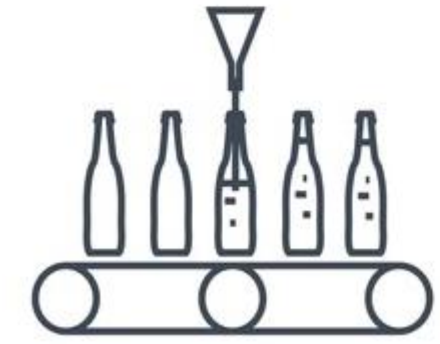
INNODATATICS
Innovation • Data • Analytics

Energy Management in Beverage Industry Using IoT sensor



CONTENTS

PROJECT OBJECTIVE

Project Background

Energy Management

Energy Management is the process of tracking and optimizing energy consumption to conserve usage in the Industry. Energy consumption by various units are monitored timely and the valuable insights helps in understanding the pattern and hence timely user intervention can save a lot of energy.

Overview of Beverage Industry

The beverage Industry also known as drink industry is the one which manufactures drinks. It produces alcoholic such as distilled spirits, wine and brewing as well as non-alcoholic drinks such as fruit juices, tea or coffee and other flavored drinks etc.

Benefits of Energy Management in Beverage Industry

- It increases overall productivity of the manufacturing plant.
- It helps to detect malfunctioning electrical devices timely.
- It is eco-friendly practice so makes the technology more constructive.

Aim:

- Main aim of this project is to manage the energy consumption by the overall plant
- To monitor the energy consumption on real time basis by individual plant units and overall plant using IoT sensor.
- To find whether at any instance energy consumption is low or high
- To find whether all the electric motors are working at optimum performance.
- To minimize power losses and overloading

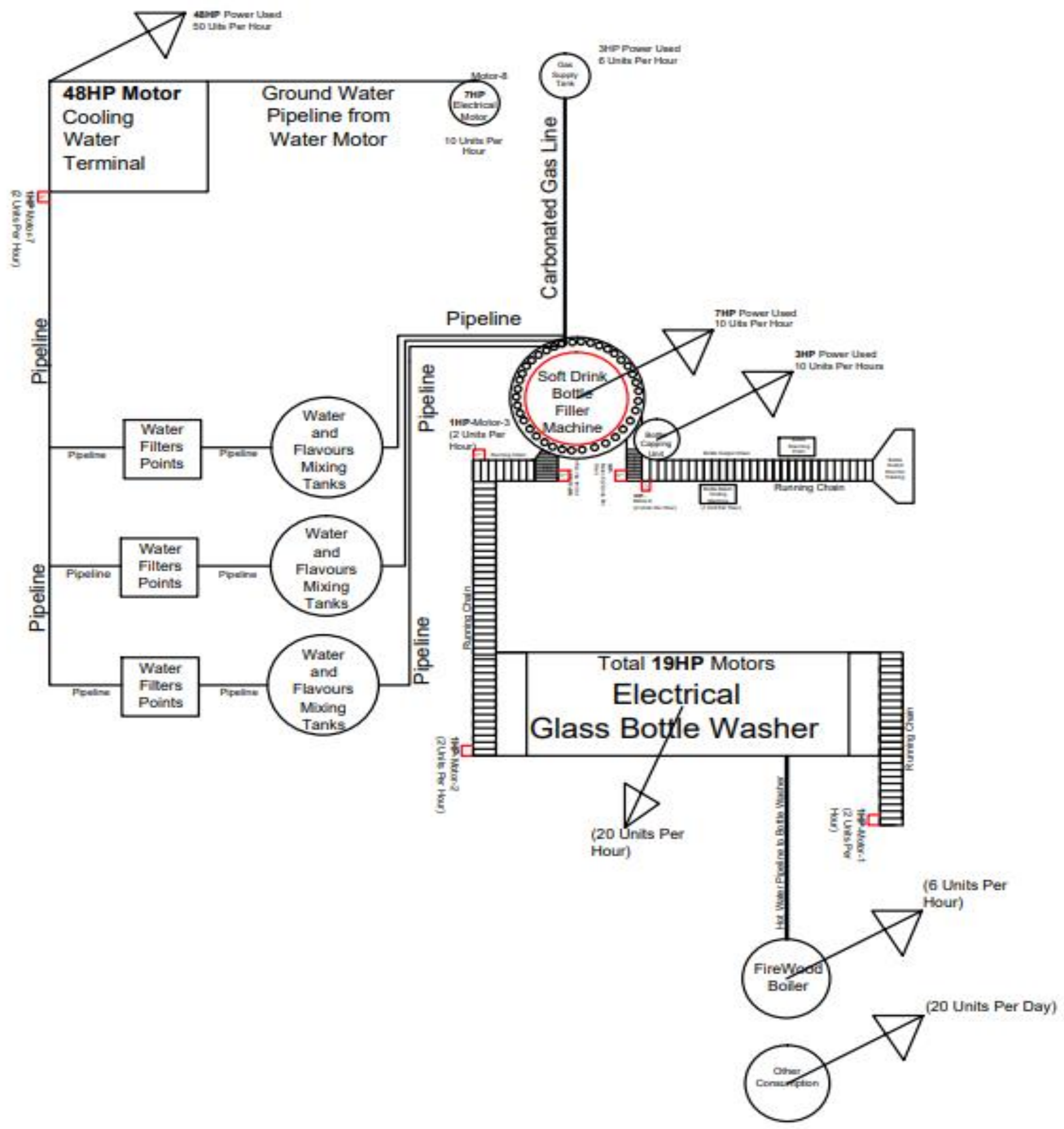
Business objective:

- Minimize: Energy losses, overloading
- Maximize: Energy efficiency

Business Constraint:

- To improve the performance of large amounts of data.
- Security of the data and working of the IoT sensor.

Flow of process in Beverage Plant:



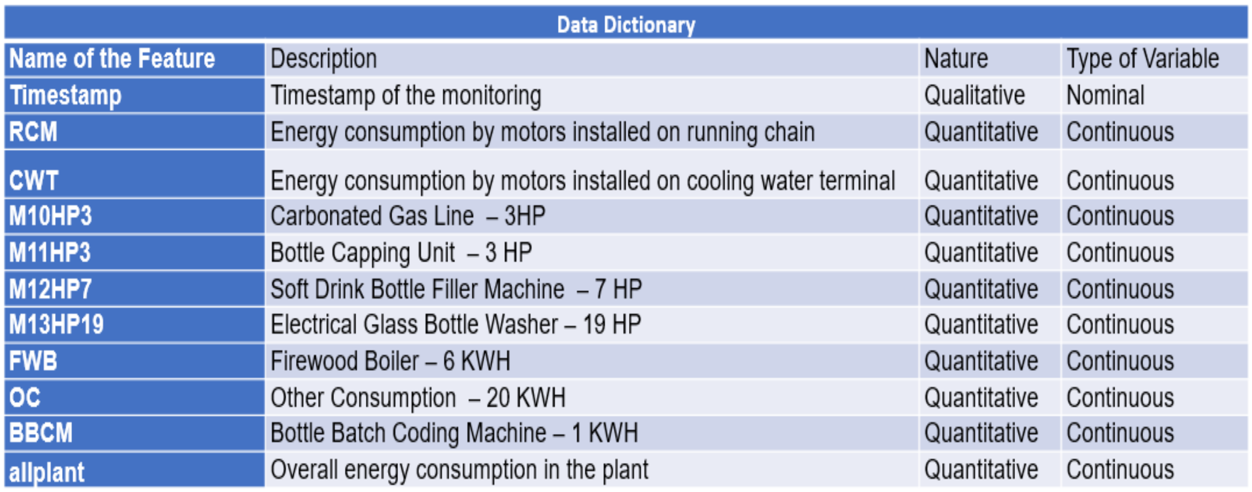
Project Dataset:

Data is collected on real time basis with the help of IoT sensor installed in the various electric motors in each of unit of the manufacturing plant. For every 10 seconds the data of energy consumption by all these individual units is obtained from the IoT sensor.

- Calculation = ((energy in kWh x 1000)/60 x 60) x 10

Electrical Energy consumed by individual motors installed in individual unit of the production system acts as independent variables and another meter is installed at the outlet of production system that measures overall electricity consumption acts as a target variable.

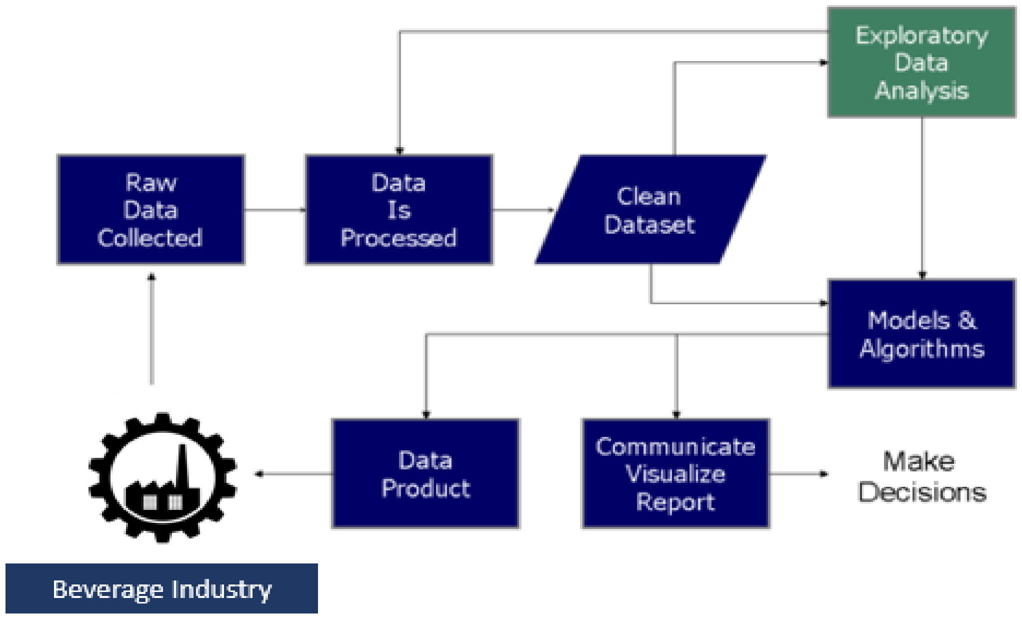
The features present in the data are:



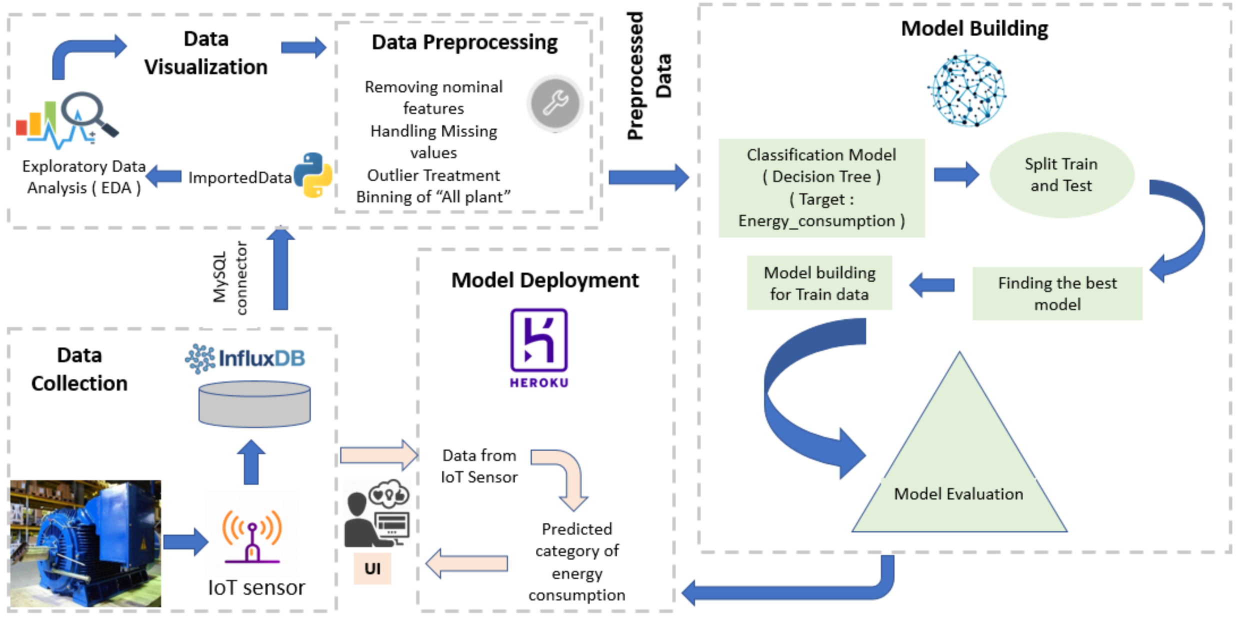
Data Dictionary			
Name of the Feature	Description	Nature	Type of Variable
Timestamp	Timestamp of the monitoring	Qualitative	Nominal
RCM	Energy consumption by motors installed on running chain	Quantitative	Continuous
CWT	Energy consumption by motors installed on cooling water terminal	Quantitative	Continuous
M10HP3	Carbonated Gas Line – 3HP	Quantitative	Continuous
M11HP3	Bottle Capping Unit – 3 HP	Quantitative	Continuous
M12HP7	Soft Drink Bottle Filler Machine – 7 HP	Quantitative	Continuous
M13HP19	Electrical Glass Bottle Washer – 19 HP	Quantitative	Continuous
FWB	Firewood Boiler – 6 KWH	Quantitative	Continuous
OC	Other Consumption – 20 KWH	Quantitative	Continuous
BBCM	Bottle Batch Coding Machine – 1 KWH	Quantitative	Continuous
allplant	Overall energy consumption in the plant	Quantitative	Continuous

INITIATION AND DATA PREPARATION

High level Flowchart of the process:



Detailed Project Architecture:



Data Preparation:

This preprocessing step involves exploratory data analysis, data visualization, checking null and duplicate values, outlier treatment , removing undesirable nominal features such as “Timestamp” and usage of binning to convert output variable “allplant” into categorical variable so that data can be made to be used for classification model.

Exploratory Data Analysis:

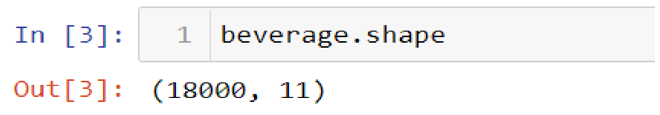
This step is used to identify and summarize the main characteristics in a data set through data analysis and investigation. We used data visualizations to better understand how best to manipulate data sources, to determine which statistical techniques are most appropriate for data analysis, and for choosing the right features for a model.

EDA includes fallowing steps:

- Details About Data
- Data cleaning
- Statistical Insights
- Univariate analysis
- Bi-variate/Multivariate analysis

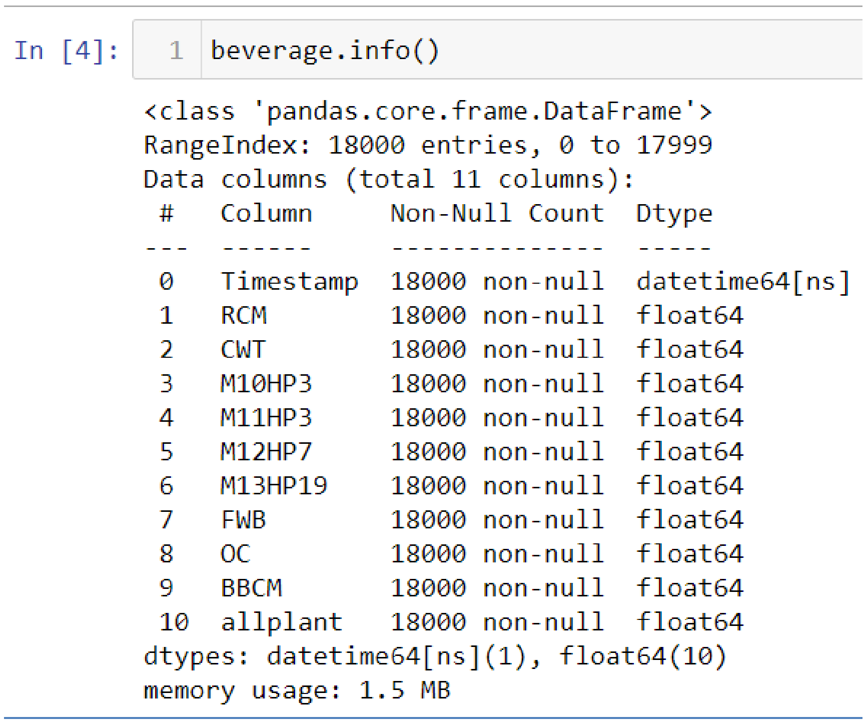
Details about Data:

The first and foremost step of any data analysis, after loading the data file, should be about checking few introductory details like, no. Of columns, no. of rows, types of features (categorical or Numerical), data types of column entries.



```
In [3]:  beverage.shape
Out[3]:  (18000, 11)
```

- Data has 18000 rows and 11 columns



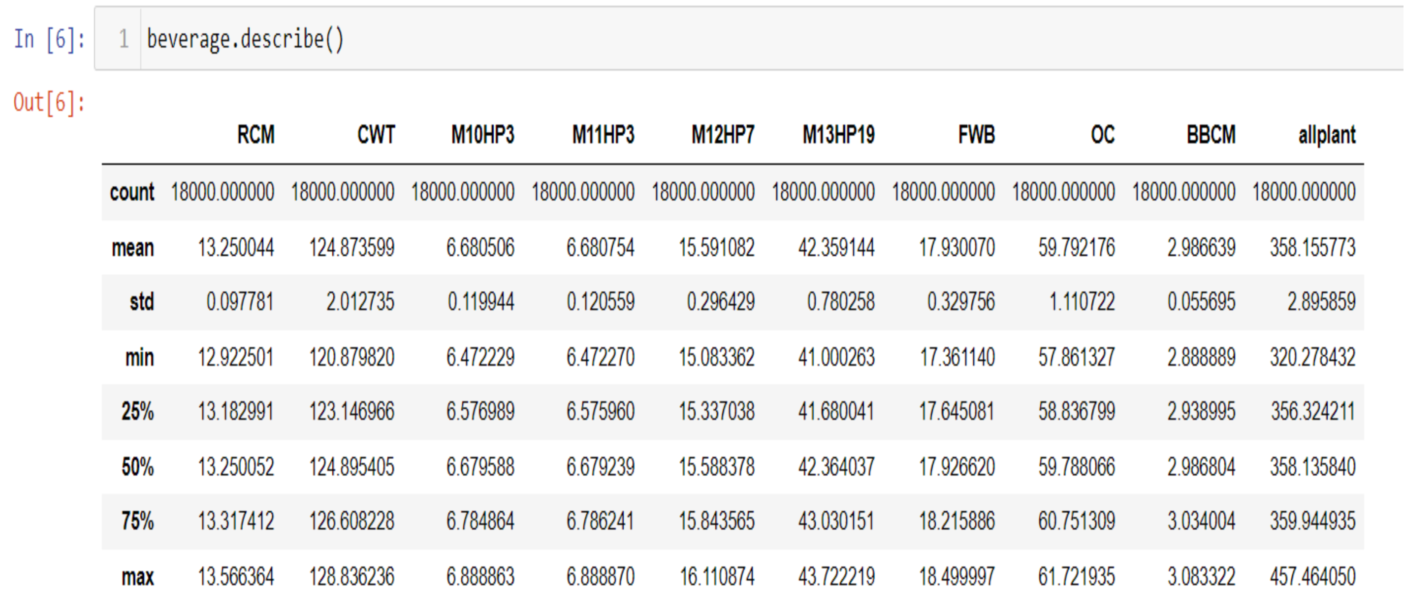
```
In [4]:  beverage.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18000 entries, 0 to 17999
Data columns (total 11 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Timestamp  18000 non-null  datetime64[ns]
 1   RCM        18000 non-null  float64
 2   CWT        18000 non-null  float64
 3   M10HP3     18000 non-null  float64
 4   M11HP3     18000 non-null  float64
 5   M12HP7     18000 non-null  float64
 6   M13HP19    18000 non-null  float64
 7   FWB        18000 non-null  float64
 8   OC         18000 non-null  float64
 9   BBCM       18000 non-null  float64
 10  allplant   18000 non-null  float64
dtypes: datetime64[ns](1), float64(10)
memory usage: 1.5 MB
```

Data Cleansing:

- **Nominal Features:** The "Timestamp" is just nominal data which is only showing the timestamp of the monitoring and as no relevance with other features, so it is removed from the dataset.
- **Missing Values:** There are no missing values and null values in the dataset.
- **Outliers:** Here in this Dataset, There are outliers in the features like "RCM" and "allplant" so the outliers were replaced with the max threshold of the feature.
- **Binning:** The target variable "allplant" is continuous data and to make the data compatible to classification problem it is then converted to categorical by the concept of binning. "allplant" is converted into two categories namely high and low. From 250 to 358 watts per 10 seconds is treated as one category i.e. low consumption and from 358 watts per 10 seconds onwards it is treated as another category i.e. high consumption.

Statistical Insights:

Below is the count, mean, standard deviation, minimum and maximum values and the quantiles of the data.



```
In [6]:  beverage.describe()
```

Out[6]:

	RCM	CWT	M10HP3	M11HP3	M12HP7	M13HP19	FWB	OC	BBCM	allplant
count	18000.000000	18000.000000	18000.000000	18000.000000	18000.000000	18000.000000	18000.000000	18000.000000	18000.000000	18000.000000
mean	13.250044	124.873599	6.680506	6.680754	15.591082	42.359144	17.930070	59.792176	2.986639	358.155773
std	0.097781	2.012735	0.119944	0.120559	0.296429	0.780258	0.329756	1.110722	0.055695	2.895859
min	12.922501	120.879820	6.472229	6.472270	15.083362	41.000263	17.361140	57.861327	2.888889	320.278432
25%	13.182991	123.146966	6.576989	6.575960	15.337038	41.680041	17.645081	58.836799	2.938995	356.324211
50%	13.250052	124.895405	6.679588	6.679239	15.588378	42.364037	17.926620	59.788066	2.986804	358.135840
75%	13.317412	126.608228	6.784864	6.786241	15.843565	43.030151	18.215886	60.751309	3.034004	359.944935
max	13.566364	128.836236	6.888863	6.888870	16.110874	43.722219	18.499997	61.721935	3.083322	457.464050

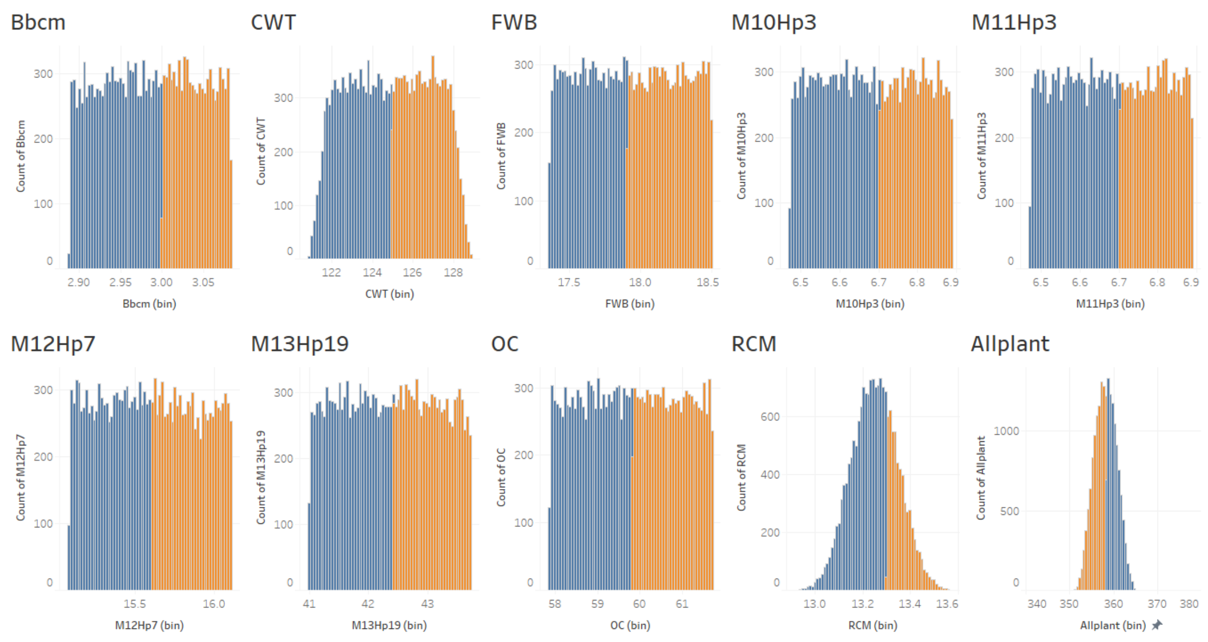
- There is not much difference between 75th %tile and max values.
- Thus observations suggests that there is no Outliers in our data set.

Data Visualization

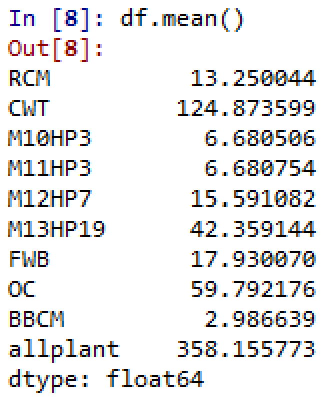
Univariate Analysis:

This shows every observation/distribution in data on a single data variable. It is shown with the help of Histogram.

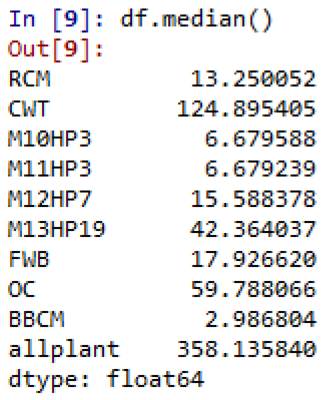
Histogram: A histogram is a value distribution plot of numerical columns. It basically creates bins in various ranges in values and plots it where we can visualize how values are distributed. We can have a look where more values lie like in positive, negative, or at the centre (mean).



Skewness: Skewness refers to a distortion or asymmetry that deviates from the symmetrical bell curve, or normal distribution, in a set of data. If the curve is shifted to the left or to the right, it is said to be skewed. Skewness can be quantified as a representation of the extent to which a given distribution varies from a normal distribution. A normal distribution has a skewness of zero.

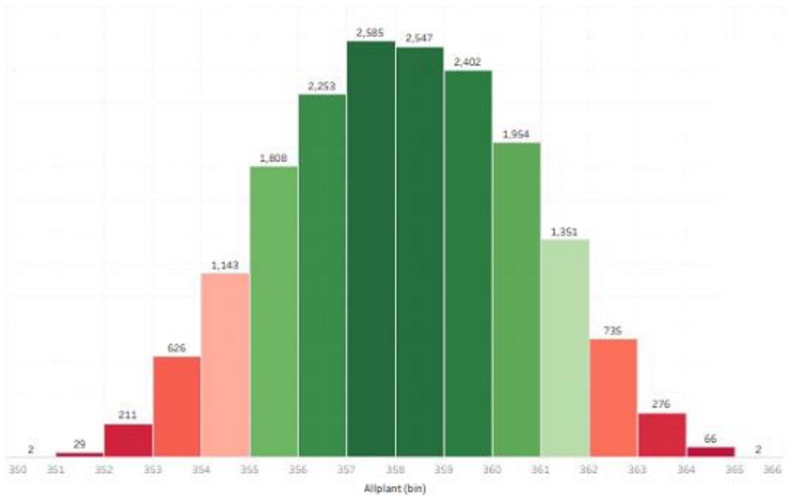


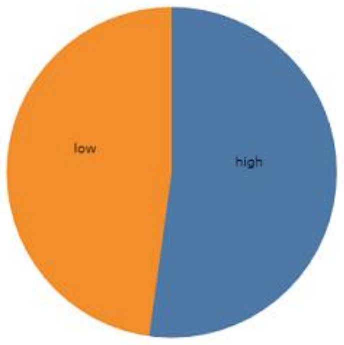
```
In [8]: df.mean()
Out[8]:
RCM           13.250044
CWT          124.873599
M10HP3         6.680506
M11HP3         6.680754
M12HP7        15.591082
M13HP19       42.359144
FWB           17.930070
OC            59.792176
BBCM           2.986639
allplant     358.155773
dtype: float64
```



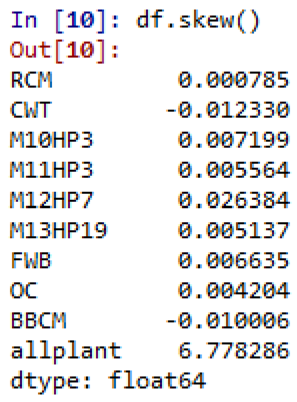
```
In [9]: df.median()
Out[9]:
RCM           13.250052
CWT          124.895405
M10HP3         6.679588
M11HP3         6.679239
M12HP7        15.588378
M13HP19       42.364037
FWB           17.926620
OC            59.788066
BBCM           2.986804
allplant     358.135840
dtype: float64
```

Mean and median values each variable are close in range. So histogram for output and every input variable having symmetric nature with less skewness values, thereby distribution of data on both half are almost equal.

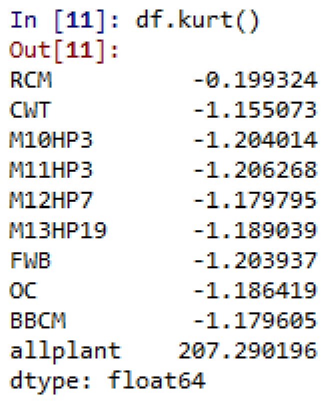




In the above figure illustrate the “Allplant” output variable data distribution in data. The histogram plot represents “Allplant” output variable continuous data distribution, and pie chart represents that of its categorical class distribution in “low” & “high” classes according to the threshold value of electrical power consumption. The continuous power consumption value of “Allplant” data that’s comes above the threshold limit treated as “high” class category and that of values comes below threshold limit treated as “low” class category.



```
In [10]: df.skew()
Out[10]:
RCM          0.000785
CWT         -0.012330
M10HP3       0.007199
M11HP3       0.005564
M12HP7       0.026384
M13HP19      0.005137
FWB          0.006635
OC           0.004204
BBCM        -0.010006
allplant     6.778286
dtype: float64
```



```
In [11]: df.kurt()
Out[11]:
RCM          -0.199324
CWT          -1.155073
M10HP3       -1.204014
M11HP3       -1.206268
M12HP7       -1.179795
M13HP19      -1.189039
FWB          -1.203937
OC           -1.186419
BBCM         -1.179605
allplant    207.290196
dtype: float64
```

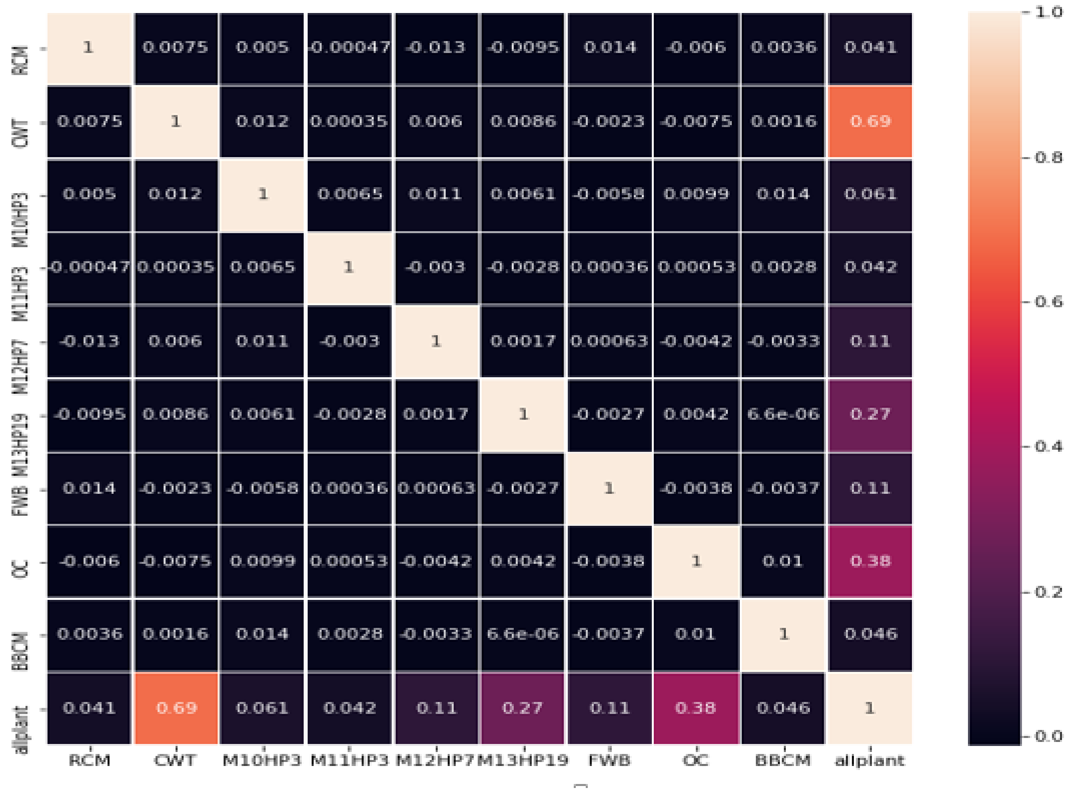
Kurtosis (ku): Kurtosis is a measure of relative peakedness of a distribution. It is a shape parameter that characterizes the degree of peakedness. A distribution is said to be leptokurtic when the degree of peakedness is greater than zero, it is mesokurtic when the degree of peakedness is equal to zero, and it is platykurtic when the degree of peakedness is less than zero.

As every input variables shows shorter or truncated tail ends with more like uniform distribution, they having platykurtic distribution with Ku value less than zero. Since the output variable "Allplant" having higher peakedness they showing leptokurtic distribution with Ku value greater than zero.

Bi-Variate analysis:

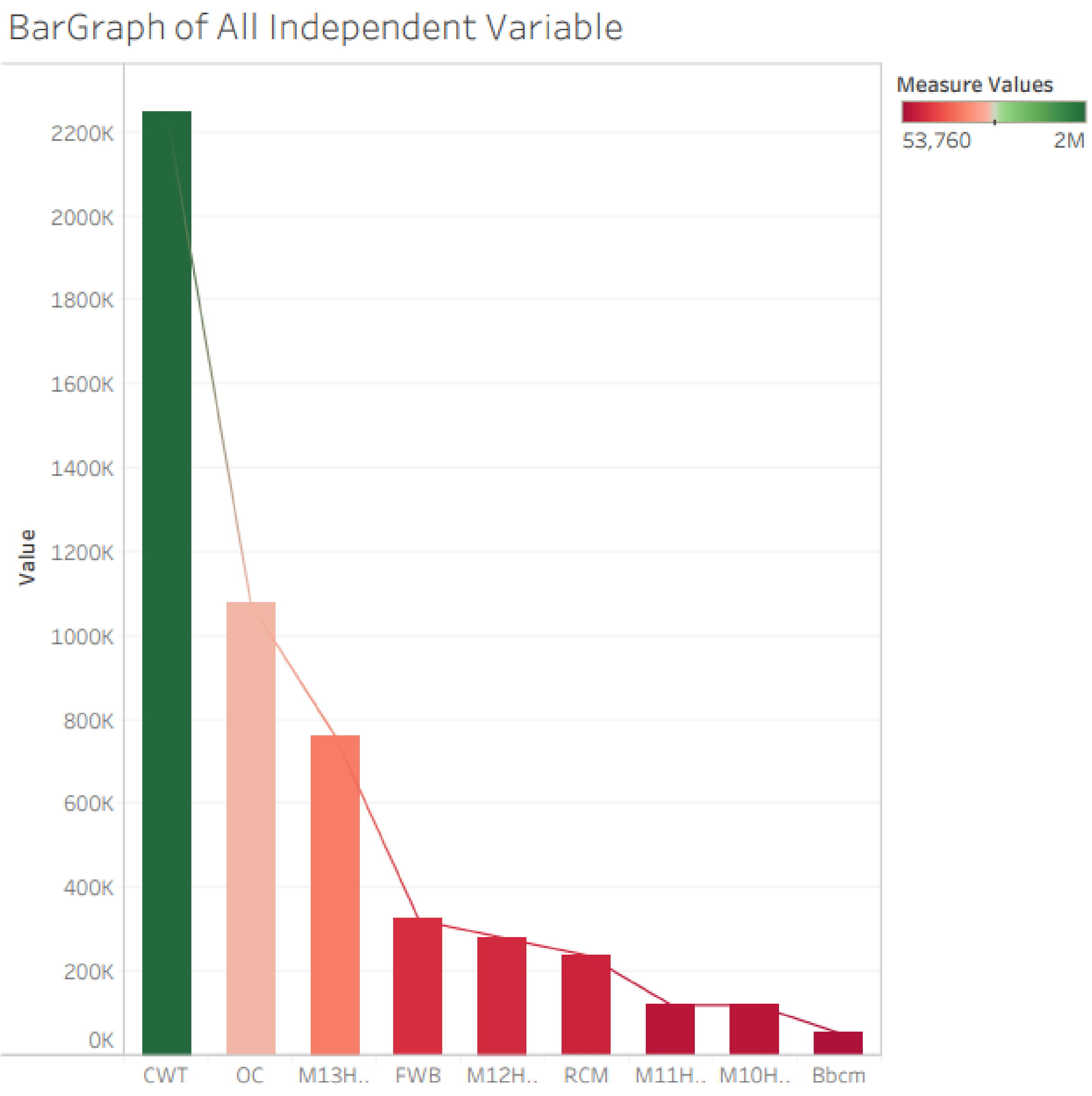
Bivariate analysis displays are done to reveal the relationship between two data variables. It shown with the help of Heat map

Heat Map: A heat map is a two-dimensional representation of data in which values are represented by colors. Here heat map provides correlation between various features.



- Lighter shades represents positive correlation while darker shades represents negative correlation.
- Here we can infer that “allplant” has strong positive correlation with “CWT” and a moderate relation with M13HP19, OC.

Power Consumption by different units of plant:



CWT, OC, M13Hp19, FWB, M12Hp7, RCM, M11Hp3, M10Hp3, Bbcm, CWT, OC, M13Hp19, FWB, M12Hp7, RCM, M11Hp3, M10Hp3 and Bbcm. Color shows CWT, OC, M13Hp19, FWB, M12Hp7, RCM, M11Hp3, M10Hp3 and Bbcm.

Fig. shows the total power consumption of each motor’s over a period of time. The motors with higher power ratings will consumes more power than that of motors with less ratings. So the bar plots illustrates that,

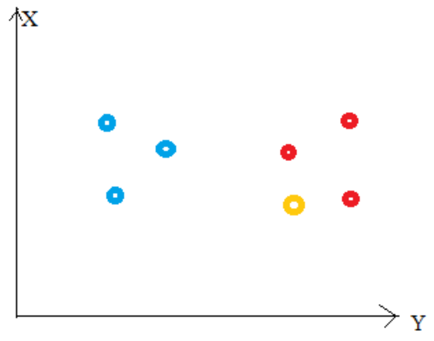
- The “CWT” motor has highest power rating, since it consumes more power compares to all others.
- The “OC” and “M13HP19” motors also have higher power ratings, since they shows significant power consumption.
- The “Bbcm” motor has the lowest power rating as its consumption is very low compares to all others.

MODEL BUILDING

KNN Algorithm:

KNN stands for K- Nearest Neighbor algorithm, is a non-parametric supervised machine learning problem which can be used for both classification as well as regression. It assumes that similar things are in close proximity. In other similar things are near to each other. K is the number of nearest neighbors that the algorithm considers for classification. In KNN no weights are learned instead the entire training dataset is stored in the memory. Therefore, model representation for the KNN model is the entire training dataset.

KNN model works by segregating the data points in different classes based upon the similarity identified by the distance between the data points by various distance metrics.



Here, nearest neighbors are those data points that have minimum distance in feature space from our new data point. And K is the number of such data points we consider in our implementation of the algorithm. Therefore, distance metric and K value are two important considerations while using the KNN algorithm.

Cons:

- Accuracy depends upon the quality of data
- Slow performance when data is large
- Sensitive to the scale of the data and irrelevant features
- Requires high memory as all the training data needs to be stored
- It is computationally expensive

In our model different values of K have been tried to get the best possible accuracy and various distance metrics such as Euclidean, Manhattan etc. have also been experimented upon.

Decision Tree Classification:

Decision Tree classification is non parametric supervised machine learning algorithm which makes a flowchart by learning simple decision rules from the data features and with the help of this tree like flow chart it predicts the class of the target variable. Each internal node represents a test on a feature, each leaf node represents class label i.e. decision taken after computing all the features and branches represent the conjunctions of features that leads to those class label. The path from root to leaf represents classification rules.

In this algorithm, Training data is used to build model. Tree generator determines which variable to split at a node and what will be the value of the split, decision of split and stop is then taken based on the condition and finally at the end terminal nodes are assigned to the label.

Decision tree algorithm is also called as greedy algorithm is used. Below are the steps of the algorithm:

- Tree is constructed in a top down recursive divide and conquer manner
- At start, all training data are at the root
- Attributes are categorical if not then they are discretized well in advance
- Input data is partitioned recursively based on selected attributes
- Test attributes at each node are selected on the basis of statistical measure such as information gain.

Parameters used in Decision Tree:

This algorithm uses two parameters such as max_depth and entropy as a criterion.

max_depth: It is one of the hyperparameter used in the decision tree classification algorithm to tune the model for optimum performance i.e. to avoid overfitting. It takes an integral value and based upon that it decides till what depth the decision tree has to be made. In our model we kept max_depth=9 which is giving optimum accuracy results.

criterion: It is another hyperparameter that is used in the decision three which is instrumental in deciding how the decision tree will grow. This parameter determines how the impurity of a split will be measured. Either "gini" or"entropy" can be used. In our model we used criterion=entropy. Decision trees uses information gain to split the node and gini or entropy criterion measures the impurity of the node. Greater the impurity, greater will be the information gain. The range of "entropy" is 0-1. Entropy and information gain can be calculated as below:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Where P_i is the probability that an arbitrary tuple belongs to class C_i

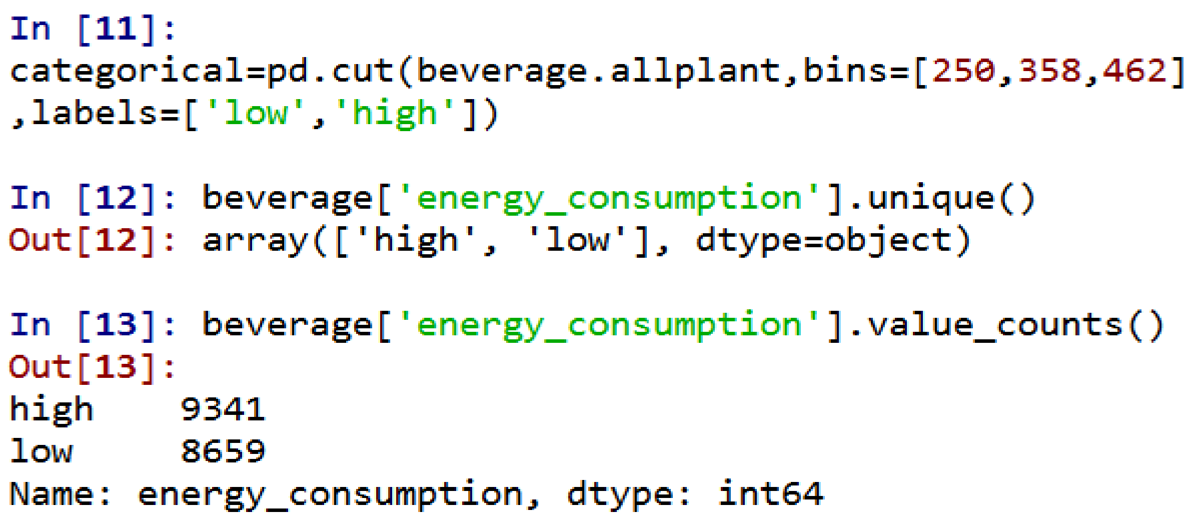
Information gain is calculated as below:

$$IG(Y, X) = E(Y) - E(Y|X)$$

Above is information gain from X on Y.

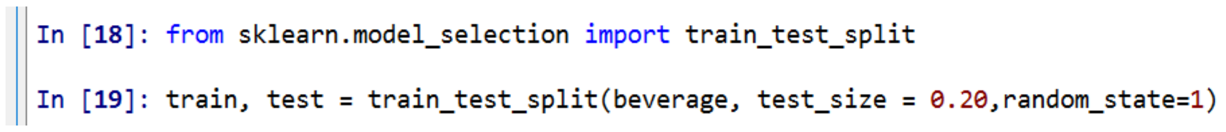
Steps Involved:

- Import the dataset
- Data preparation which includes data cleansing and binning of output variable "allplant" to convert it into categorical feature



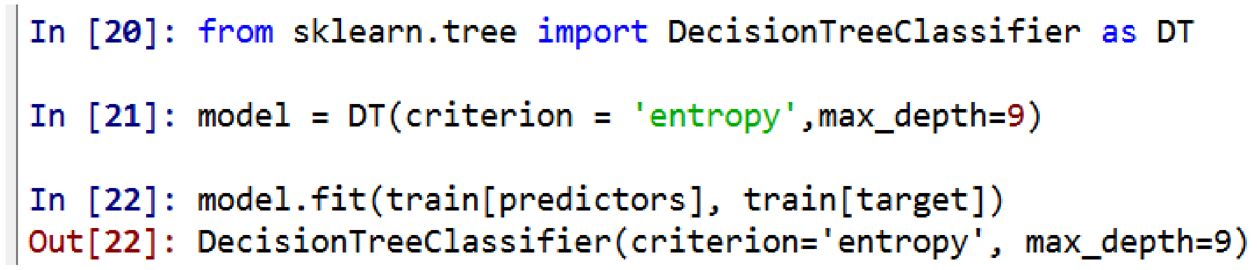
```
In [11]:
categorical=pd.cut(beverage.allplant,bins=[250,358,462]
,labels=['low','high'])

In [12]: beverage['energy_consumption'].unique()
Out[12]: array(['high', 'low'], dtype=object)

In [13]: beverage['energy_consumption'].value_counts()
Out[13]:
high    9341
low     8659
Name: energy_consumption, dtype: int64
```

- Setting predictors and target
- Splitting the data into test and train. Train = 80% and Test = 20%.



```
In [18]: from sklearn.model_selection import train_test_split

In [19]: train, test = train_test_split(beverage, test_size = 0.20,random_state=1)
```

- Setting of parameters for the model and fit the data

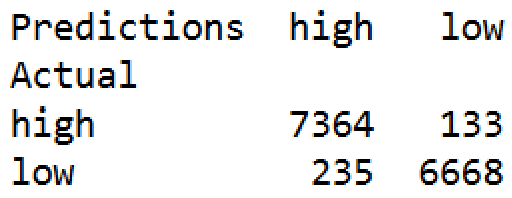


```
In [20]: from sklearn.tree import DecisionTreeClassifier as DT

In [21]: model = DT(criterion = 'entropy',max_depth=9)

In [22]: model.fit(train[predictors], train[target])
Out[22]: DecisionTreeClassifier(criterion='entropy', max_depth=9)
```

max_depth = 9 and criterion is set as "entropy "

- Summarization of model performance using confusion matrix for train and test data.

  For Train:



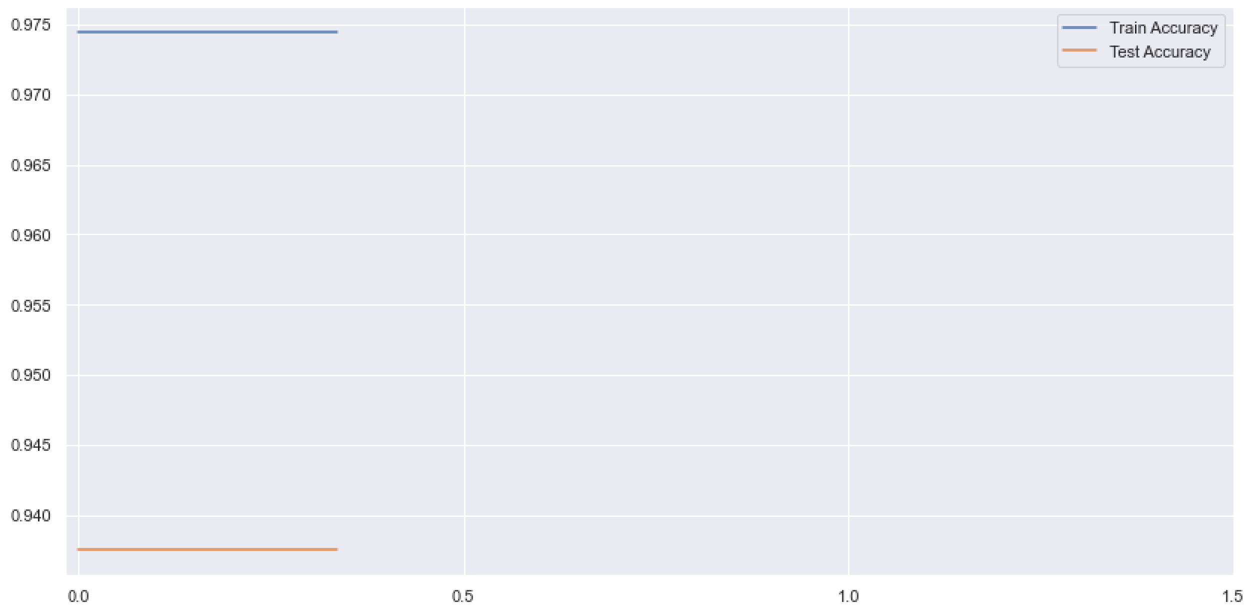
```
Predictions  high   low
Actual
high         7364   133
low           235  6668
```

For Test:

```
Predictions  high   low
Actual
high         1736   108
low           123  1633
```

- Model Evaluation by calculating train and test accuracy.

```
In [31]: np.mean(Y_test_pred == test[target]) # Test Data
Accuracy
Out[31]: 0.9358333333333333

In [32]: np.mean(Y_train_pred == train[target]) # Train Data
Accuracy
Out[32]: 0.9744444444444444
```



Above graph shows the model performance after applying post pruning which shows the model has attained the optimum performance So, based upon the performance of both the models it was decided to finalize decision tree classification algorithm as for a large dataset in the project not only it is giving faster performance but also giving optimal prediction accuracy.

DEPLOYMENT

In the Deployment Actually we have used the following tools:

- Python.
- Python Flask
- Html, CSS.
- Heroku
- GitHub

Python:

A tool used for Manipulation of data and model deployment using different libraries and Modules like Pandas, Flask, statsmodels.

Python Flask:

Python Flask is an open-source python library that makes it easy to create custom web apps for machine learning and data science. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third party libraries provide common functions.

```
from flask import Flask
```

Html, CSS:

HTML (Hypertext markup language) is use for making structure of the website to take inputs manually from the user.

CSS (Cascading Style sheet) is used for describing the presentation of the html document or styling the html structure to make it look good

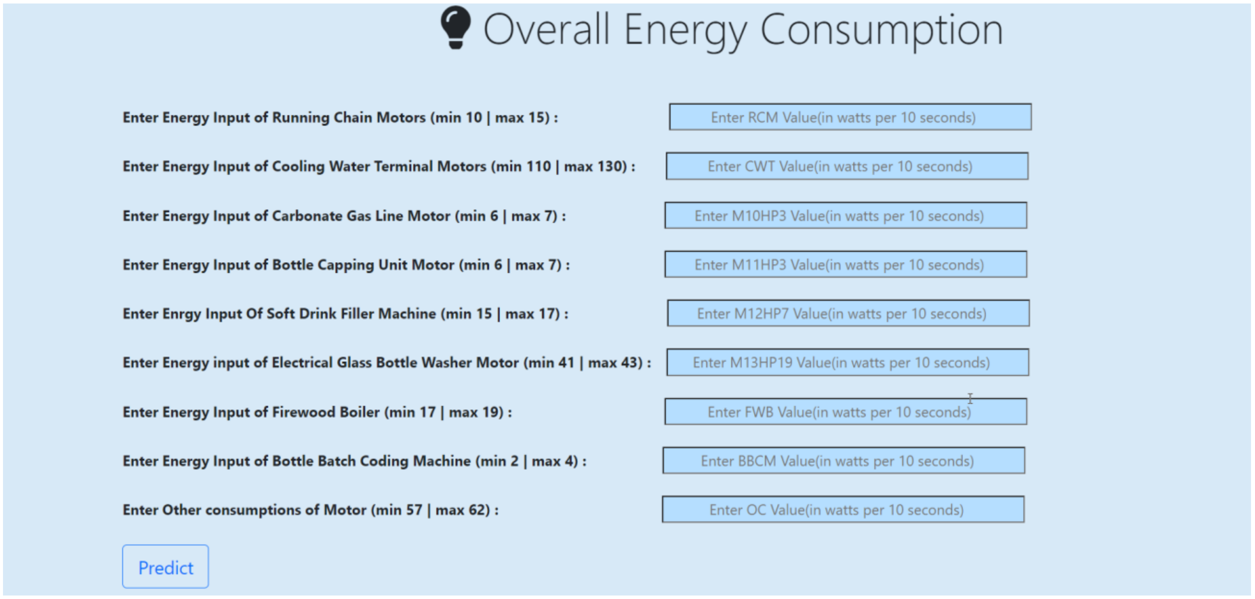
Heroku.

Heroku is platform as a service (PaaS) that enables developers to build, run and operate application entirely in the cloud

GitHub.

Heroku integrates with GitHub to make it easy to deploy code living on GitHub to apps running on Heroku. When GitHub integration is configured for the Heroku app, Heroku can automatically build and release (is the build is successful) pushes to specified GitHub Repository

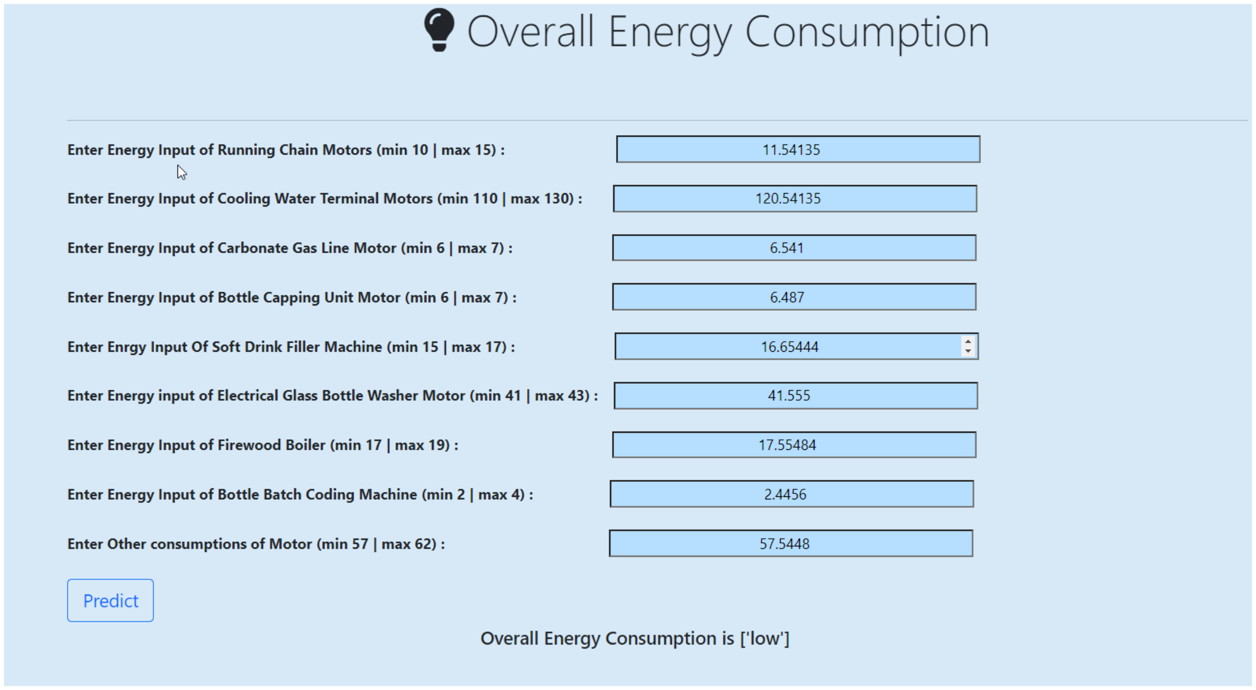
Output



Overall Energy Consumption

Enter Energy Input of Running Chain Motors (min 10 | max 15) :	Enter RCM Value(in watts per 10 seconds)
Enter Energy Input of Cooling Water Terminal Motors (min 110 | max 130) :	Enter CWT Value(in watts per 10 seconds)
Enter Energy Input of Carbonate Gas Line Motor (min 6 | max 7) :	Enter M10HP3 Value(in watts per 10 seconds)
Enter Energy Input of Bottle Capping Unit Motor (min 6 | max 7) :	Enter M11HP3 Value(in watts per 10 seconds)
Enter Enrgy Input Of Soft Drink Filler Machine (min 15 | max 17) :	Enter M12HP7 Value(in watts per 10 seconds)
Enter Energy input of Electrical Glass Bottle Washer Motor (min 41 | max 43) :	Enter M13HP19 Value(in watts per 10 seconds)
Enter Energy Input of Firewood Boiler (min 17 | max 19) :	Enter FWB Value(in watts per 10 seconds)
Enter Energy Input of Bottle Batch Coding Machine (min 2 | max 4) :	Enter BBCM Value(in watts per 10 seconds)
Enter Other consumptions of Motor (min 57 | max 62) :	Enter OC Value(in watts per 10 seconds)

We have to enter Energy Consumption of motors in watts per 10 seconds that are collected from the IoT used in the industry and after Clicking Predict Button we get output if the consumption is Low or High



Overall Energy Consumption

Enter Energy Input of Running Chain Motors (min 10 | max 15) : 11.54135

Enter Energy Input of Cooling Water Terminal Motors (min 110 | max 130) : 120.54135

Enter Energy Input of Carbonate Gas Line Motor (min 6 | max 7) : 6.541

Enter Energy Input of Bottle Capping Unit Motor (min 6 | max 7) : 6.487

Enter Enrgy Input Of Soft Drink Filler Machine (min 15 | max 17) : 16.65444

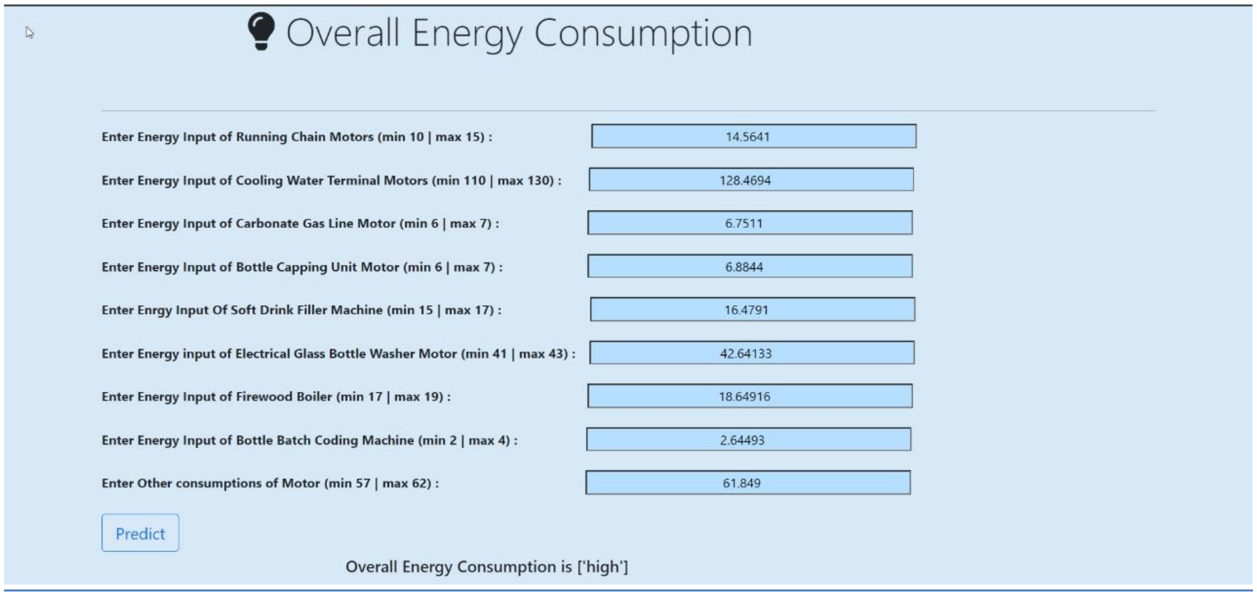
Enter Energy input of Electrical Glass Bottle Washer Motor (min 41 | max 43) : 41.555

Enter Energy Input of Firewood Boiler (min 17 | max 19) : 17.55484

Enter Energy Input of Bottle Batch Coding Machine (min 2 | max 4) : 2.4456

Enter Other consumptions of Motor (min 57 | max 62) : 57.5448

Overall Energy Consumption is ['low']



Overall Energy Consumption

Enter Energy Input of Running Chain Motors (min 10 | max 15) : 14.5641

Enter Energy Input of Cooling Water Terminal Motors (min 110 | max 130) : 128.4694

Enter Energy Input of Carbonate Gas Line Motor (min 6 | max 7) : 6.7511

Enter Energy Input of Bottle Capping Unit Motor (min 6 | max 7) : 6.8844

Enter Enrgy Input Of Soft Drink Filler Machine (min 15 | max 17) : 16.4791

Enter Energy input of Electrical Glass Bottle Washer Motor (min 41 | max 43) : 42.64133

Enter Energy Input of Firewood Boiler (min 17 | max 19) : 18.64916

Enter Energy Input of Bottle Batch Coding Machine (min 2 | max 4) : 2.64493

Enter Other consumptions of Motor (min 57 | max 62) : 61.849

Overall Energy Consumption is ['high']

Links

Link to the Web app: https://energy-consumption-iot.herokuapp.com/

Link to the GitHub repository: https://sgithub.com/Aniket240811/Energy-Management-Using-Iot-Sensors

CONCLUSION

- The model will helps to predict all plant electrical power consumption meter reading as "high" & "low" in accordance with a threshold value for power consumption under constant production rate.

- At constant production rate, if all plant power consumption crosses the threshold value limit and comes with higher values, that indicating increased power losses happening in plant due to less efficient electrical part or devices. The model will predict such situations as output power consumption category in "high".

- Such predictions will enable the plant operators to enforce predictive maintenance on inefficient electrical parts of plant and rectify extra power losses issues.

- IoT enabled monitoring will reduce the manual task of keeping a check on electricity consumption and with this model whenever the energy consumption will be high , user will automatically get the report and hence immediate steps then can be taken to decrease the further consumption

- It provides energy consumption data for each unit so this will give information about the unit at a particular instance which is decreasing the overall production efficiency

- By the individual electricity consumption it also will help in keeping a check over functionality of the electrical component and hence will add positively towards the overall efficiency of the production system