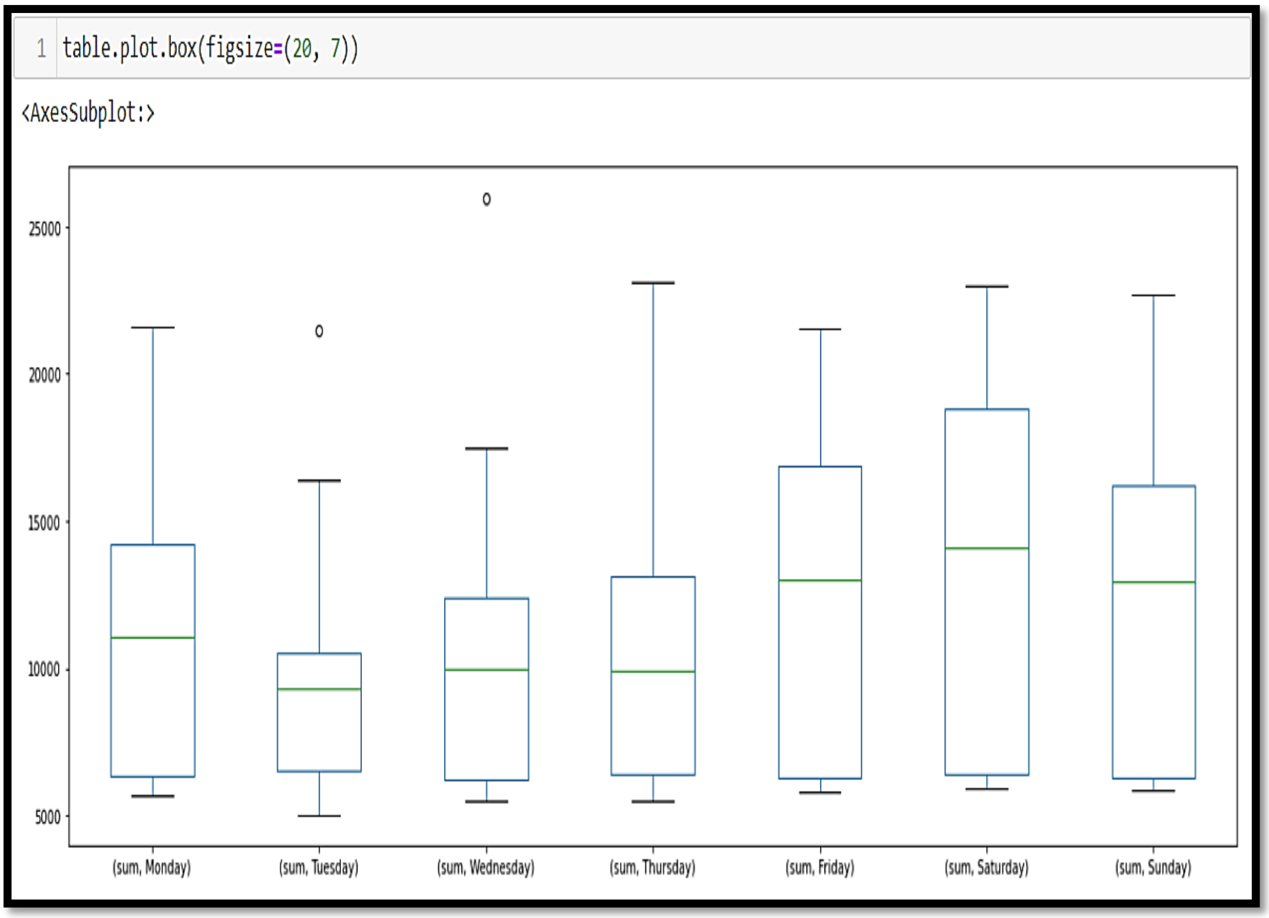
To check how the appliances energy consumption is related to weekdays and hours of the day, we are using a heatmap. Heatmap basically contains values representing various shades of the same colour for each value to be plotted. Usually, the darker shades of the chart represent higher values than the lighter shade. For a very different value a completely different colour can also be used.



[Fig – 13 Heatmap for energy consumption with respect to weekday and hours]

Conclusion from the above heatmap is that energy consumption is low during the early mornings and late-night hours because these are the sleeping hours and therefore less appliances will be working and hence energy consumption is low during these hours.

Now to check weekly energy consumption, we will be using box plot again. Now we can clearly see the outliers of each day and apart from that we can also see the median energy consumption of a particular day of a week. From the below boxplot we can also see that weather the maximum amount of energy consumed in a day is below the median energy consumed for that or above it.

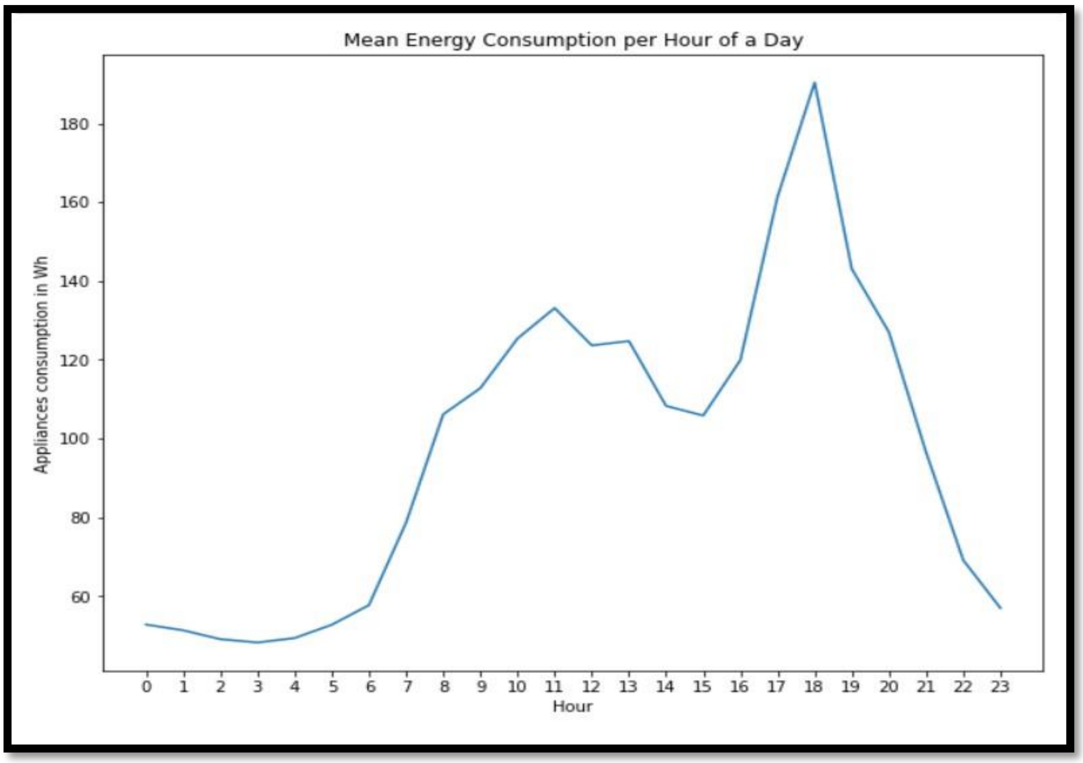


```
table.plot.box(figsize=(20, 7))
```

```
<AxesSubplot:>
```

[Fig – 14 Boxplot for energy consumption with respect to each weekday]

Mean Energy Consumption for the whole day using simple graph is as shown below and it proves our initial conclusions that in the early mornings and late nights energy consumption is very low. Moreover, energy consumption is high at evenings.



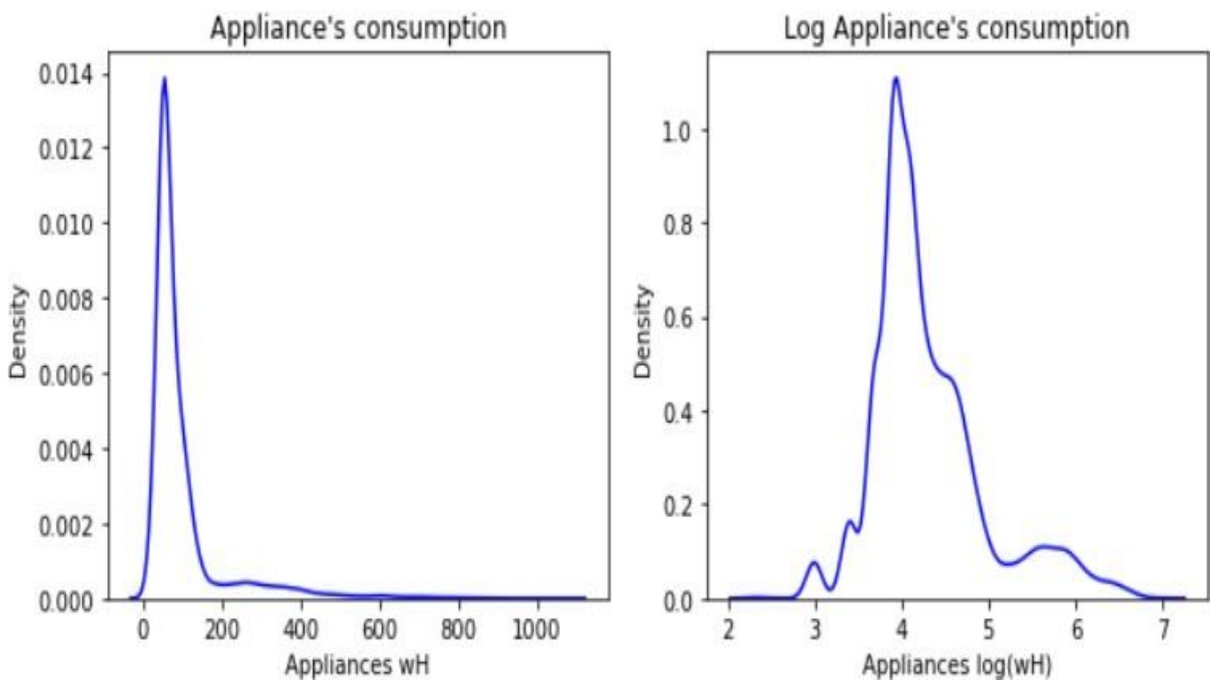
[Fig – 15 Graph of Mean Energy Consumption per hour of a day]

More clearly we can conclude as :-

- High electricity consumption is observed during evening hours between 16:00 to 20:00
- At night hours from 23:00 to 6:00, the power load is below 50Wh which is quite obvious as most appliances at this time will be off or on standby.
- Between 9 to 13, consumption is > 100wh as it is breakfast and after that the consumption is less than 100Wh.

Now as the appliances energy consumption graph was skewed therefore, I thought of logging that function so that the regression models give a better fit to that. As when we take the log of the data and it starts showing more of normal distribution behaviour so then we can take advantage of many features of a normal distribution, like well-defined mean, standard deviation (and hence z-scores), symmetry, etc.

```
Text(0.5, 0, 'Appliances log(wH)')
```



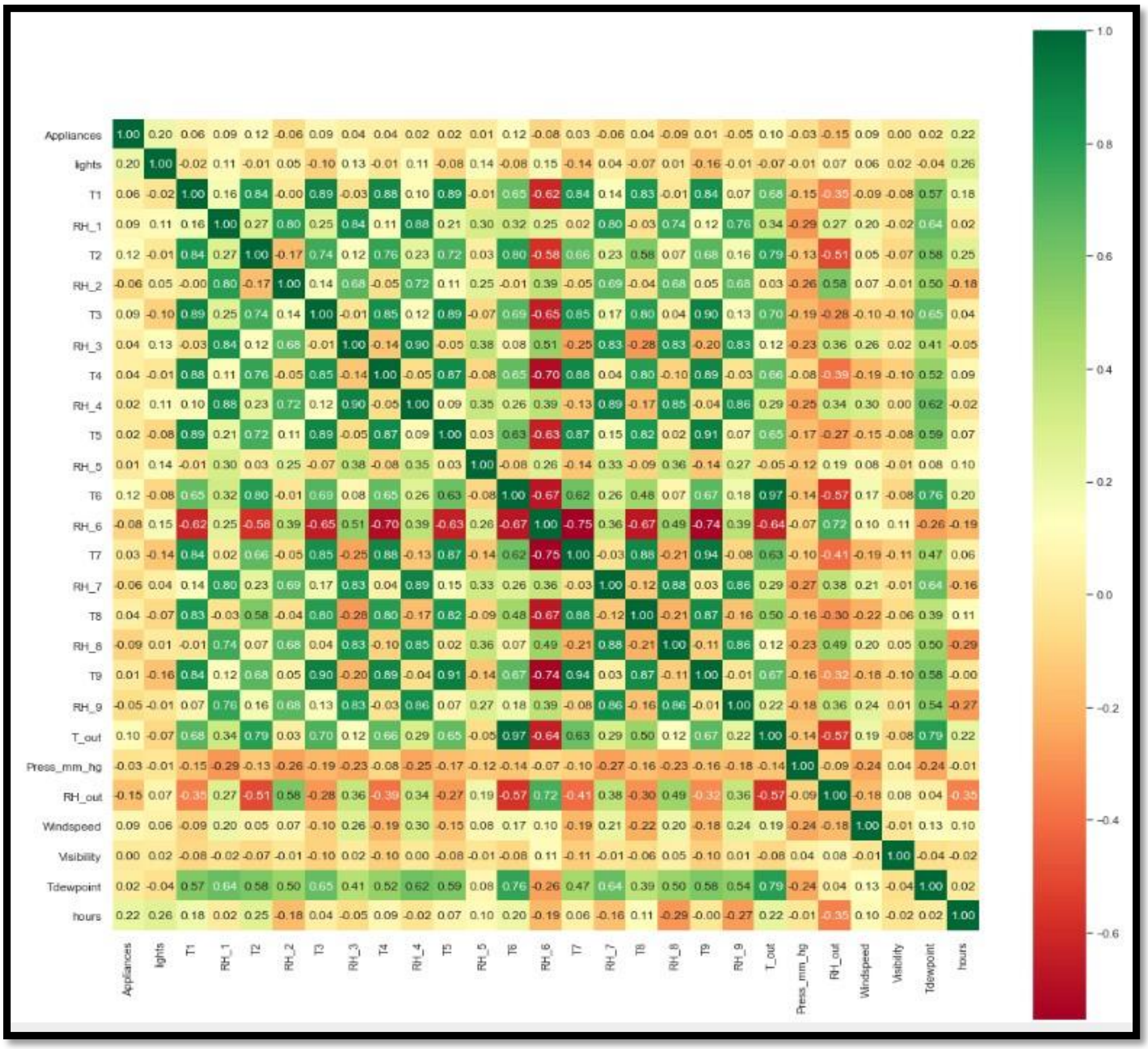
[Fig – 16 Graph of Energy Consumption and log Energy consumed by appliances per hour of a day]

As we can see from the above graph that Appliances consumption graph after logging is showing more sort of normal distribution then it will give more accuracy as compared to the original data.

Now to check Correlation among the variables, we will be using heatmaps. Below the col is a list of all the features included with the target variable.

```
col = ['Appliances', 'lights', 'T1', 'RH_1', 'T2', 'RH_2', 'T3', 'RH_3', 'T4',

    'RH_4', 'T5', 'RH_5', 'T6', 'RH_6', 'T7', 'RH_7', 'T8', 'RH_8', 'T9',

    'RH_9', 'T_out', 'Press_mm_hg', 'RH_out', 'Windspeed', 'Visibility',

    'Tdewpoint','hours']
```

The heatmap for the Pearson correlation for all these variables is shown below



[Fig – 17 Graph Of Pearson correlation between all the variables]

From above heatmap we can clearly conclude following things:-

- The energy consumption is highly correlated with
    1. Hours : 0.22
    2. Lights : 0.20
    3. T2 : 0.12
    4. T6 : 0.12

So here we were not getting a good analysis about the dataset therefore we thought of making the triangular correlation plot to check what we are missing. When building a machine learning model in real-life, it's almost rare that all the variables in the dataset are useful to build a model. Adding redundant variables reduces the generalization capability of the model and may also reduce the overall accuracy of a classifier. Furthermore, adding more and more variables to a model increases the overall complexity of the model. So now we will try to do some feature selection process for our dataset and therefore we combined all the data and segregated like columns together

col_temp = ["T1","T2","T3","T4","T5","T6","T7","T8","T9"]

col_hum = ["RH_1","RH_2","RH_3","RH_4","RH_5","RH_6","RH_7","RH_8","RH_9"]

col_weather = ["T_out", "Tdewpoint","RH_out","Press_mm_hg",

"Windspeed","Visibility"]

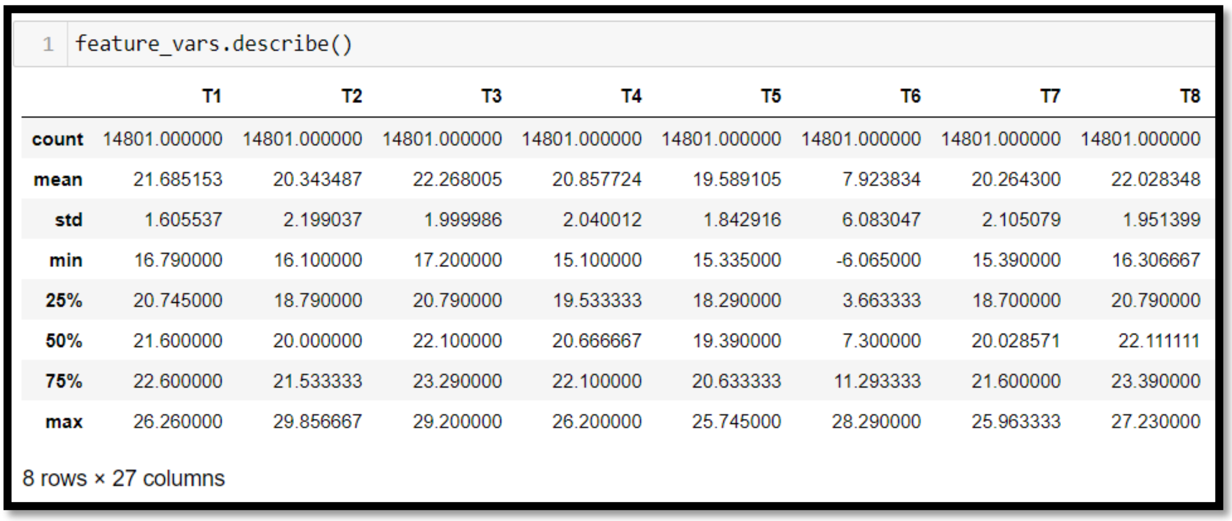
col_light = ["lights"]

col_randoms = ["rv1", "rv2"]

col_target = ["Appliances"]

Now we make two variable lists that is feature_vars which will contain the data of all the feature in above form and target_vars which will contain target data feature_vars = train[col_temp + col_hum + col_weather + col_light + col_randoms]

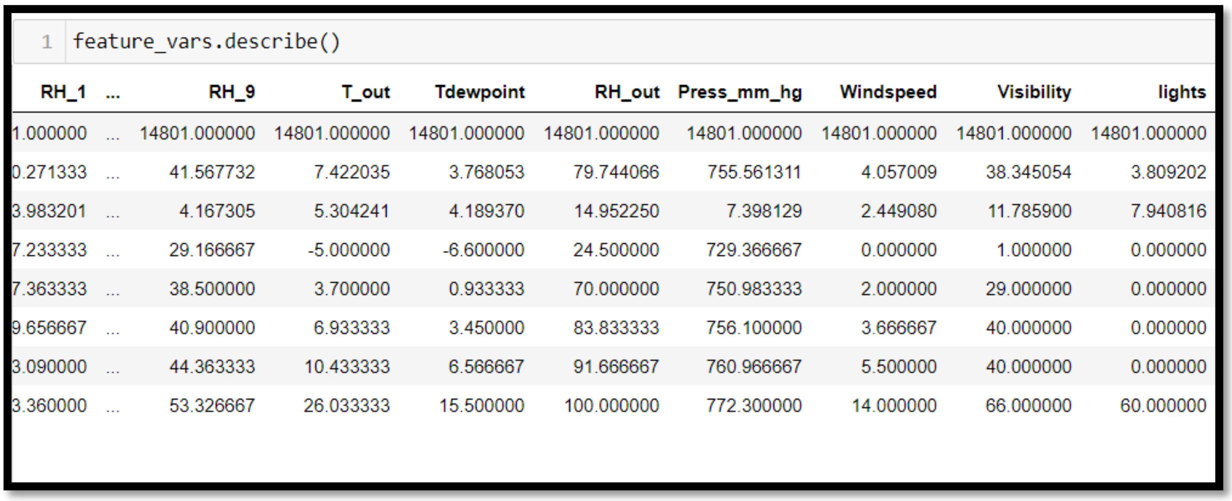
target_vars = train[col_target]



```
feature_vars.describe()
```

```
                T1            T2            T3            T4            T5            T6            T7            T8
count  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000
mean      21.685153     20.343487     22.268005     20.857724     19.589105      7.923834     20.264300     22.028348
std        1.605537      2.199037      1.999986      2.040012      1.842916      6.083047      2.105079      1.951399
min       16.790000     16.100000     17.200000     15.100000     15.335000     -6.065000     15.390000     16.306667
25%       20.745000     18.790000     20.790000     19.533333     18.290000      3.663333     18.700000     20.790000
50%       21.600000     20.000000     22.100000     20.666667     19.390000      7.300000     20.028571     22.111111
75%       22.600000     21.533333     23.290000     22.100000     20.633333     11.293333     21.600000     23.390000
max       26.260000     29.856667     29.200000     26.200000     25.745000     28.290000     25.963333     27.230000

8 rows × 27 columns
```

Fig – 18(a)

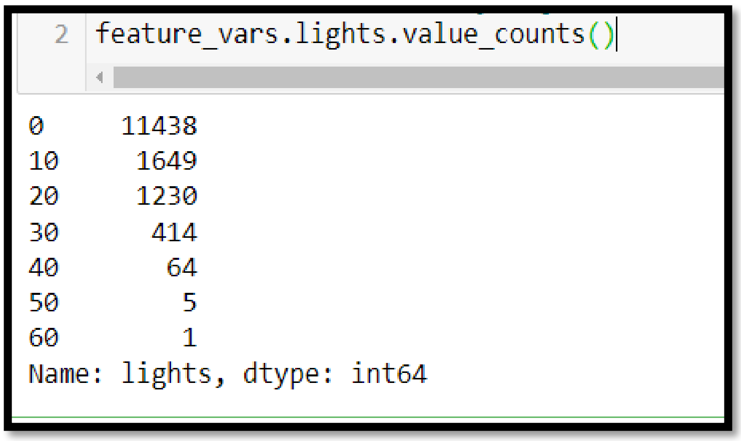


```
feature_vars.describe()
```

```
      RH_1  ...         RH_9         T_out     Tdewpoint        RH_out  Press_mm_hg     Windspeed    Visibility        lights
1.000000    ...  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000  14801.000000
0.271333    ...     41.567732      7.422035      3.768053     79.744066    755.561311      4.057009     38.345054      3.809202
3.983201    ...      4.167305      5.304241      4.189370     14.952250      7.398129      2.449080     11.785900      7.940816
7.233333    ...     29.166667     -5.000000     -6.600000     24.500000    729.366667      0.000000      1.000000      0.000000
7.363333    ...     38.500000      3.700000      0.933333     70.000000    750.983333      2.000000     29.000000      0.000000
9.656667    ...     40.900000      6.933333      3.450000     83.833333    756.100000      3.666667     40.000000      0.000000
3.090000    ...     44.363333     10.433333      6.566667     91.666667    760.966667      5.500000     40.000000      0.000000
3.360000    ...     53.326667     26.033333     15.500000    100.000000    772.300000     14.000000     66.000000     60.000000
```

[Fig – 18(b) Result of describe function on newly formed feature variables]

As from above result we can see that 75% of lights counts to 0. Therefore, we need to take its proper information is necessary so we are counting the values of lights to get the overview of lights distribution in quantity.



```
feature_vars.lights.value_counts()
```

```
0     11438
10     1649
20     1230
30      414
40       64
50        5
60        1
Name: lights, dtype: int64
```

[Fig – 19 Result of .value_counts() func on light column]

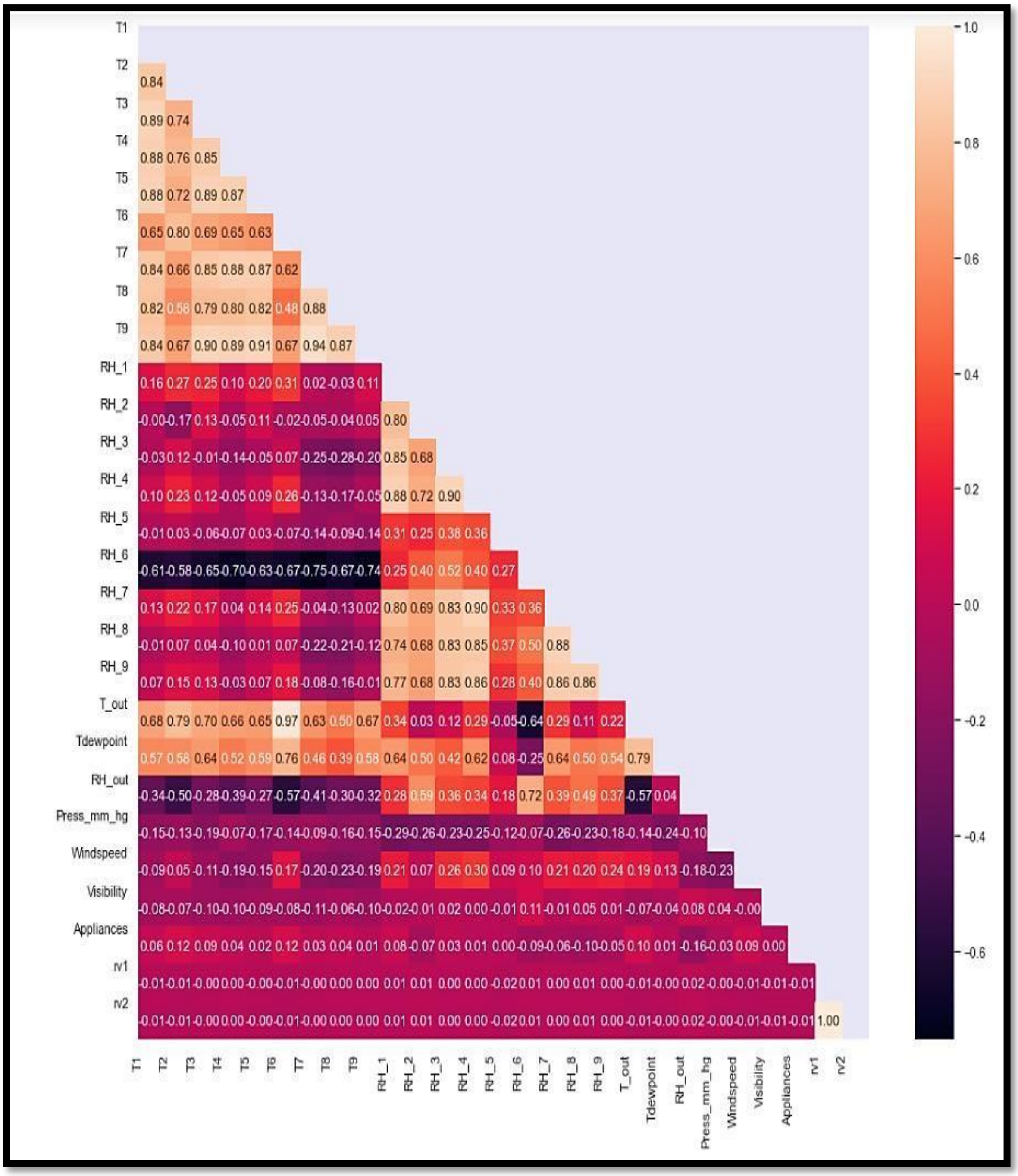
As we can clearly see from above . Approximately 11438 values of this column are 0 so variance of this column will be very less therefore we can think of removing this column as due to lot of zero entries light column is of not much use

Now we will see the weather, temperature, appliances and random column to see the correlation so that we can get the clear idea that if we can remove some other variable also. **Correlation test** is used to evaluate the association between two or more variables. Generally to make these correlation plots we use pearson correlation. The Pearson product-moment correlation coefficient (or Pearson correlation coefficient, for short) is a measure of the strength of a linear association between two variables and is denoted by *r*. Basically, a Pearson product-moment correlation attempts to draw a line of best fit through the data of two variables, and the Pearson correlation coefficient, *r*, indicates how far away all these data points are to this line of best fit (i.e., how well the data points fit this new model/line of best fit).[11]

The basic idea of heatmaps is that they replace numbers with colors of varying shades, as indicated by the scale on the right. Cells that are lighter have higher values of *r*. This type of visualization can make it much easier to spot linear relationships between variables than a table of numbers.[12]

So, for this new correlation plot we have made the following array train_corr

train_corr = train[col_temp + col_hum + col_weather +col_target+col_randoms] and now on masking the repeated values we get the following plot.



[Fig – 20 Result of Pearson correlation plot for train_corr components made above]

The conclusions and observations from above correlation plots are :-

- All the temperature variables from T1-T9 and Tout have positive correlation with the target appliances. For the indoor temperatures, the correlations are high as expected, since the ventilation is driven by the HRV unit and minimizes air temperature differences
- We can see a high degree of correlation between T9 and T3, T5, T7, T8 also T6 & Tout has high correlation (that is of 0.97) (both temperatures from outside). Hence T6 & T9 can be removed from training set as information provided by them can be provided by other fields.
- Weather attributes - Visibility, Tdewpoint, Press_mm_hg has low correlation values
- Random variables have no role to play
- The random variables rv1, rv2 and Visibility, Tdewpoint, Press_mm_hg have low correlation with the target variable.
- Due to above conclusions, I have decided to drop rv1, rv2, Visibility, T6,T9

Modelling Techniques

As we have clearly now chosen our important features so moving forward to the process, we can now split training and testing dataset into independent and dependent variables using the following formulations

train_X = train[feature_vars.columns]

train_y = train[target_vars.columns]

test_X = test[feature_vars.columns]

test_y = test[target_vars.columns]

As a result of feature selection and due to conclusion made above the necessary columns are being removed from both training and testing data. We have not removed the lights column as it was showing high correlation with the Appliances energy consumption.

train_X.drop(["rv1","rv2","Visibility","T6","T9"],axis=1 , inplace=True)

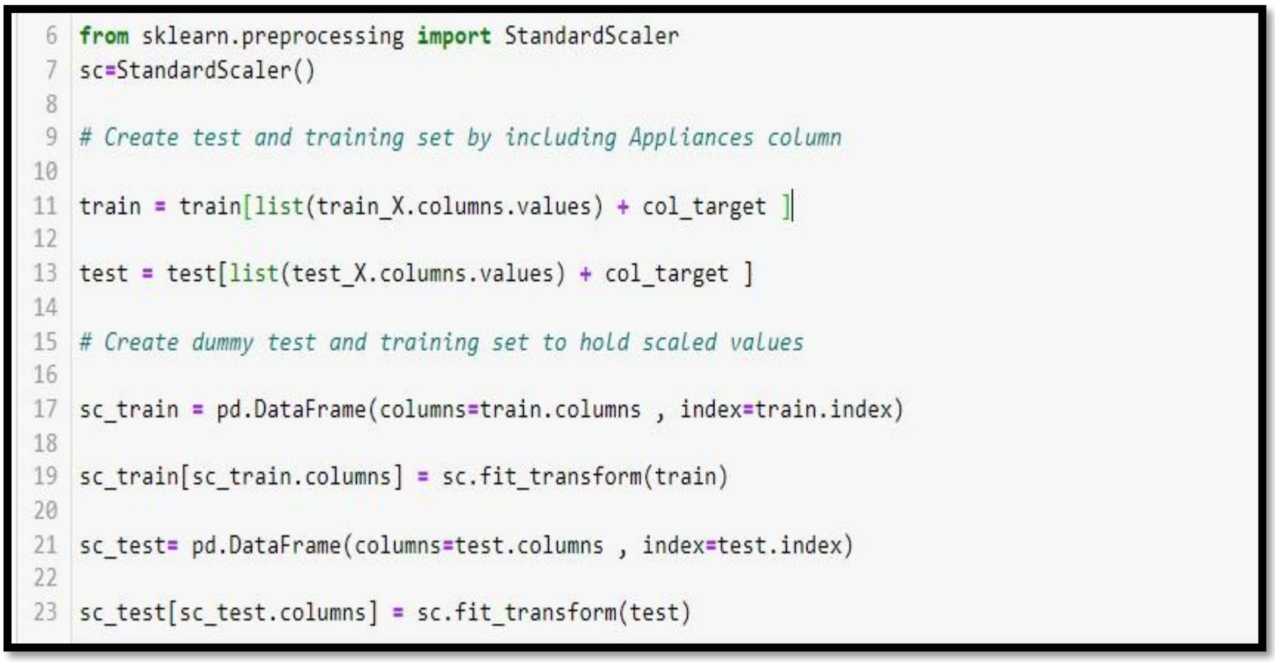
test_X.drop(["rv1","rv2","Visibility","T6","T9"], axis=1, inplace=True)

Feature Scaling for the concerned data

Scaling or Feature Scaling is the process of changing the scale of certain features to a common one. This is typically achieved through normalization and standardization (scaling techniques).[13]

- Normalization is the process of scaling data into a range of [0,1]. It's more useful and common for regression tasks
- Standardization is the process of scaling data so that they have a mean value of 0 and a standard deviation of 1. It's more useful and common for classification tasks

Variables that are measured at different scales do not contribute equally to the model fitting & model learned function and might end up creating a bias. Thus, to deal with this potential problem feature-wise standardized (μ=0, σ=1) is usually used prior to model fitting. To perform standardization, Scikit-Learn provides us with the StandardScaler class[14].



```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()

# Create test and training set by including Appliances column

train = train[list(train_X.columns.values) + col_target ]

test = test[list(test_X.columns.values) + col_target ]

# Create dummy test and training set to hold scaled values

sc_train = pd.DataFrame(columns=train.columns , index=train.index)

sc_train[sc_train.columns] = sc.fit_transform(train)

sc_test= pd.DataFrame(columns=test.columns , index=test.index)

sc_test[sc_test.columns] = sc.fit_transform(test)
```

[Fig – 21 Making of dummy variable to hold the scaled values]

Now we have created dummy test and training set to hold the scaled values. We will train our models on these scaled values to get a better accuracy.

As we already decided that this is a Regression problem. Regression analysis as we clearly know is a type of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable (s) (predictor). The regression methods which we will be using are

- LinearRegression
- SVR
- RandomForestRegressor
- XG Boost

Algorithms Used For The Case Study

Linear Regression

In linear regression we wish to fit a function in this

Form $Y = \beta 0+\beta 1X1+\beta 2X2+\beta 3X3$ where X is the vector of features and $\beta 0$, $\beta 1$,

$\beta 2$, $\beta 3$ are the coefficients we wish to learn. It updates β at every step by

reducing the loss function as much as possible. As modification to Linear regression model, we can apply Regularization techniques to penalize the coefficient values of the features, since higher values generally tend towards overfitting and loss of generalization.

Support Vector Machine

The Support Vector Regression (SVR) uses the same principles as the SVM for classification. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem.

Random Forests

A Random Forest is an ensemble technique capable of performing both regression tasks with the use of multiple decision trees and a technique called bagging. and works well on high dimensional data

XG Boost Regressor.

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks.

Metrics Used For The Case Study

1- mape (mean absolute percentage error) :-

Mean absolute percentage error is commonly used as a loss function for regression problems and in model evaluation, because of its very intuitive interpretation in terms of relative error.

MAPE = (1 / sample size) x Σ[(|actual - forecast|) / |actual|] x 100

Accuracy will hence be given by = 100 - MAPE

2- r2 Score :-

R2 score is used to evaluate the performance of a linear regression model.

It is the amount of the variation in the output dependent attribute which is predictable from the input independent variable(s). It is used to check how well-observed results are reproduced by the model,

depending on the ratio of total deviation of results described by the model.

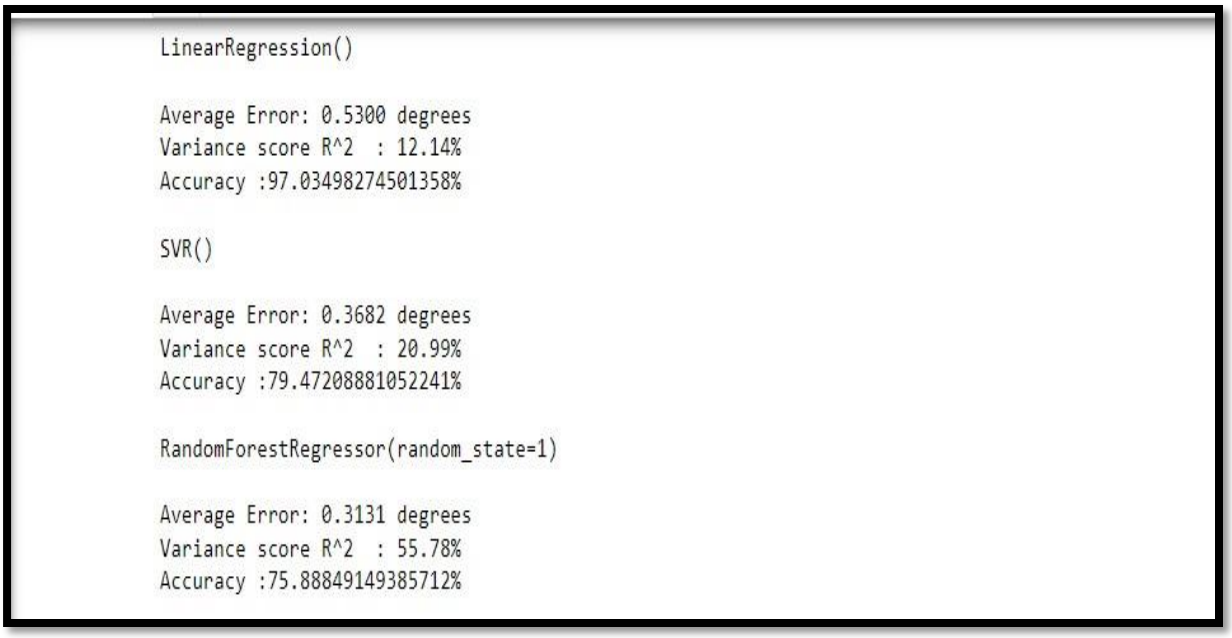
Mathematical formula - R2= 1- SSres / SStot

where,

SSres is the sum of squares of the residual errors.

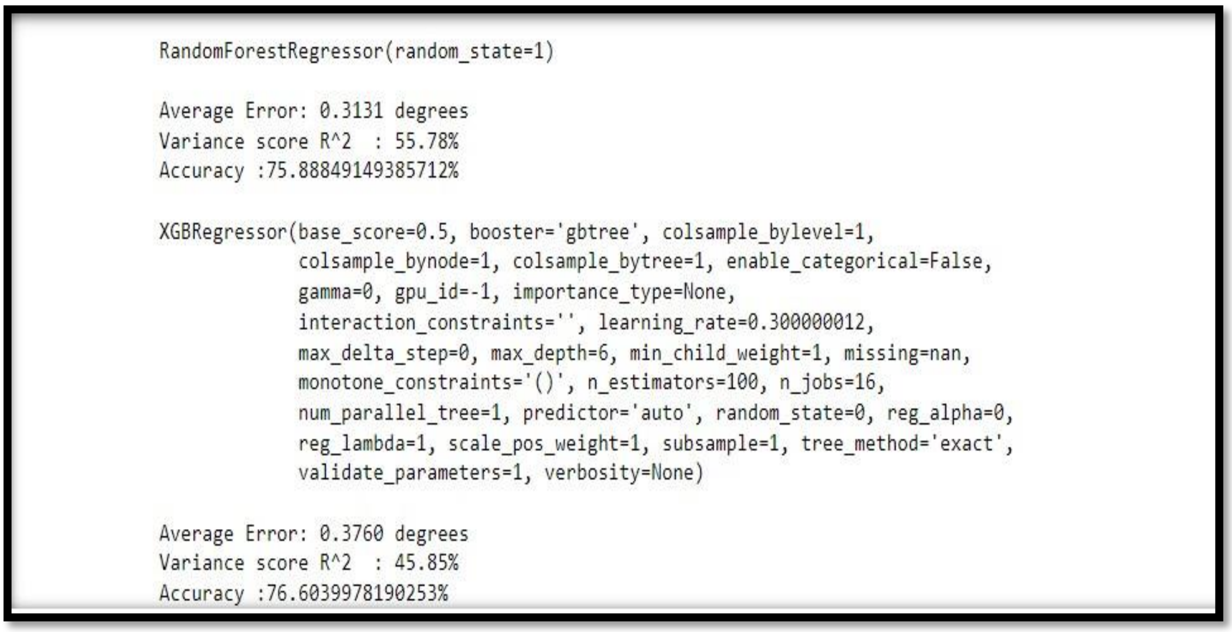
SStot is the total sum of the errors.

So after training the models and evaluating we are get the following result and accuracy as shown in the picture below



```
LinearRegression()

Average Error: 0.5300 degrees
Variance score R^2  : 12.14%
Accuracy :97.03498274501358%

SVR()

Average Error: 0.3682 degrees
Variance score R^2  : 20.99%
Accuracy :79.47208881052241%

RandomForestRegressor(random_state=1)

Average Error: 0.3131 degrees
Variance score R^2  : 55.78%
Accuracy :75.88849149385712%
```

[Fig – 22 Results of Linear Regressor, SVR and random forest regressor]



```
RandomForestRegressor(random_state=1)

Average Error: 0.3131 degrees
Variance score R^2  : 55.78%
Accuracy :75.88849149385712%

XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.300000012,
             max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=100, n_jobs=16,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)

Average Error: 0.3760 degrees
Variance score R^2  : 45.85%
Accuracy :76.6039978190253%
```

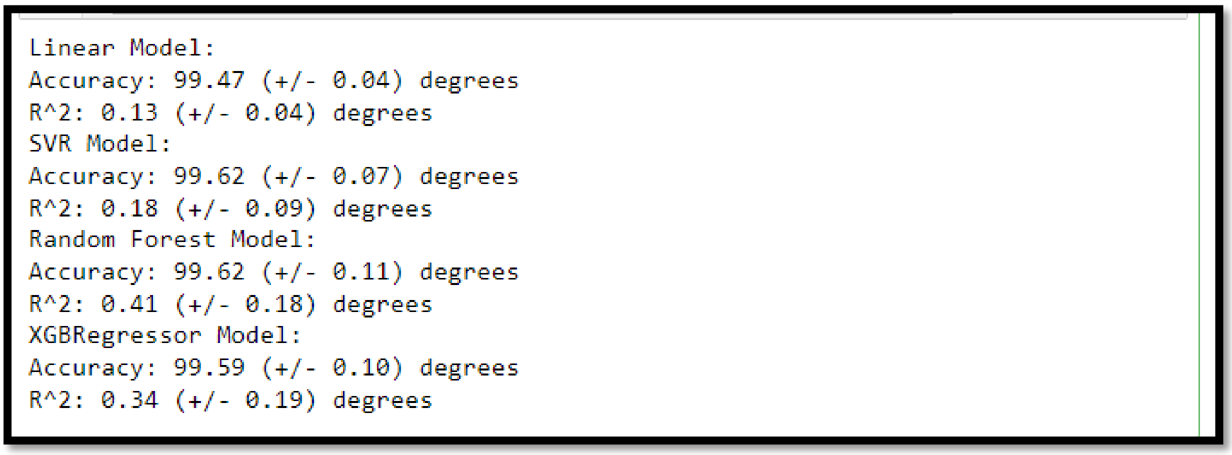
[Fig – 23 Results of XGB Regressor on the test data]

Now for fine tuning we will use cross Validation Technique. **Cross-Validation** also referred to **as out of sampling technique** is an essential element of a data science project. It is a resampling procedure used to evaluate machine learning models and access how the model will perform for an independent test dataset.[14]

We are using Time series cross validation for our data set. The order of the data is very important for time-series related problem. For the time-series dataset, the split of data into train and validation is according to the time also referred to as forward chaining method or rolling cross-validation. For a particular iteration, the next instance of train data can be treated as validation data.
So we have trained our models using TimeSeriesSplit as 10 for our data set as shown below

The method that can be used for cross-validating the time-series model is cross-validation on a rolling basis. Start with a small subset of data for training purpose, forecast for the later data points and then checking the accuracy for the forecasted data points. The same forecasted data points are then included as part of the next training dataset and subsequent data points are forecasted.

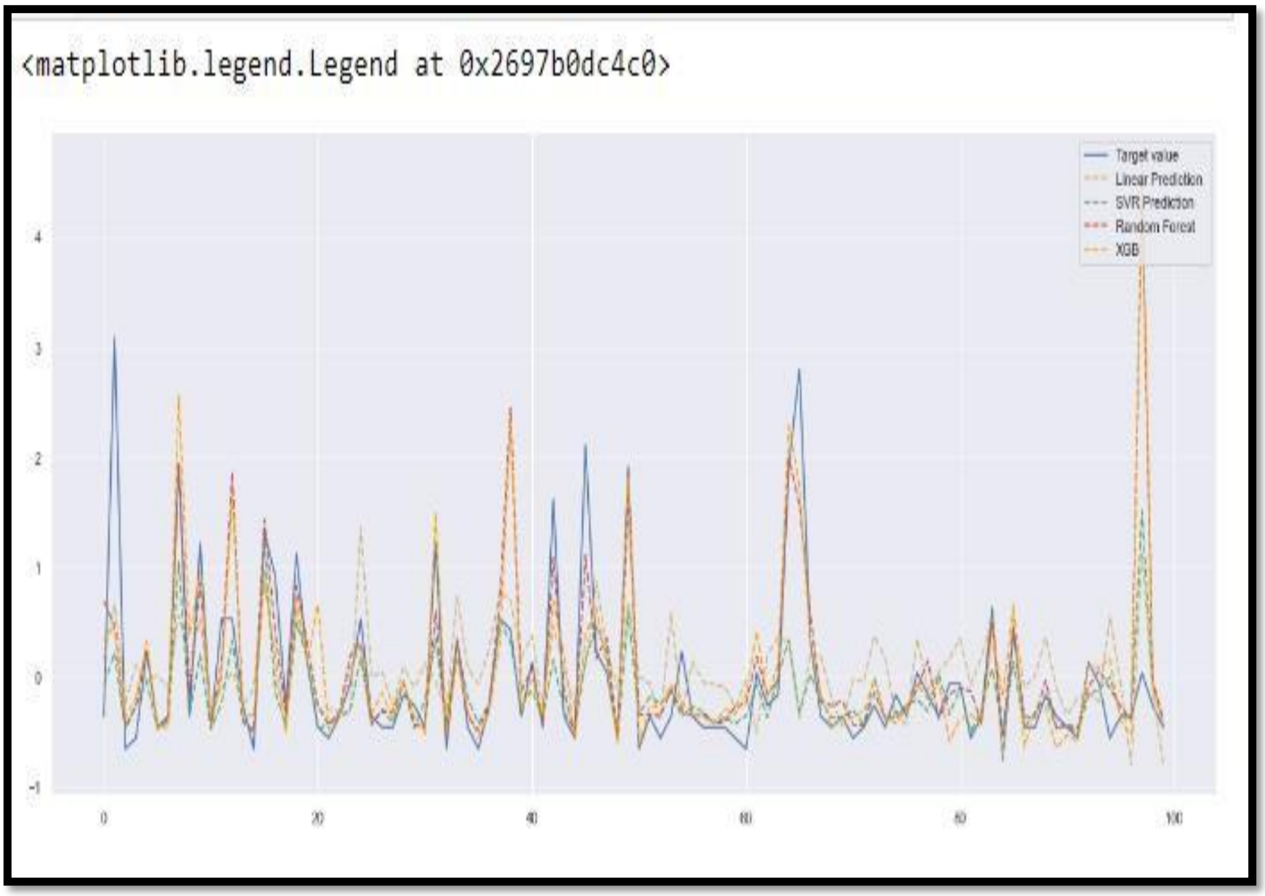
Now after the training using cross validation and we will check the Model performance on test data and as you can clearly see that accuracy is increased as a result of this cross validation. As we can clearly see in the results that because of cross validation accuracy of the models is increased.



```
Linear Model:
Accuracy: 99.47 (+/- 0.04) degrees
R^2: 0.13 (+/- 0.04) degrees
SVR Model:
Accuracy: 99.62 (+/- 0.07) degrees
R^2: 0.18 (+/- 0.09) degrees
Random Forest Model:
Accuracy: 99.62 (+/- 0.11) degrees
R^2: 0.41 (+/- 0.18) degrees
XGBRegressor Model:
Accuracy: 99.59 (+/- 0.10) degrees
R^2: 0.34 (+/- 0.19) degrees
```

[Fig – 24 Results of all the models after cross validation]

As we can clearly see that because of cross validation, we can see the result in form of increased accuracy. Now the final process is plotting the graph of all the algorithms to get the picture in whole.



```
<matplotlib.legend.Legend at 0x2697b0dc4c0>
```

[Fig – 25 Results of comparison graph for all the models]

Observations on the basis of above-

- Random Forest is best algorithm for this data set having the highest accuracy and high R2 score as compared to others. It captures the highs and lows of the data in a more efficient way.
- More improvement can be done to these models by doing some hyperparameter tuning.
- Log of appliances can be done because logging changes the skew distribution into normal distribution. So, linear regression model will give a much better accuracy with logging

Challenges & Learning gained during project

- It is very important to check the intercorrelation between all the variables in order to remove the redundant features with high correlation values.
- While scaling data, it is useful to maintain separate copies of dataframe which can be created using index and column names of original dataframe.
- The pipeline of adding algorithms should be easy to manage
- Seaborn and pyplot are good libraries to plot various properties of dataframe

<u>Conclusions</u>

The top 3 important features are humidity attributes, which leads to the conclusion that humidity affects power consumption more than temperature. Windspeed is least important as the speed of wind doesn’t affect power consumption inside the house. So, controlling humidity inside the house may lead to energy savings. In this way we can also understand how much energy is consumed by the house on daily basis so we can get average of energy consumption for the whole locality which will eventually help in energy distribution.

NOTE : - The python file in .py format, jupyter notebook (.ipynb format) and Html format is in the complete folder itself. Dataset is also attached there for reference

Link to the dataset used :- Appliances Energy Prediction | Kaggle

References

[1] https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning

[2] https://www.analyticssteps.com/blogs/introduction-machine-learning-supervised-and-unsupervised-learning

[3] https://www.ibm.com/cloud/learn/supervised-learning

[4] Machine Learning Random Forest Algorithm - Javatpoint

[5] K-Nearest Neighbors (KNN) Algorithm | by Afroz Chakure | DataDrivenInvestor

[6] Understanding Energy Consumption for Appliances | Analytics Vidhya (medium.com)

[7] What is Exploratory Data Analysis? | by Prasad Patil | Towards Data Science

[8] What is Exploratory Data Analysis? | IBM

[9] Pandas DataFrame.describe() - javatpoint

[10] What Is Data Visualization? Definition & Examples | Tableau

[11] Pearson Product-Moment Correlation - When you should run this test, the range of values the coefficient can take and how to measure strength of association. (laerd.com)

[12] 7. Correlation and Scatterplots — Basic Analytics in Python (sfu.ca)

[13] Feature Scaling Data with Scikit-Learn for Machine Learning in Python (stackabuse.com)

[14] Understanding 8 types of Cross-Validation | by Satyam Kumar | Towards Data Science