Artificial Intelligence Based Voltage Quality Detection and Estimation in Smart Home

Achshah R M, Dr.T.Ruban Deva Prakash

[1] Department of Artificial Intelligence & Data Science, Eswari Engineering College, Chennai, India; [2] Principal, Heera College of Engineering & Technology, Thiruvananthapuram, India.
Email: samuelachsha108@gmail.com

ABSTRACT

Smart homes are gaining more popularity in recent times as it improves the quality of living using Internet of Things (IoT) and Artificial Intelligence (AI). Most of the devices has smart sensors and electronic controllers which can interact with each other automatically via internet through automation technologies like Z-Wave, ZigBee and X10 to make life comfortable for users due to controllability from remote locations. Green energy sources are used for supplying power through power electronic converters which itself is a source of harmonics. The smart home devices often break down due to hardware failures which includes blown fuse, melted capacitors, failing circuit boards etc., thus making life miserable. Many problems commence from poor voltage quality. The need of the hour is intelligent controller based smart voltage conditioning device. The first step in designing conditioning device is selection of intelligent classification algorithm which detects the type of issue and activate the correct controller instantaneously. The controller generate switching pulses for power electronic compensating device based on estimation of voltage quality problem.Separate best fitting AI models are chosen for real time estimation of voltage magnitude related issues and harmonic pollution. The major quality issues identified are voltage sag, voltage swell, voltage flicker, momentary interruptions and harmonics. This paper compares decision tree classifier, random forest classifier and XG Boost Machine Learning (ML) algorithms for voltage quality classification model. Also the performance of linear regression, ridge regression, lasso regression, decision tree regressor, support vector machines and random forest regressor is analysed for real time estimation of voltage quality.

Keywords : Voltage quality; Voltage sag; Voltage swell; Momentary interuption; Harmonic pollution; Machine learning; Intelligent classification algorithm; Intelligent prediction algorithm; Decision tree; Random forest; XG Boost; Regression; Support vector machines

1 INTRODUCTION

SMART homes are integral part of smart micro-grid. Now a days, smart homes are becoming popular and size of global smart home market across world is expected to reach 7, 44,600 crores rupees with 26 crores smart homes by the end of 2021. Smart home penetration rate will reach 12.2% world-wide in the end of 2021. Smart home has devices has IP address and connected to a central home management system through internet. The user can control all devices remotely through mobile app. The devices has inbuilt smart sensors, cameras and computerized controllers. The devices can communicate among themselves through management system. The devices like washing machine, refrigerator, air conditioners, automatic door openers, electric vehicle chargers, electric oven etc has intelligent algorithms which can learn by itself and automatically adapt to the requirements and habits of user. Solar and wind electric generators are used for supplying power. They are also connected to smart micro-grid which facilitate bidirectional power flow. Home management system automatically controls the operating time of devices in energy saving mode. Time of demand tariff is under implementation in smart micro-grid to control demand according to availability of power. Hence devices like washing machine and electric vehicle chargers are operated in energy saving mode which are automatically switched on when the tariff rate is low. The refrigerators and other unnecessary loads are automatically switched off during peak load period at which cost of electricity is very high. The major issue with smart home are unnecessary shut down of computerized control systems, data loss and damage of devices due to poor voltage quality. Malfunctioning of controllers leads waste of time and cost. Overusage of computerized controls generate current harmonics. This pulsed current create flat topped voltage waveform which is rich in 3^{rd}, 5^{th}, and 7^{th} harmonic. This non-sinusoidal voltage affects the smooth accurate operation of different controllers leading to malfunction. Also the presence of voltage sag, voltage swell, voltage flicker and momentary interruptions leads to failure of sensitive devices. The life expectancy is unpredictable under low voltage quality. Blown fuse, melted capacitor and failing circuit board can amount to critical hardware failure.

L. Jiang, D-Y. Liu and B. Yang[1] in 2004 introduced the smart home system architecture along with the functionality of individual layers.He explained the communication between devices and microcontroller. M. Jahn and et.al. [2] in 2010 explained the issues related to power supply in smart home along with energy saving.Manohar Mishra [3] in 2019 proposed methods to detect and estimate power quality. He combined signal processing techniques in front end along with machine learning algorithms in which selection of algorithm is by trial and

error approach. Achshah and et.al. [4] in 2021 explained an approach to select the best algorithm for machine learning based on the result of exploratory data analysis.Thaha H. S and et.al [8] discussed power quality issues in composite micro-grid mainly due to integration green energy sources and highly non-linear loads. He proposed neural network based dynamic voltage restorer for mitigation [9]. T.Ruban Deva Prakash and et.al [7] in 2007 proposed intelligent method for harmonic estimation and elimination using shunt active filter. G.Justin Sunil Dhas and et.al [5] in 2012 explained a two neuron model for voltage flicker mitigation using generalized unified power flow controller. He also devised a method to detect voltage sag [6] in 2012 using artificial neural network. The general approach is to detect the type of quality issue and estimate its value for mitigation using different compensating devices. The above approaches are not suitable for real time application. This paper discusses AI models for real time detection of type of voltage quality issue and estimation.

Due to the increased use of sensitive load equipment in smart home and the awareness of customers about the commercial consequences of disturbances originating on the power supply system necessitated new controllers to improve voltage quality.It becomes inevitable to develop a novel voltage quality conditioning device which automatically senses and predicts the voltage quality problems which prevails in smart home and initiates proper control action to mitigate the same without human intervention, thus imparting self-healing feature to the supply system. The first step in designing voltage quality conditioning device is voltage quality classification. AI based ML classification algorithms are used for real time applications. Voltage quality issues like sag, swell, flicker, harmonics and interruption occurs at different time intervals. Different control algorithms are used for mitigating the issues using same compensating device. The corresponding algorithm should be activated instantly on occurrence of event. The proposed classification algorithm detects the nature of issue and activate the correct controller immediately. The controller needs estimation detail of voltage quality issue to generate switching pulses for power electronic compensating device. Hence separate estimation algorithmsare proposed for voltage magnitude issue and harmonic issue. The performance of different ML algorithms are tested for classification and estimation. The best fitting AI model is chosen for each case based on performance comparison.

2 OVERVIEW

Any power quality problem manifested in voltage, current or frequency deviations result in failure or malfunctioning of electrical equipment. Maintaining a sinusoidal waveform of bus voltages at rated voltage and rated frequency is referred as voltage quality. In the smart home system system, the increased use of nonlinear loads such as power electronics based devices, adjustable speed drives (ASD), personal computers (PCs), uninterruptable power supplies (UPS) and electric ovens create distorted (non-sinusoidal) currents even when supplied with purely sinusoidal voltage. Distorted currents can cause the voltage distortions throughout the system which deteriorate the quality of the electric power that is supplied to the equipment and neighboring houses. The main objective of this project is to develop a novel power quality conditioning device which automatically senses and predicts the voltage quality problems which prevails in smart home and initiates proper control action to mitigate the same without human intervention, thus imparting self-healing feature to the system. The device intelligent voltage quality conditioner (IVQC) can simultaneously solve all the above voltage quality problems associated with smart home such as voltage flicker, voltage sag, voltage swell and harmonic pollution. It has the capacity to provide both real and reactive power support, capacity to sense data from remote locations to estimate voltage quality, capacity to predict and diagnose the type of voltage quality problem, capacity to select necessary control technique automatically without human intervention and capacity to learn by experience. Machine learning classifier algorithms are used for classification or detection of prevailing voltage quality issues and regressor algorithms are used for estimation/ prediction of quality for mitigating the same. The basic structure of this work is outlined in figure-1.

The voltage in smart home is sampled at a time interval of 0. 556 milliseconds and fed to AI based classification algorithm. This algorithm detects the type of quality issue on real time basis and activate estimation algorithm. If the issue is voltage sag/swell/flicker/interruption AI based voltage magnitude estimation algorithm is activated which outputs the estimated value of the problem in terms of percentage of nominal value. In case of harmonic pollution the AI based harmonic estimation algorithm is activated that produces the magnitude of 3^{rd}, 5^{th} and 7^{th} harmonic components which are the dominating voltage harmonics in smart home as output in pu. The output of estimation algorithms is given to AI control algorithms to produce the switching pulses for compensator which is a voltage source inverter with dc link (battery) using sinusoidal pulse width modulation. If the voltage of system is less than nominal value the output of compensator is added to system voltage for compensation. If the system voltage is greater than nominal value the output of compensator has subtractive polarity to correct the voltage. In case of harmonic voltage, the negative value of harmonic voltages are injected to cancel the harmonic components thereby maintaining sinusoidal component of fundamental voltage. This paper demonstrates the classification and estimation algorithm. The performance of control algorithms is validated using digital simulation. Smart homes are basic building blocks of smart micro grid. A smart microgrid will provide greater control over energy costs, a more reliable energy supply for consumers, reduced peak demand, integration of more renewable power sources, and reduced CO_2 emissions and other pollutants. Smart grids with capability for integrating low carbon energy sources into power networks will provide more electricity to meet rising demand, increase energy efficiency, increase reliability and quality of power supplies. This work helps in increasing the oppurtunity for smart homes thereby smart microgrids.

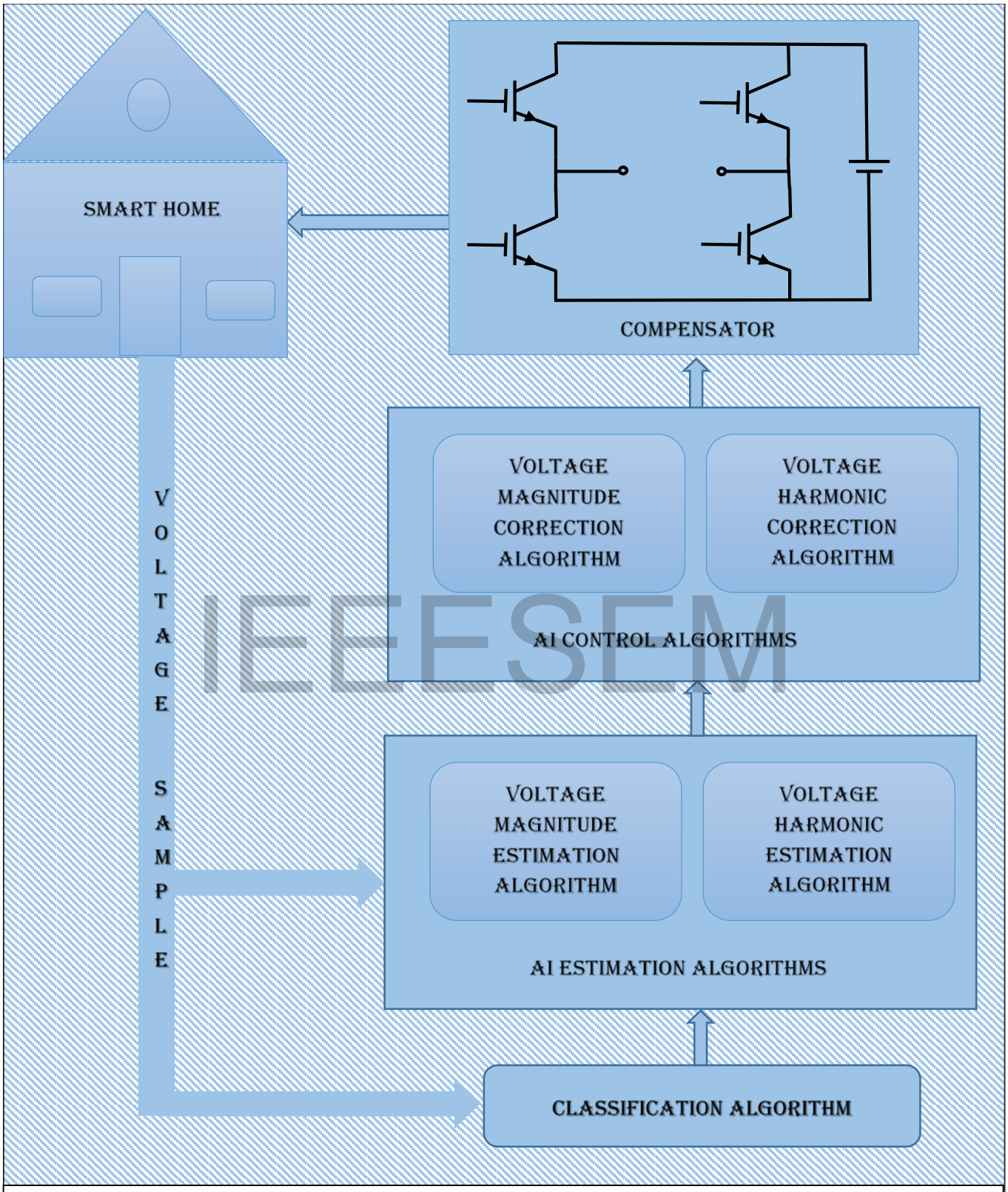


Fig. 1. Block **diagram of proposed system**

2 VOLTAGE HARMONICS

In order to ensure quality of power supply the voltage and current waveforms should be purely sinusoidal. But smart home has non-linear loads as most of the devices have computerized controllers. The switched mode power supply draws non-continuous current just like pulses. This pulse current train causes distortion in voltage waveform. The resulting voltage wave becomes non-sinusoidal with flat top almost like a trapezoidal shape. According to Fourier any non-sinusoidal waveform is composed several sine waves, one with frequency same as original non-sinusoidal waveform called fundamental waveand others with frequencies multiples of fundamental wave called harmonics as shown in equation (1). Fourier analysis of the resultant voltage wave has components with frequencies thrice, fifth and seventh time of fundamental called third harmonic, fifth harmonic and seventh harmonic respectively as shown in figure 2. Other order harmonics have negligibly small magnitudes and hence neglected. The 3rd, 5th and 7th harmonics are called the dominant harmonics. The flat topped voltage waveform is represented as

$$f(\omega t) = A_1 \sin\omega t + A_3 \sin 3\omega t + A_5 \sin 5\omega t + A_7 \sin 7\omega t \qquad (1)$$

Harmonics cause additional voltage drop, temperature rise, and increase in inductive inductance of feeder thereby affecting neighbouring houses. Harmonics in voltage affects the accurate operation of automatic controller which leads to malfunctioning of IoT based sensitive device in smart home. The presence of harmonic problem should be detected by AI classifier model and the magnitudes of A_3,A_5 and A_7 should be estimated using AI regression model. Selection of ML algorithm for classification and estimation is also discussed in this paper. Harmonic problem can occur simultaneously with other issues like flicker, sag and swell. Hence separate regression algorithm is designed for harmonic estimation.

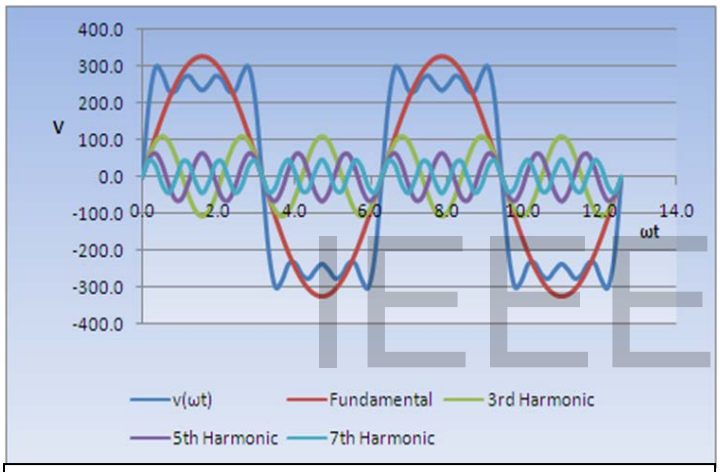


Fig. 2. Harmonic analysis of voltage waveform from smart home

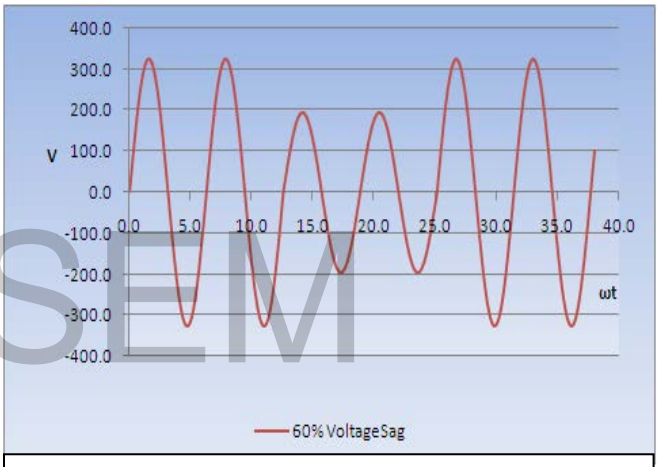


Fig. 3. Voltage sag waveform representing 60% sag for 40 ms

4 VOLTAGE SAG

Voltage sag is reduction in rms value of voltage from 10% to 90% for short interval ranging from 0.01s to 60s as shown in figure 3. When devices like iron box, heater, dryer etc are switched on, there is sudden flow of inrush current which causes in dip in voltage. During starting of motors, compressors in air conditioners and refrigerators, high starting current is drawn which is several times of rated current due to lack of back emf. This high starting current causes voltage drop. Also sag occurs due to momentary short circuit. Voltage sag causes malfunctioning of sensitive electronic controllers and tripping of loads thus interrupting the ongoing process. This makes automation in smart home useless unless it is detected and mitigated instantly.

5 VOLTAGE SWELL

Voltage swell is increase in rms value of voltage above normal as shown in figure 4. Magnitude of voltage increases in the range of 110% to 180% for a duration of 0.01s to 60s. The home management system automatically switches off all the unimportant loads including washing machine, electric vehicle charger etc when there is hike in electricity tariff. Under time of demand tariff scheme tariff rate is increased to manage demand during peak hours. Switching off of such loads leads to voltage swell. This causes data loss and stoppage of damage of sensitive equipment. Hence detection and estimation of swell is very important for mitigation.

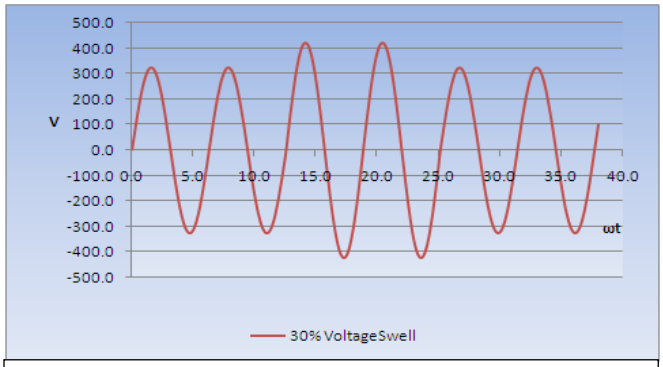


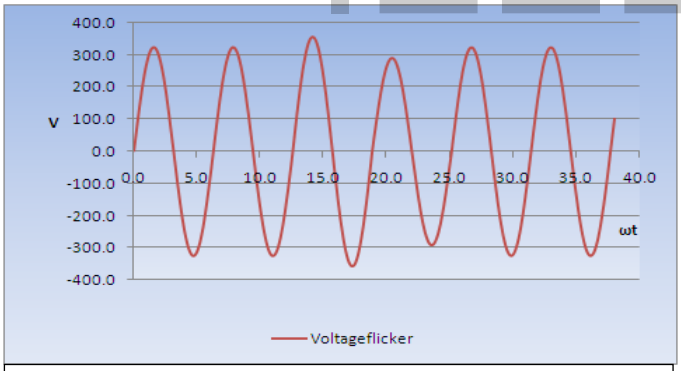
Fig. 4. Voltage swell waveform representing 30% swell for 40 ms

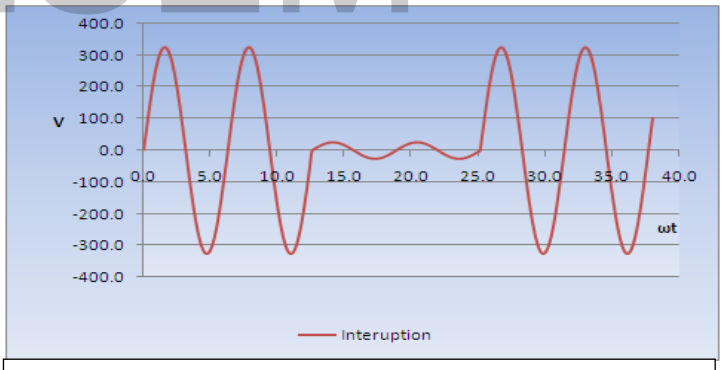
6 VOLTAGE FLICKER

Voltage flicker is continuous fluctuation in voltage between 90% of nominal value to 110% of nominal value. Figure 5 shows voltage flicker in third cycle with 10% increase above rated value and in fourth cycle with 10% decrease below rated value. Continuous voltage fluctuation causes flickering of lamps, TV screens and computer monitor which causes irritation to human eyes. Most of the devices have immunity against flicker, but under certain cases sensitive devices become susceptible to flicker. Voltage flicker reduces the life time of IoT technology based devices used in smart homes. Thus flicker mitigation is among the primary roles of voltage conditioning devices.

7 INTERRUPTIONS

Temporary decrease in voltage magnitude below 10% of rated value for duration less than 60s is considered as interruption in power quality. Figure 6 demonstrates voltage interruption in 230v, 50Hz smart home. The main cause of interruption is short circuit in electronic boards due to overheating of components. The voltage becomes normal immediately when the faulty part is isolated by fuse or breaker. The main cause of overheating of circuit components is the presence of harmonics which are high frequency components. Interruptions lead to unwanted shutdown of computerised devices in smart home leading to irritation of users and it will take certain time for restarting. Voltage should be fed by compensating device during interruption to avoid the above problems.





8. TRAINING DATA

In this paper three AI models are trained for detecting the type of voltage quality issue, estimation of harmonic components and estimation of sag/swell/interruption. The training data are generated by simulating the voltage quality problem using standard mathematical functions given below:

8.1 Voltage harmonics

The dominant harmonics for flat topped voltage wave are third, fifth and seventh. Hence the function for simulating harmonics due to computerised controllers in 230V smart home is

$$f(\omega t) = 325.269 * \sin(314 * 0.00056 * n) + k * 108.42 * \sin(3 * 314 * 0.00056 * n) + k * 65 * \sin(5 * 314 * 0.00056 * n) + k * 46.5 * \sin(3 * 314 * 0.00056 * n)$$

n=0,1,2,3,... and 0.8 < k < 1.2

Where 325.269 is the maximum voltage ($V_{rms} \times \sqrt{2}$) for 230V system, 314 is angular frequency (2πf) of 50Hz voltage, 0.00056s is the sampling time, n is the sample number and k is the modulation index.

8.2 Voltage sag

Voltage sag for single phase, 50Hz, 230V smart home system is simulated using

$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } 0 < t < t_1$$
$$v(\omega t) = 325.269 * m * \sin(314 * 0.00056 * n) \quad \text{for } t_1 < t < t_2$$
$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } t > t_2$$

Where t_1 is the instant of occurrence of sag and t_2 is the instant when voltage becomes normal, m is the percentage sag index which takes value from 0.1 to 0.9 and (t_2-t_1) ranges from 0.01s to 60s.

8.3 Voltage swell

Voltage sag for single phase, 50Hz, 230V smart home system is simulated using

$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } 0 < t < t_1$$
$$v(\omega t) = 325.269 * p * \sin(314 * 0.00056 * n) \quad \text{for } t_1 < t < t_2$$
$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } t > t_2$$

Where t_1 is the instant of occurrence of swell and t_2 is the instant when voltage becomes normal, p is the percentage swell index which takes value from 1.1 to 1.8 and (t_2-t_1) ranges from 0.01s to 60s.

8.4 Voltage flicker

Voltage flicker is simulated using the function

$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } 0 < t < t_1$$
$$v(\omega t) = 325.269 * g * \sin(314 * 0.00056 * n) \quad \text{for } t_1 < t < t_2$$
$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } t > t_2$$

Where t_1 is the instant of occurrence of flicker and t_2 is the instant when voltage becomes normal, g is the percentage flicker index which varies continuously between 0.9 and 1.1.

8.5 Interruption

Interruption is simulated using the function

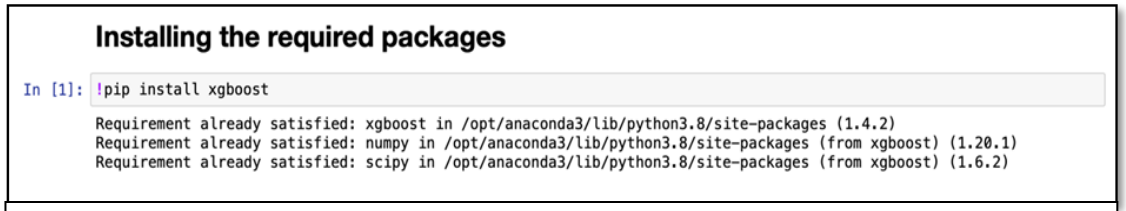
$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } 0 < t < t_1$$
$$v(\omega t) = 325.269 * f * \sin(314 * 0.00056 * n) \quad \text{for } t_1 < t < t_2$$
$$v(\omega t) = 325.269 * \sin(314 * 0.00056 * n) \quad \text{for } t > t_2$$

Where t_1 is the instant of occurrence of interruption and t_2 is the instant when voltage becomes normal, f is the percentage interruption index which takes value from 0.01 to 0.09 and (t_2-t_1) ranges from 0.01s to 60s.

9. Data Pre-processing and Exploratory Data Analysis (EDA)

The accuracy and performance of the machine learning model depends on the quality of the data. This makes data pre-processing very crucial. Around 80% of project time is spend in the data pre-processing as our model can only be as good as the data provided. As part of the

data pre-processing we first install the additional packages needed. Figure 7 shows XG Boost package being installed, which is required for XG Boost machine learning algorithm used, for the voltage quality classification model. There was no need for additional packages in case of voltage quality estimation model and voltage harmonic estimation model. Next, all libraries required is imported. Pandas, NumPy, Matplotlib, Seaborn and SciPy are some of the basic libraries we need for loading the data into a data frame, analysing, visualisation, cleaning, exploring and manipulating data. Scikit-learn is one of the famous free software machine learning library. Most of the algorithms and evaluation metrics we need for our models are from scikit-learn . The required libraries imported for the voltage quality classification model is listed in figure 8, for the voltage quality estimation model is shown in figure 9 and for voltage harmonic estimation model is listed in figure 10. Loading the dataset (.csv file) into the data frame using the pandas read_csv() function is the next step. If the dataset is not in the same folder as the coding file then path of the dataset also has to be mentioned. In figure 11, the Voltage Quality.csv file containing the training dataset and Voltage Quality Test.csv file containing the test dataset for the voltage quality classification model is being loaded into the data frame called data and test. The Regression data.csv file contains the dataset for both training and testing the voltage quality estimation model being loaded into data frame called data is represented in figure 12. The training and testing dataset for voltage harmonic estimation model is the Harmonic estimation.csv file. This file being loaded into data, a pandas data frame, is shown in figure 13.



Installing the required packages

```
In [1]: !pip install xgboost

        Requirement already satisfied: xgboost in /opt/anaconda3/lib/python3.8/site-packages (1.4.2)
        Requirement already satisfied: numpy in /opt/anaconda3/lib/python3.8/site-packages (from xgboost) (1.20.1)
        Requirement already satisfied: scipy in /opt/anaconda3/lib/python3.8/site-packages (from xgboost) (1.6.2)
```

Fig. 7. Installing the required package for voltage quality classification model from the coding file

Importing all the necessary libraries

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sn
        import matplotlib.pyplot as plt
        from sklearn.preprocessing import LabelEncoder
        from scipy import stats
        from sklearn.model_selection import train_test_split, GridSearchCV
        from sklearn.linear_model import LinearRegression, Ridge, Lasso
        from sklearn.svm import SVR
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_percentage_error

        import warnings
        warnings.filterwarnings('ignore')

        %matplotlib inline
```


Fig. 8. Importing the required libraries for voltage quality classification model from the coding file

Importing the required libraries

```
In [2]: import pandas as pd
        import numpy as np
        import seaborn as sn
        import matplotlib.pyplot as plt
        from matplotlib import rcParams
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.svm import SVC
        from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
        from xgboost import XGBClassifier
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.model_selection import GridSearchCV

        %matplotlib inline

        import warnings
        warnings.filterwarnings("ignore")
```

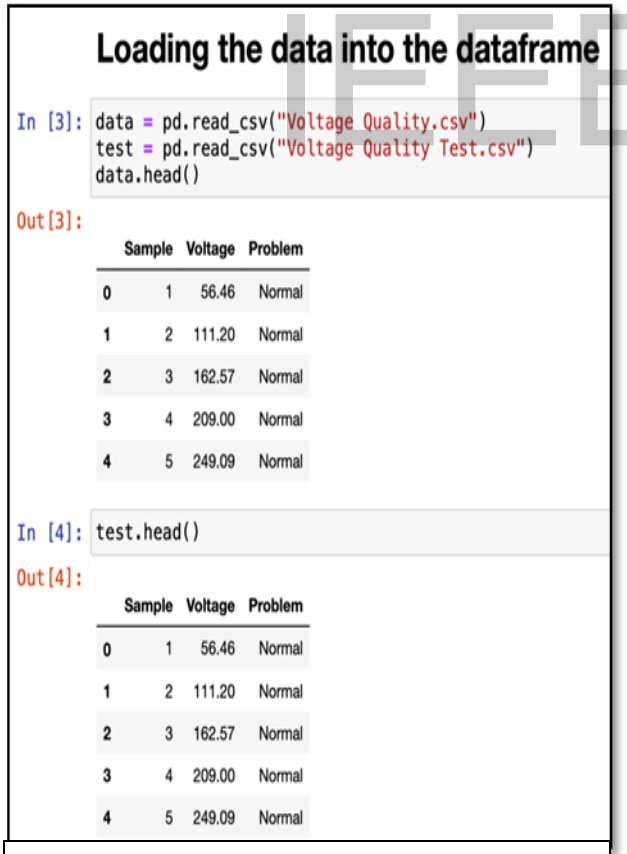
Fig. 9. Importing the required libraries for voltage quality estimation model from the coding file

After loading the dataset into the data frame we need to check if the data has been correctly loaded. Therefore, we use head () function from pandas to see the first 5 records. The same is shown in figure 10, figure 11 as well as figure 12. Next, we do exploratory data analysis to analyse the dataset and summarise their main characteristics, often using data visualisation methods. Using the shape function we will see the number of rows and columns in the dataset to know the size and dimension of the dataset. In the voltage quality classification model the training dataset contains 3366 rows and 3 columns whereas the testing dataset only contains 1020 records and 3 columns as shown in figure 14. The total number of rows is 2686 and columns is 3 in the voltage quality estimation model, which is displayed in figure 15. Figure 16 shows that the voltage harmonics estimation model has 3996 rows and 5 columns.

Importing all the necessary libraries

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sn
        import matplotlib.pyplot as plt
        from sklearn.preprocessing import LabelEncoder
        from scipy import stats
        from sklearn.model_selection import train_test_split, GridSearchCV
        from sklearn.linear_model import LinearRegression, Ridge, Lasso
        from sklearn.svm import SVR
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_percentage_error

        import warnings
        warnings.filterwarnings('ignore')

        %matplotlib inline
```

Fig. 10. Importing the required libraries for voltage harmonic estimation model from the coding file



Loading the data into the dataframe

```
In [3]: data = pd.read_csv("Voltage Quality.csv")
        test = pd.read_csv("Voltage Quality Test.csv")
        data.head()
```

Out[3]:

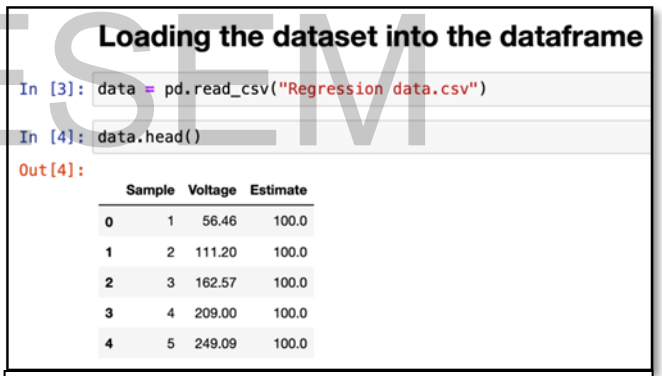
	Sample	Voltage	Problem
0	1	56.46	Normal
1	2	111.20	Normal
2	3	162.57	Normal
3	4	209.00	Normal
4	5	249.09	Normal

```
In [4]: test.head()
```

Out[4]:

	Sample	Voltage	Problem
0	1	56.46	Normal
1	2	111.20	Normal
2	3	162.57	Normal
3	4	209.00	Normal
4	5	249.09	Normal

Fig. 11. Loading the dataset into data frame and checking for voltage quality classification model from the coding file



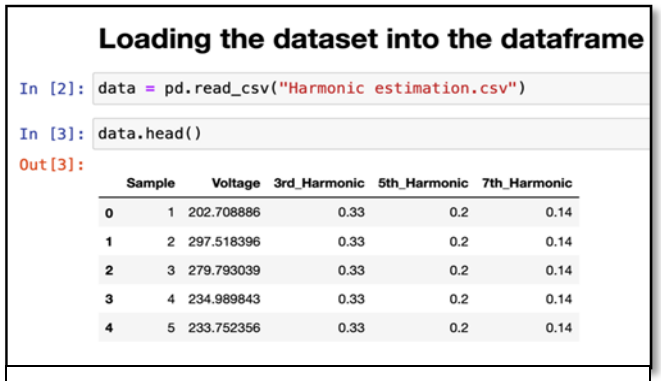
Loading the dataset into the dataframe

```
In [3]: data = pd.read_csv("Regression data.csv")
In [4]: data.head()
```

Out[4]:

	Sample	Voltage	Estimate
0	1	56.46	100.0
1	2	111.20	100.0
2	3	162.57	100.0
3	4	209.00	100.0
4	5	249.09	100.0

Fig. 12. Loading the dataset into data frame and checking for voltage quality estimation model from the coding file



Loading the dataset into the dataframe

```
In [2]: data = pd.read_csv("Harmonic estimation.csv")
In [3]: data.head()
```

Out[3]:

	Sample	Voltage	3rd_Harmonic	5th_Harmonic	7th_Harmonic
0	1	202.708886	0.33	0.2	0.14
1	2	297.518396	0.33	0.2	0.14
2	3	279.793039	0.33	0.2	0.14
3	4	234.989843	0.33	0.2	0.14
4	5	233.752356	0.33	0.2	0.14

Fig. 13. Loading the dataset into data frame and checking for voltage harmonic estimation model from the coding file

The data types of column is checked to see if any of the categorical data needs to be converted to numerical data but none of the models had categorical data that needed to be encoded. In figure 17 it can be seen that sample is integers, voltage is float and problem is object in voltage quality classification model. In the voltage quality estimation model, as shown in figure 15, sample is integers, voltage is float and estimate is float. Figure 16 shows that sample is integer, voltage, 3rd harmonic, 5th harmonic and 7th harmonic is float in the voltage harmonic estimation model.

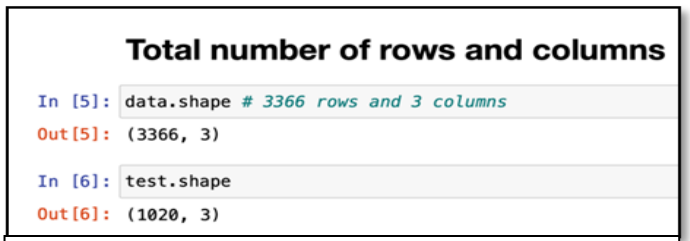


Figure 14 Checking the size of voltage quality classification model from

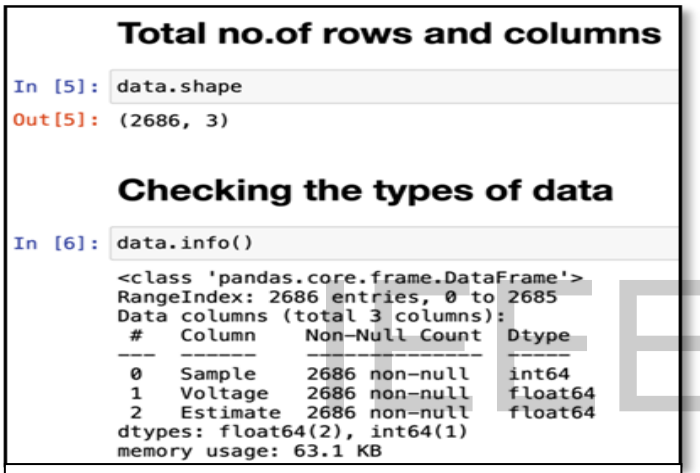


Fig. 15. Checking the size and data types of voltage quality estimation model from the coding file

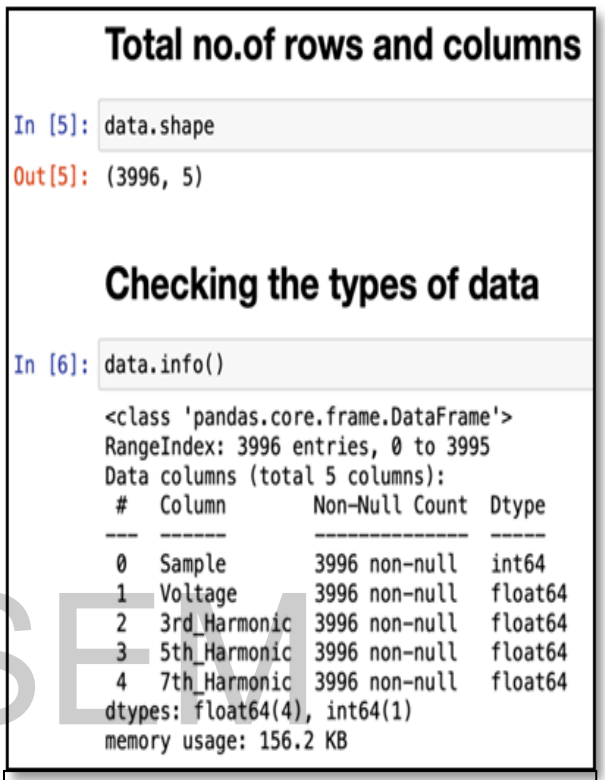


Fig. 16. Checking the size and data types of voltage harmonic estimation model from the coding file.

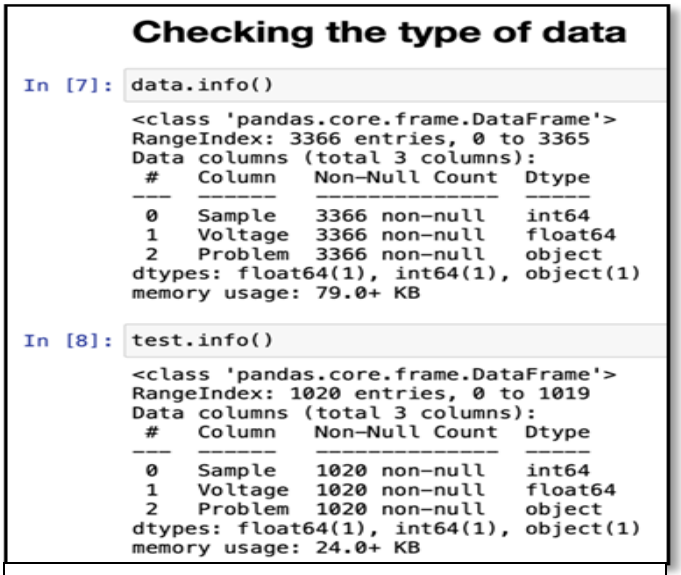


Fig. 17. Checking the data types of voltage quality classification model from the coding file

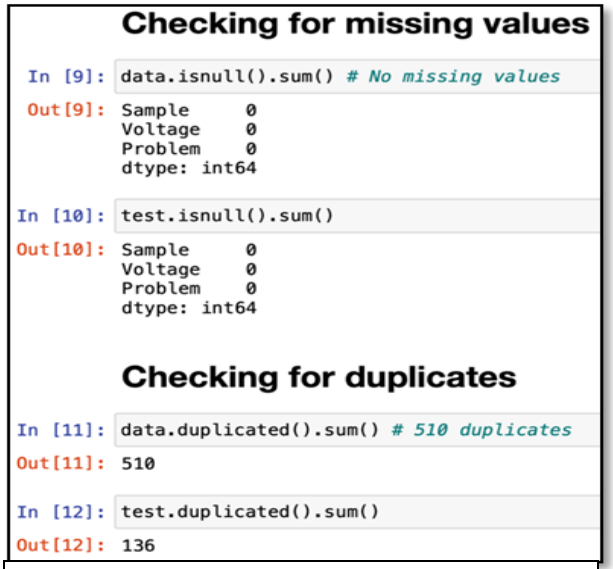


Fig. 18. Checking the missing values and duplicates of voltage quality classification model from the coding file

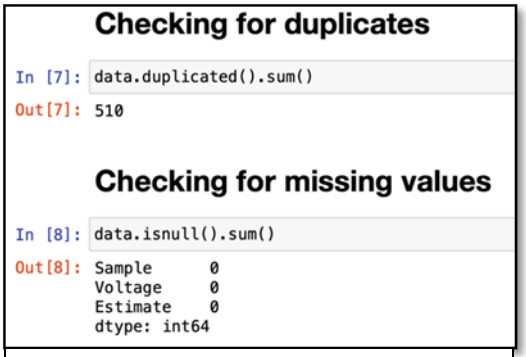


Fig. 19. Checking the missing values and duplicates of voltage quality estimation model from the coding file

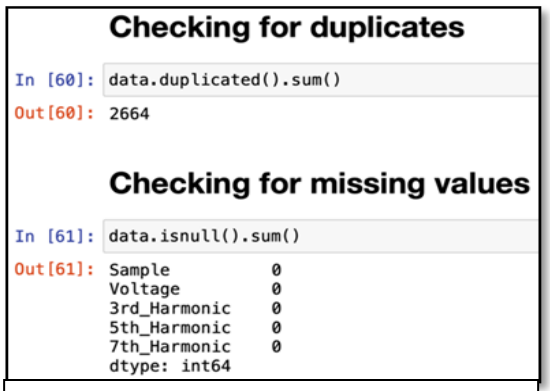


Fig. 20. Checking the missing values and duplicates of voltage harmonic estimation model from the coding file

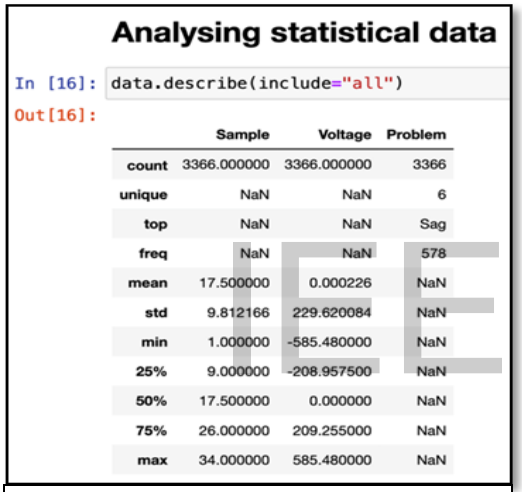


Fig. 21. Statistical information of voltage quality classification model from the coding file

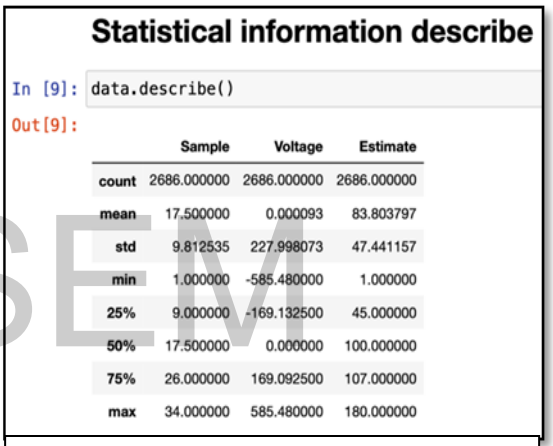
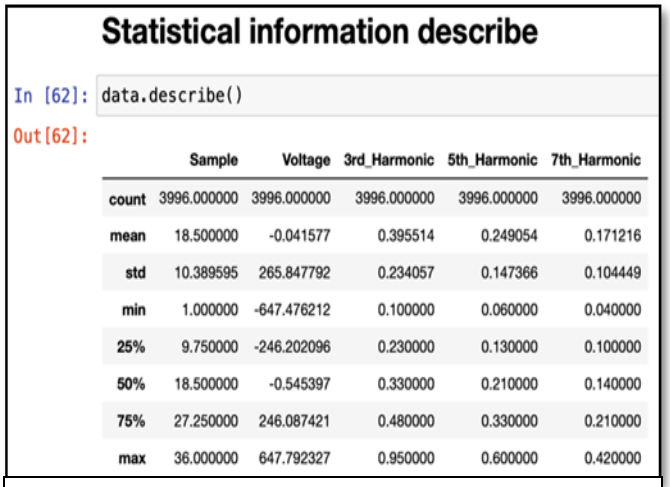


Fig. 22. Statistical information of voltage quality estimation model from the coding file

The data types of column is checked to see if any of the categorical data needs to be converted to numerical data but none of the models had categorical data that needed to be encoded. In figure 17 it can be seen that sample is integers, voltage is float and problem is object in voltage quality classification model. In the voltage quality estimation model, as shown in figure 15, sample is integers, voltage is float and estimate is float. Figure 16 shows that sample is integer, voltage, 3rd harmonic, 5th harmonic and 7th harmonic is float in the voltage harmonic estimation model. It is important to check for duplicates and missing values but if they need to be removed depends on the problem. There are no missing values in our models but there are duplicates, which does not need to be removed. Figure 18 shows the missing values and duplicates being checked for the voltage quality classification model. Checking missing values and duplicates for the voltage quality estimation model is illustrated in figure 19. Voltage harmonic estimation model being checked for missing values and duplicates is represented in figure 20.

We need to understand the dataset hence we analyse the statistical information like mean , standard deviation, first quartile, second quartile and third quartile etc. The describe() function will give the statistical data analysis. As it can be seen in figure 21, which shows the statistical data of the voltage quality classification model, the problem contains 6 unique values out which harmonic is the most occurring value. Figure 22 displays the statistical data of the voltage quality estimation model and figure 23 displays the statistical data of voltage harmonic estimation model. Outliers are the extreme values also known as anomalies. Outliers can affect the performance of our model so we have to remove all the outliers. However, none of our data contains outliers. The result of searching for outliers is shown in figures 24, 25 and 26. Data visualisation helps to understand the trend in the data and key feature in data. Figure 29 shows the box plot and distribution plot of voltage in the voltage quality classification model. The data seems to be normally distributed and as we already know there are no outliers seen in the box plot. The voltage histogram for voltage quality estimation model is shown in figure 27 and for voltage harmonic estimation

model is shown in figure 28. The figure 30 shows the number of records in each voltage quality problem and a bar chart to easily compare. This is very important as the data can be imbalanced in many cases and the model will become biased. If the data was imbalanced we will do techniques like Synthetic Minority Over-sampling Technique (SMOTE) but since our data is balanced we don't have to. A histogram of the estimation in voltage quality estimation model is draw in figure 31.



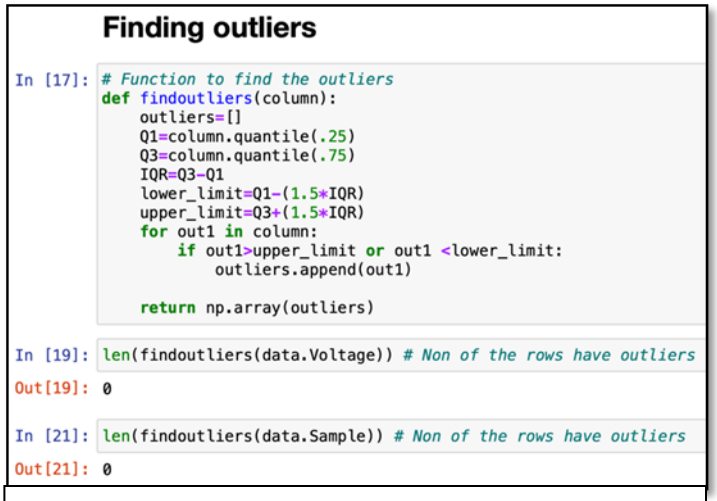
Statistical information describe

```
In [62]: data.describe()
```

Out[62]:

	Sample	Voltage	3rd_Harmonic	5th_Harmonic	7th_Harmonic
count	3996.000000	3996.000000	3996.000000	3996.000000	3996.000000
mean	18.500000	-0.041577	0.395514	0.249054	0.171216
std	10.389595	265.847792	0.234057	0.147366	0.104449
min	1.000000	-647.476212	0.100000	0.060000	0.040000
25%	9.750000	-246.202096	0.230000	0.130000	0.100000
50%	18.500000	-0.545397	0.330000	0.210000	0.140000
75%	27.250000	246.087421	0.480000	0.330000	0.210000
max	36.000000	647.792327	0.950000	0.600000	0.420000

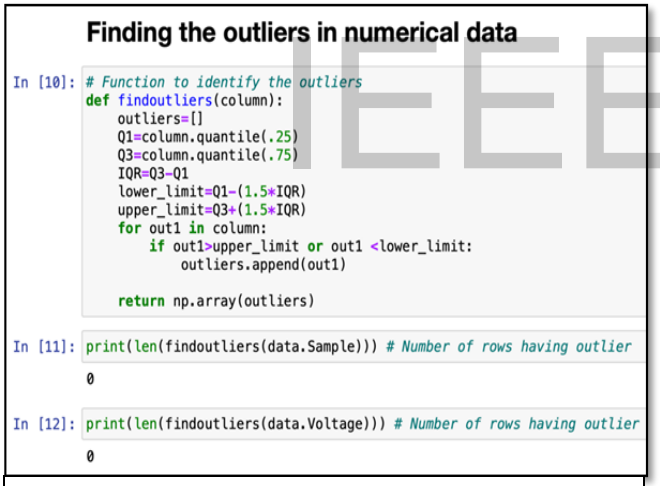
Fig. 23. Statistical information of voltage harmonic estimation model from the coding file



Finding outliers

```
In [17]: # Function to find the outliers
def findoutliers(column):
    outliers=[]
    Q1=column.quantile(.25)
    Q3=column.quantile(.75)
    IQR=Q3-Q1
    lower_limit=Q1-(1.5*IQR)
    upper_limit=Q3+(1.5*IQR)
    for out1 in column:
        if out1>upper_limit or out1 <lower_limit:
            outliers.append(out1)

    return np.array(outliers)

In [19]: len(findoutliers(data.Voltage)) # Non of the rows have outliers
Out[19]: 0

In [21]: len(findoutliers(data.Sample)) # Non of the rows have outliers
Out[21]: 0
```

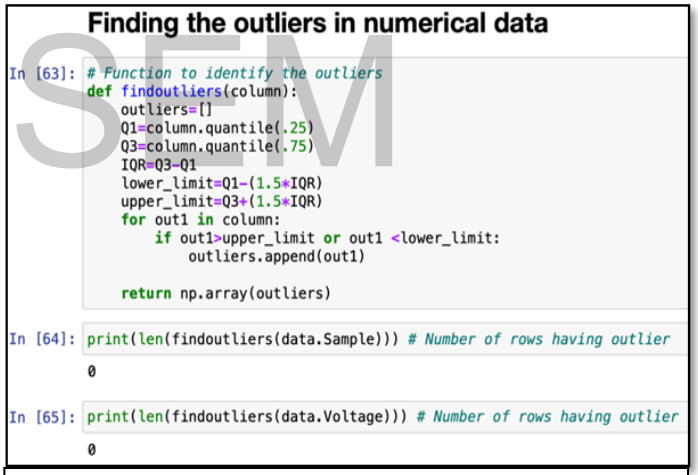
Fig. 24. Finding outliers of voltage quality classification model from the coding file



Finding the outliers in numerical data

```
In [10]: # Function to identify the outliers
def findoutliers(column):
    outliers=[]
    Q1=column.quantile(.25)
    Q3=column.quantile(.75)
    IQR=Q3-Q1
    lower_limit=Q1-(1.5*IQR)
    upper_limit=Q3+(1.5*IQR)
    for out1 in column:
        if out1>upper_limit or out1 <lower_limit:
            outliers.append(out1)

    return np.array(outliers)

In [11]: print(len(findoutliers(data.Sample))) # Number of rows having outlier
0

In [12]: print(len(findoutliers(data.Voltage))) # Number of rows having outlier
0
```

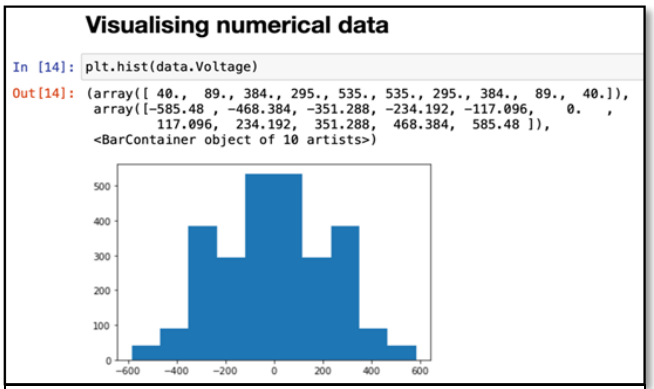
Fig. 25. Outliers of voltage quality estimation model from coding file



Finding the outliers in numerical data

```
In [63]: # Function to identify the outliers
def findoutliers(column):
    outliers=[]
    Q1=column.quantile(.25)
    Q3=column.quantile(.75)
    IQR=Q3-Q1
    lower_limit=Q1-(1.5*IQR)
    upper_limit=Q3+(1.5*IQR)
    for out1 in column:
        if out1>upper_limit or out1 <lower_limit:
            outliers.append(out1)

    return np.array(outliers)

In [64]: print(len(findoutliers(data.Sample))) # Number of rows having outlier
0

In [65]: print(len(findoutliers(data.Voltage))) # Number of rows having outlier
0
```

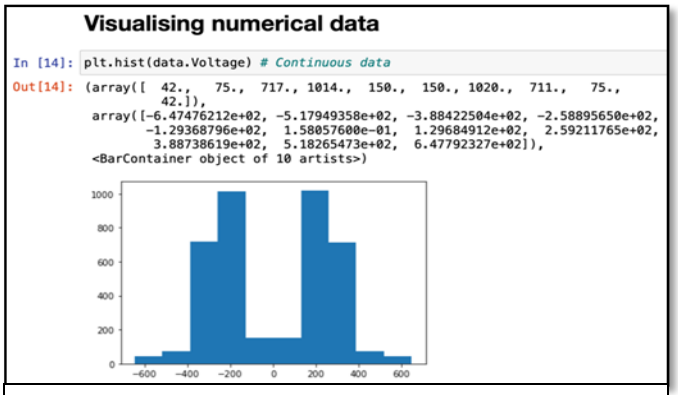
Fig. 26. Outliers of voltage harmonic estimation model from coding file



Visualising numerical data

```
In [14]: plt.hist(data.Voltage)
Out[14]: (array([ 40.,  89., 384., 295., 535., 535., 295., 384.,  89.,  40.]),
 array([-585.48 , -468.384, -351.288, -234.192, -117.096,    0.   ,
         117.096,  234.192,  351.288,  468.384,  585.48 ]),
 <BarContainer object of 10 artists>)
```


Fig. 27. Visualising voltage in voltage quality estimation model



Visualising numerical data

```
In [14]: plt.hist(data.Voltage) # Continuous data
Out[14]: (array([  42.,   75.,  717., 1014.,  150.,  150., 1020.,  711.,   75.,
          42.]),
 array([-6.47476212e+02, -5.17949358e+02, -3.88422504e+02, -2.58895650e+02,
        -1.29368796e+02,  1.58057600e-01,  1.29684912e+02,  2.59211765e+02,
         3.88738619e+02,  5.18265473e+02,  6.47792327e+02]),
 <BarContainer object of 10 artists>)
```


Fig. 28. Visualising voltage in voltage harmonic estimation model

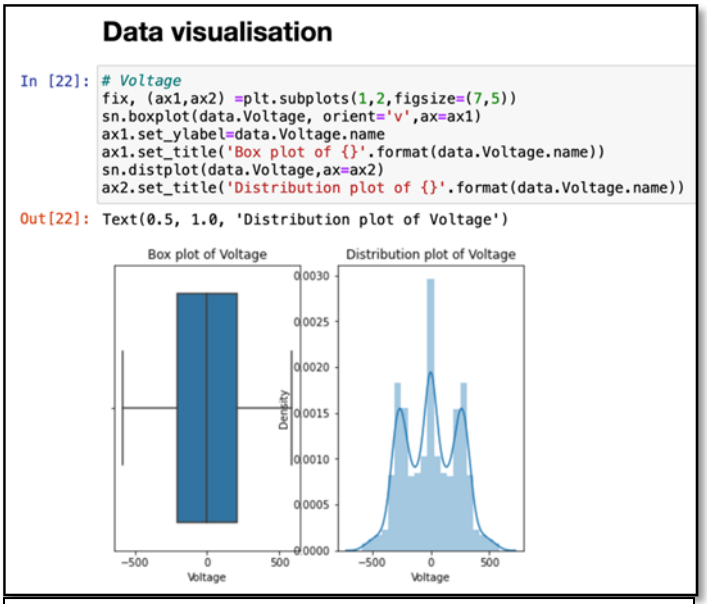


Fig. 29. Visualising voltage in voltage quality classification model

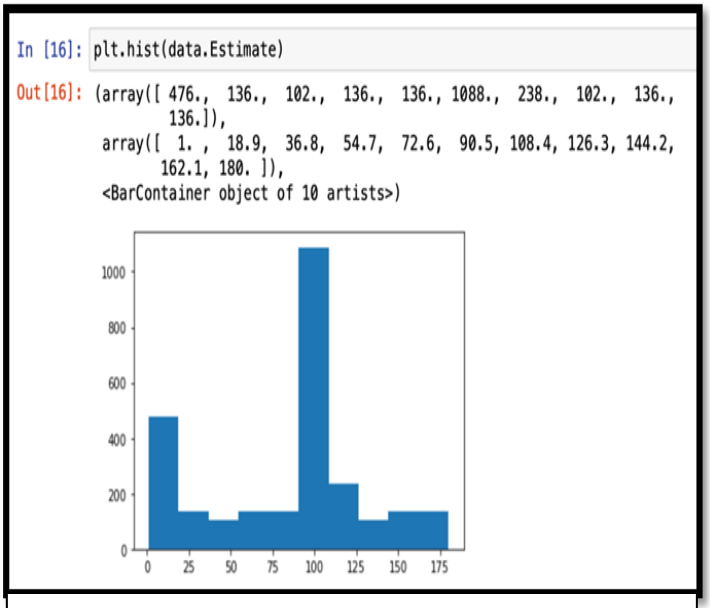


Fig. 31. Visualising estimation in voltage quality estimation model

Fig. 30. Visualising problem in voltage quality classification model from the coding file

10. ML ALGORITHMS FOR CLASSIFICATION

The independent and dependent variables in this problem should be defined. Figure 32 shows the independent variable, X, as sample and voltage from training dataset, dependent, y, as problem from the training dataset, independent variable for test, x_test, as sample and voltage from test data and dependent variable for test, y_test, as problem from test dataset. After defining the independent and dependent variable we can start with the algorithms. For our classification problem decision tree, XG Boost and random forest looks to be he accurate model. Therefore, we can train and test the algorithm using all three algorithms and choose the best algorithm for our model. Firstly, we have defined decision tree algorithm as model_dt. Next, trained the model_dt with our X and y. The code for same can be seen in figure 33. After training we need to test the performance of model_dt for which I have used accuracy score, classification report and confusion metrics. Classification report measures the quality of predictions. It shows a report of precision, recall, f1 score and support. Precision shows the accuracy of positive class. The equation 1 shows the equation used to calculate precision.

$$Precision = \frac{TP}{TP + FP}$$

Recall is also called as sensitivity or true positive rate. It computes the ratio of positive classes correctly predicted. Equation 2 shows the equation used to calculate recall.

$$Recall = \frac{TP}{TP + FN}$$

F1 score is the average of recall and precision. The equation 3 shows equation to calculate f1 score.

$$F1\ Score = \frac{Recall + Precision}{2}$$

Support is the number of occurrences of each class in the dataset. Confusion metrics shows a metrics which shows predicted vs actual count. The accuracy score, classification report and confusion metrics of model_dt is shown in the figure 34. The accuracy score is 95.39% for the model_dt and classification report also has above 90% for almost all except harmonics. In confusion metrics 170 records of flicker was correctly predicted as flicker, out of 170 records 149 was correctly predicted as harmonics, 170 was correctly predicted as interruption, 170 was correctly predicted as normal, from 170 records 157 was correctly predicted as sag and from 170 records 157 was correctly predicted as swell.

Support is the number of occurrences of each class in the dataset. Confusion metrics shows a metrics which shows predicted vs actual count. The accuracy score, classification report and confusion metrics of model_dt is shown in the figure 34. The accuracy score is 95.39% for the model_dt and classification report also has above 90% for almost all except harmonics. In confusion metrics 170 records of flicker was correctly predicted as flicker, out of 170 records 149 was correctly predicted as harmonics, 170 was correctly predicted as interruption, 170 was correctly predicted as normal, from 170 records 157 was correctly predicted as sag and from 170 records 157 was correctly predicted as swell. The same procedures are repeated for XG Boost and Random forest, look at figure 35 and 37. The accuracy score, classification report and confusion metrics for XG Boost is same as decision tree, view figure 36. The accuracy score for random forest is 97.75%. highest compared all the other models, which is clearly visible in bar chart, figure 38. We can finalize random forest and do hyper tuning to get higher accuracy score.

Grid search is used for hyper tuning rather than randomized search. The result of grid search can be seen in fig-

Fig. 32. Defining independent and dependent variable in voltage quality classification model

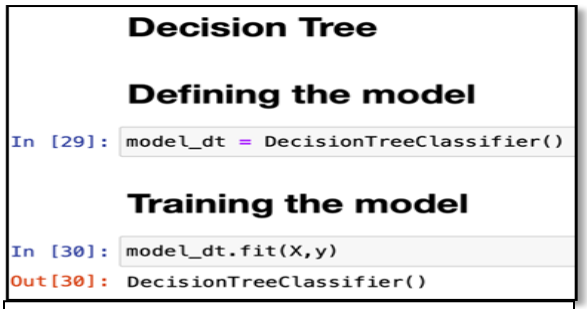
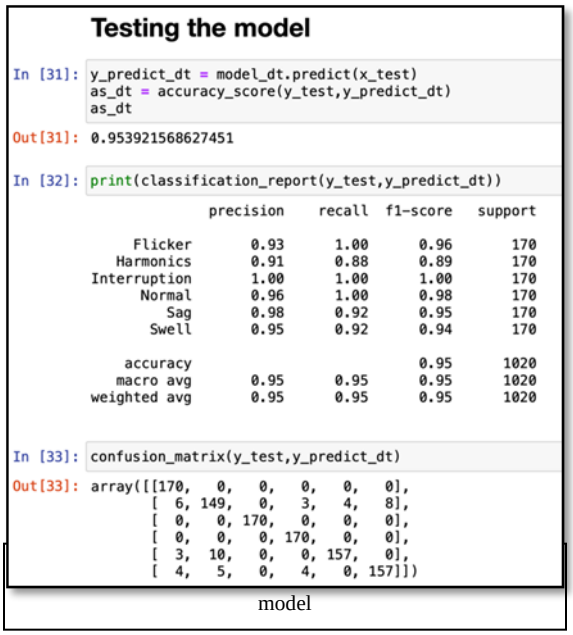


Fig. 33. Defining and training decision tree in voltage quality classification model

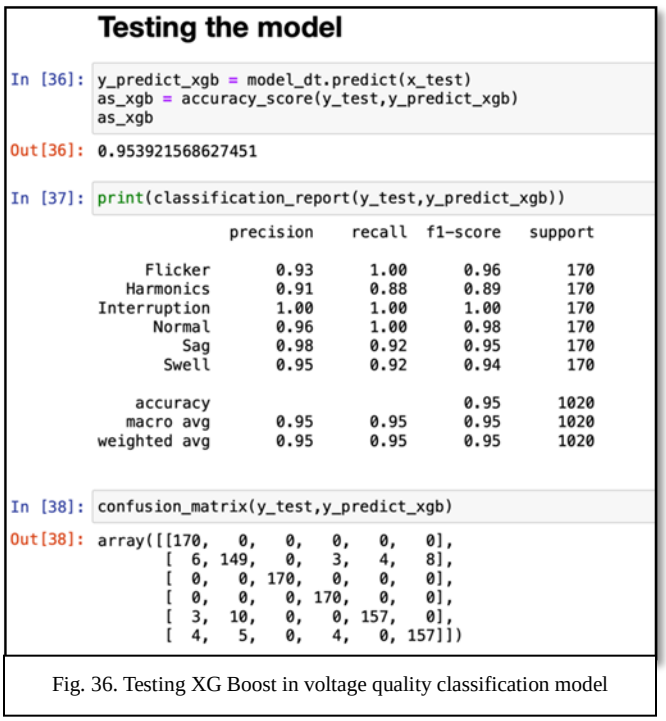
ure 39. The accuracy score of model is 98.92% after hyper tuning, look at figure 40.



Testing the model

```
In [31]: y_predict_dt = model_dt.predict(x_test)
         as_dt = accuracy_score(y_test,y_predict_dt)
         as_dt
Out[31]: 0.953921568627451

In [32]: print(classification_report(y_test,y_predict_dt))
                    precision    recall  f1-score   support

           Flicker       0.93      1.00      0.96       170
         Harmonics       0.91      0.88      0.89       170
      Interruption       1.00      1.00      1.00       170
            Normal       0.96      1.00      0.98       170
               Sag       0.98      0.92      0.95       170
             Swell       0.95      0.92      0.94       170

          accuracy                           0.95      1020
         macro avg       0.95      0.95      0.95      1020
      weighted avg       0.95      0.95      0.95      1020

In [33]: confusion_matrix(y_test,y_predict_dt)
Out[33]: array([[170,   0,   0,   0,   0,   0],
                [  6, 149,   0,   3,   4,   8],
                [  0,   0, 170,   0,   0,   0],
                [  0,   0,   0, 170,   0,   0],
                [  3,  10,   0,   0, 157,   0],
                [  4,   5,   0,   4,   0, 157]])
```

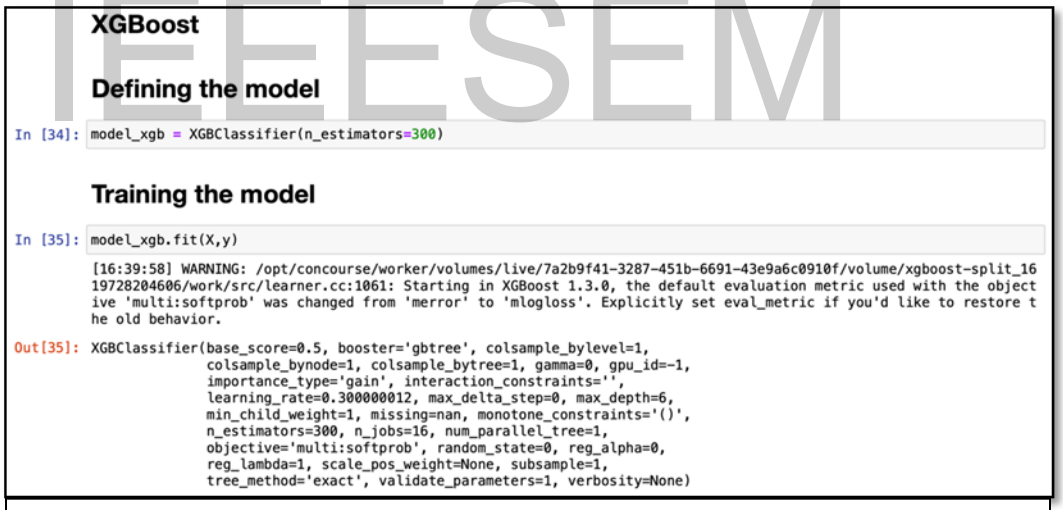
model



Testing the model

```
In [36]: y_predict_xgb = model_dt.predict(x_test)
         as_xgb = accuracy_score(y_test,y_predict_xgb)
         as_xgb
Out[36]: 0.953921568627451

In [37]: print(classification_report(y_test,y_predict_xgb))
                    precision    recall  f1-score   support

           Flicker       0.93      1.00      0.96       170
         Harmonics       0.91      0.88      0.89       170
      Interruption       1.00      1.00      1.00       170
            Normal       0.96      1.00      0.98       170
               Sag       0.98      0.92      0.95       170
             Swell       0.95      0.92      0.94       170

          accuracy                           0.95      1020
         macro avg       0.95      0.95      0.95      1020
      weighted avg       0.95      0.95      0.95      1020

In [38]: confusion_matrix(y_test,y_predict_xgb)
Out[38]: array([[170,   0,   0,   0,   0,   0],
                [  6, 149,   0,   3,   4,   8],
                [  0,   0, 170,   0,   0,   0],
                [  0,   0,   0, 170,   0,   0],
                [  3,  10,   0,   0, 157,   0],
                [  4,   5,   0,   4,   0, 157]])
```

Fig. 36. Testing XG Boost in voltage quality classification model



XGBoost

Defining the model

```
In [34]: model_xgb = XGBClassifier(n_estimators=300)
```

Training the model

```
In [35]: model_xgb.fit(X,y)
[16:39:58] WARNING: /opt/concourse/worker/volumes/live/7a2b9f41-3287-451b-6691-43e9a6c0910f/volume/xgboost-split_16
19728204606/work/src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metric used with the object
ive 'multi:softprob' was changed from 'merror' to 'mlogloss'. Explicitly set eval_metric if you'd like to restore t
he old behavior.

Out[35]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
                       colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
                       importance_type='gain', interaction_constraints='',
                       learning_rate=0.300000012, max_delta_step=0, max_depth=6,
                       min_child_weight=1, missing=nan, monotone_constraints='()',
                       n_estimators=300, n_jobs=16, num_parallel_tree=1,
                       objective='multi:softprob', random_state=0, reg_alpha=0,
                       reg_lambda=1, scale_pos_weight=None, subsample=1,
                       tree_method='exact', validate_parameters=1, verbosity=None)
```

Fig. 35. Defining and training XG Boost in voltage quality classification model

Grid search is used for hyper tuning rather than randomized search. The result of grid search can be seen in figure 39. The accuracy score of model is 98.92% after hyper tuning, look at figure 40.

11 ML ALGORITHMS FOR ESTIMATION

Similar to classification algorithm we need to define the independent and dependent variable. For the voltage quality estimation model sample and voltage are the independent variable and estimation is the dependent variable. Figure 41 shows the X and y variable being defined. For the voltage harmonic estimation model also sample and voltage are the independent variable but 3rd

harmonic, 5th harmonic and 7th harmonic is the dependent variable. This is displayed in figure 42. Next step is to split the data into training and testing dataset, for this we use train_test_split() function from sklearn.model_selection. We just simply pass our X, y, and test size or train size. Random state is also given to get a constant result. Figure 41 shows splitting for voltage quality estimation model and figure 42 shows splitting for voltage harmonic estimation model.

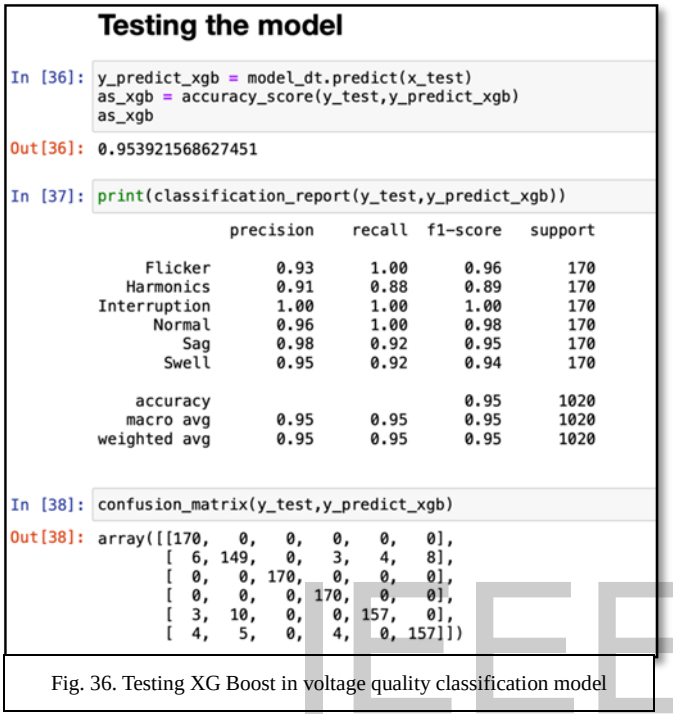


Fig. 36. Testing XG Boost in voltage quality classification model

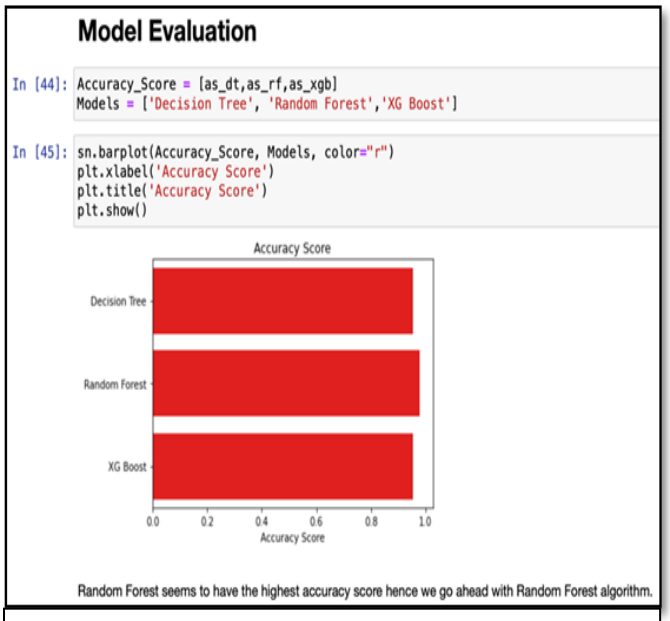


Fig. 38. Comparing the accuracy score of all the models in voltage quality classification

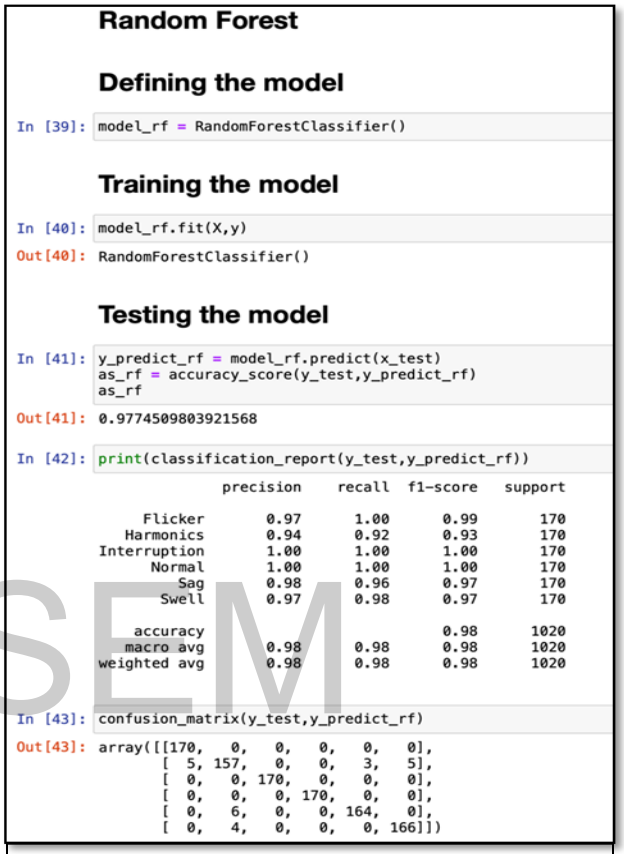


Fig. 37. Defining, training and testing random forest in voltage quality classification model

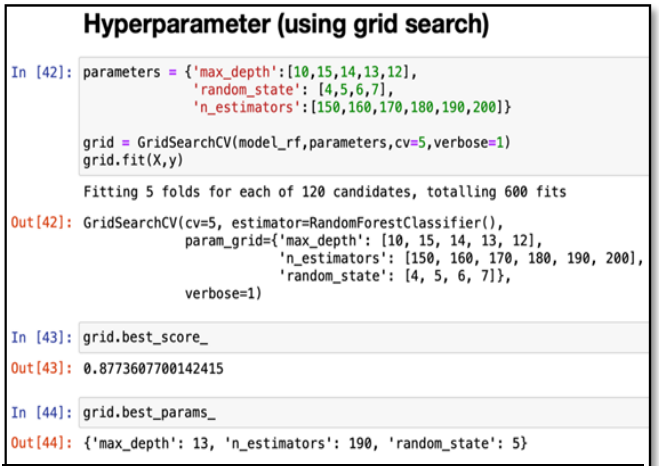


Fig. 39. Hypertuning in voltage quality classification model

Training Random Forest with best parameters

```
In [45]: model_rf = RandomForestClassifier(n_estimators=190,max_depth=13,random_state=5)
         model_rf.fit(X,y)
Out[45]: RandomForestClassifier(max_depth=13, n_estimators=190, random_state=5)
```

Testing Random Forest with best parameters

```
In [51]: y_predict = model_rf.predict(x_test)
         accuracy_score(y_test,y_predict)
Out[51]: 0.9892156862745098

In [52]: print(classification_report(y_test,y_predict))
                    precision    recall  f1-score   support

           Flicker       1.00      1.00      1.00       170
         Harmonics       0.94      1.00      0.97       170
      Interruption       1.00      1.00      1.00       170
            Normal       1.00      1.00      1.00       170
               Sag       1.00      0.96      0.98       170
             Swell       1.00      0.98      0.99       170

          accuracy                           0.99      1020
         macro avg       0.99      0.99      0.99      1020
      weighted avg       0.99      0.99      0.99      1020


In [53]: confusion_matrix(y_test,y_predict)
Out[53]: array([[170,   0,   0,   0,   0,   0],
                [  0, 170,   0,   0,   0,   0],
                [  0,   0, 170,   0,   0,   0],
                [  0,   0,   0, 170,   0,   0],
                [  0,   7,   0,   0, 163,   0],
                [  0,   4,   0,   0,   0, 166]])
```

Fig. 40. Random forest with best parameter in voltage quality classification model

Defining independent and dependent

```
In [17]: X = data.iloc[:,:-1]
         y = data.Estimate
```

Spliting the data into training and testing data

```
In [20]: x_train, x_test, y_train, y_test = train_test_split( X, y, train_size = 0.6, random_state = 23)
```

Fig. 41 Defining and splitting the data in voltage quality estimation model from the coding file

Defining independent and dependent

```
In [15]: X = data.iloc[:,:-3]
         y = data.iloc[:,-3:]
```

Spliting the data into training and testing data

```
In [18]: x_train, x_test, y_train, y_test = train_test_split( X, y, train_size = 0.6, random_state = 23)
```

Fig. 42. Defining and splitting the data in voltage harmonic estimation model from the coding file

Next like for classification we have to define, train and test the data with different algorithms. R2 score and mean absolute percentage error are metrics used to evaluate the performance of the model. It is amount of variation in the target predicted variable and actual target variable. Figure 43 shows linear regression being defined, trained and tested for the voltage quality estimation model. Ridge regression is defined, trained and tested in figure 44 for the voltage quality estimation model. Lasso regression for voltage quality estimation model is defined trained and tested in figure 45. Figure 46 shows SVM being defined, trained and tested for the voltage quality estimation model. Decision tree is defined, trained and tested in figure 47 for the voltage quality estimation model. Random forest for voltage quality estimation model is defined trained and tested in figure 48. Then we compare the results from all the algorithms using bar chart as shown in figure 49 and figure 50. The r2 score has to be highest and mean absolute percentage error should be the lowest for best performance of the model. Random forest seems to give the best performance out of all the algorithms hence we choose random forest.

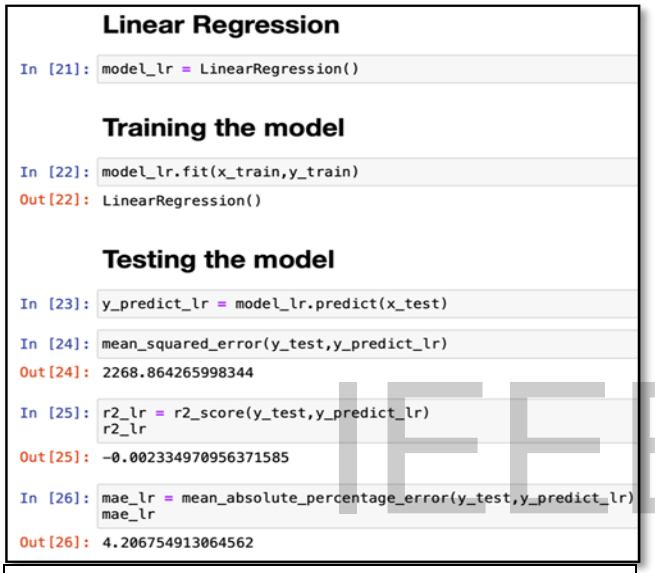


Fig. 43. Defining training and testing linear regression in voltage quality estimation model

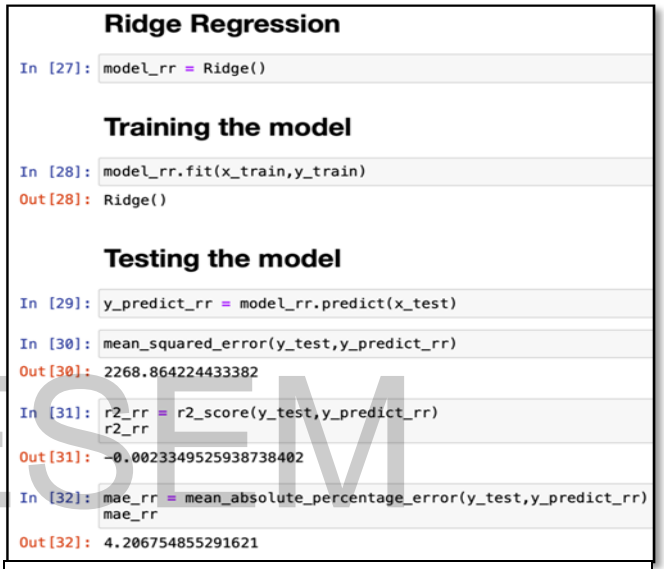


Fig. 44. Defining training and testing ridge regression in voltage quality estimation model

Fig. 45. Defining training and testing lasso regression in voltage quality estimation model

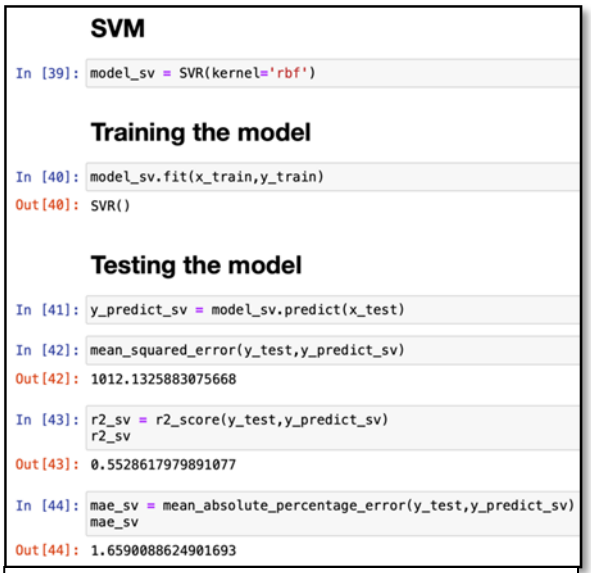


Fig. 46. Defining training and testing SVM in voltage quality estimation model

Decision Tree

```
In [45]: model_dt = DecisionTreeRegressor()
```

Training the model

```
In [46]: model_dt.fit(x_train,y_train)
Out[46]: DecisionTreeRegressor()
```

Testing the model

```
In [47]: y_predict_dt = model_dt.predict(x_test)

In [48]: mean_squared_error(y_test,y_predict_dt)
Out[48]: 27.834883720930232

In [49]: r2_dt = r2_score(y_test,y_predict_dt)
         r2_dt
Out[49]: 0.9877031527253058

In [50]: mae_dt = mean_absolute_percentage_error(y_test,y_predict_dt)
         mae_dt
Out[50]: 0.06356220573501774
```

Fig. 47. Defining training and testing decision tree in voltage quality estimation model

Random Forest

```
In [51]: model_rf = RandomForestRegressor()
```

Training the model

```
In [52]: model_rf.fit(x_train,y_train)
Out[52]: RandomForestRegressor()
```

Testing the model

```
In [53]: y_predict_rf = model_rf.predict(x_test)

In [54]: mean_squared_error(y_test,y_predict_rf)
Out[54]: 17.379236209302327

In [55]: r2_rf = r2_score(y_test,y_predict_rf)
         r2_rf
Out[55]: 0.9923222307820914

In [56]: mae_rf = mean_absolute_percentage_error(y_test, y_predict_rf)
         mae_rf
Out[56]: 0.057738086315154445
```

Fig. 48. Defining training and testing random forest in voltage quality estimation model

Model Evaluation

```
In [57]: Mean_Absolute_Percentage_Error = [mae_lr,mae_rr,mae_ll,mae_sv,mae_dt,mae_rf]
         R2_Score = [r2_lr,r2_rr,r2_ll,r2_sv,r2_dt,r2_rf]
         Models = ['Linear Regression', ' Ridge Regression' , ' Lasso Regression', 'Support Vector Machine',
                   'Decision Tree', 'Random Forest']
```

Mean Absolute Percentage Error Comparison Chart

```
In [58]: sn.barplot(Mean_Absolute_Percentage_Error, Models, color="m")
         plt.xlabel('Mean Absolute Percentage Error')
         plt.title('Mean Absolute Percentage Error')
         plt.show()
```

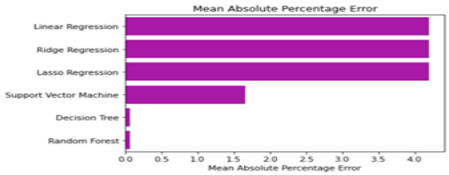
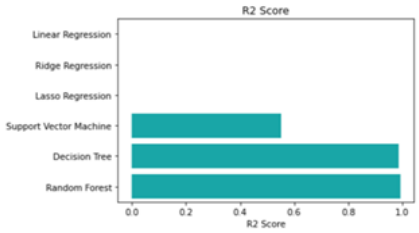


Fig. 49. Comparing chart in voltage quality estimation model

R2 Score Comparison Chart

```
In [59]: sn.barplot(R2_Score, Models, color="c")
         plt.xlabel('R2 Score')
         plt.title('R2 Score')
         plt.show()
```



From the charts it can be seen that the Mean Absolute Percentage Error is lowest for Random Forest and the R2 Score is highest for random forest. Therefore, we can conclude that random forest is the best algorithm for this data.

Fig. 50. Comparing chart in voltage quality estimation model

Hyper tuning is done to see if we can improve the model performance. We use grid search that was used in classification model. Figure 51 shows the result of hyper tuning. Figure 52 shows the result after training the model with best parameters. The same process is repeated for voltage harmonic estimation model as well. Figure 53 and 54 shows the comparison chart of all the algorithms for voltage harmonic estimation model. Decision tree seems to give best results for voltage harmonic estimation model hence we finalize decision tree algorithm for this model, look at figure 55.

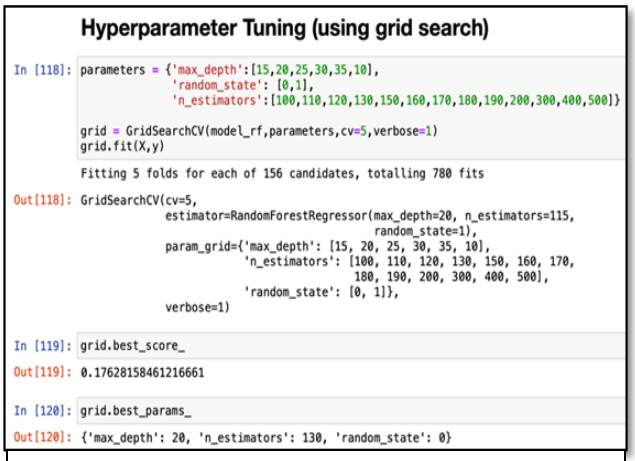


Fig. 51. Hypertuning in voltage quality estimation model

Fig. 52. Random forest in voltage quality estimation model

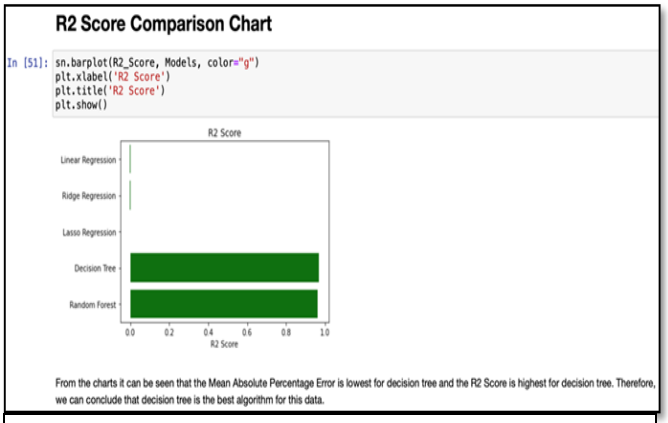


Fig. 54. Comparing chart in voltage harmonic estimation model

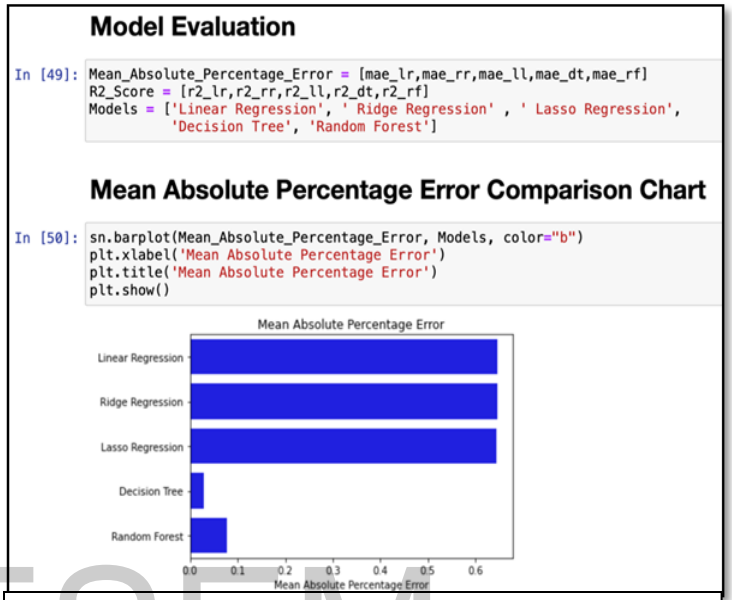


Fig. 53. Comparing chart in voltage harmonic estimation model

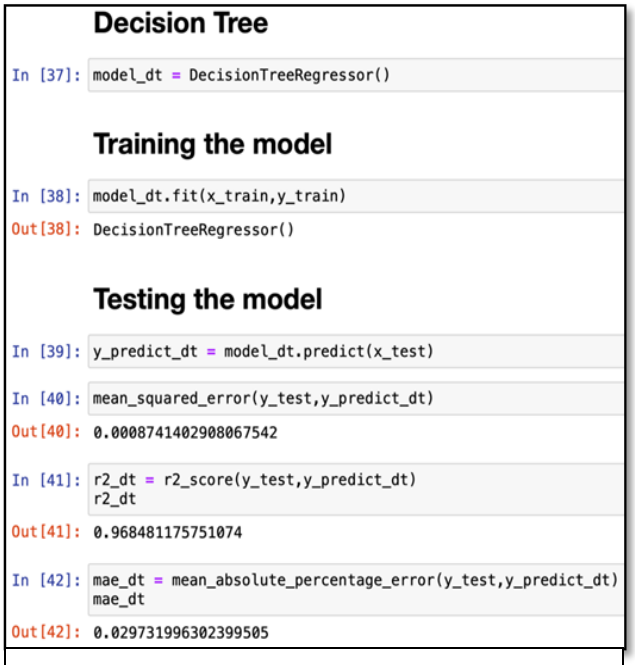


Fig. 55. Defining, training and testing decision tree in voltage harmonic estimation model

12 VALIDATION

The output obtained from the estimator at the beginning of sampling interval is added at a time delay of 0.28 millisecond i.e. at the middle of sampling interval. The output of compensator is assumed to remain constant from middle of one sampling interval to the middle of next sampling interval. The waveforms with and without compensator is shown below.

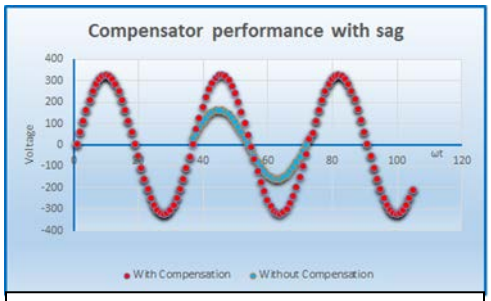


Fig. 56. Compensator performance during sag

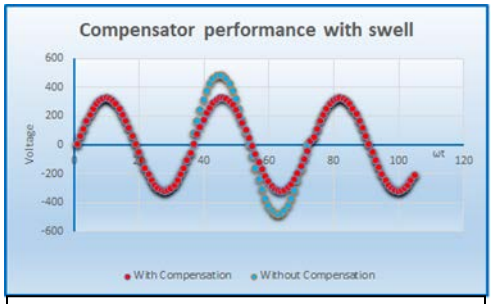


Fig. 57. Compensator performance during swell

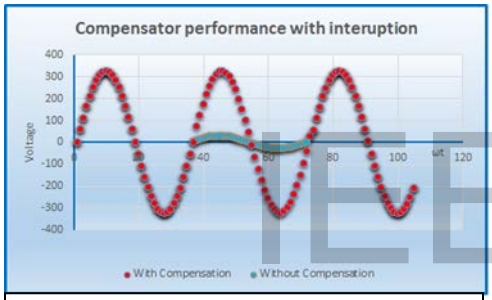


Fig. 58. Compensator performance during interruption

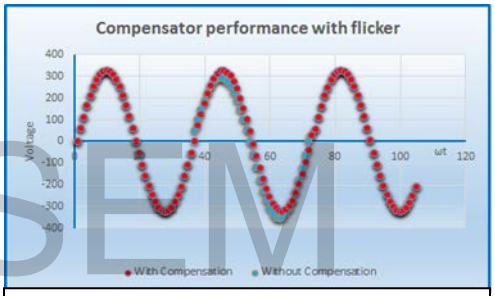


Fig. 59. Compensator performance during flicker

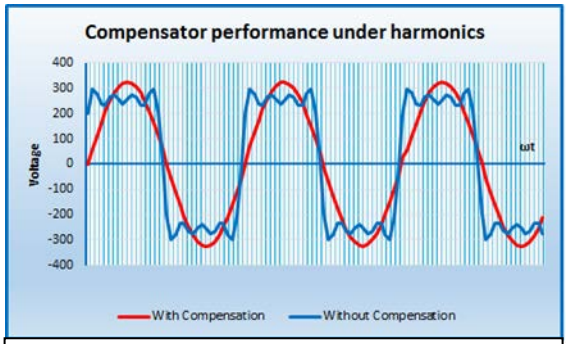


Fig. 60. Compensator performance during harmonic pollution

13 CONCLUSION

Training data of voltage is generated against sampling intervals by simulating various issues like sag, swell, flicker, interruption and harmonics using standard mathematical functions. Based on the results of Explorative Data Analysis (EDA) process for the generated test data decision tree classifier, random forest classifier and XG Boost Machine Learning (ML) algorithms are selected for voltage quality classification problem. It is concluded that random forest is best suited with accuracy score of 99%.Linear regression, ridge regression, lasso regression, decision tree regressor, support vector machines and random forest regressor are

trained for estimation by generating training sets using simulation. Random forest regressor outperformed others with an accuracy score of 99%. The output generated by AI classifier model on real time basis activates the correct control algorithm and AI regression model estimator generates switching pulses for compensating device using controller. Thus, this project benefits the user by reducing field service costs, down time of devices, product damage and stress. It increases competitiveness and opportunity of smart home system. This initiative helps the power sector in maintaining clean and green power, reducing energy wastage and increasing life time of power system components. The coding files are uploaded to github, click the link to view:
https://github.com/Achshah-RM/VOLTAGE-QUALITY-CLASSIFICATION-MODEL-
https://github.com/Achshah-RM/VOLTAGE-QUALITY-ESTIMATION-MODEL
https://github.com/Achshah-RM/VOLTAGE-HARMONIC-ESTIMATION-MODEL

REFERENCES

[1] L. Jiang, D-Y. Liu and B. Yang, "Smart home research," IEEE Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29, pp. 659 – 663, August 2004 .
[2] M. Jahn, M. Jentsch, C. R. Prause, F. Pramudianto, A. Al-Akkad & R. Reiners, "The energy aware smart home," 5th International Conference on Future Information Technology (FutureTech), 2010.
[3] Manohar Mishra,"Power quality disturbance detection and classification usingsignal processing and soft computing techniques: A comprehensive review",International Transaction Electrical Energy System, John Wiley & Sons, Ltd., 2019
[4] Achshah R M, Dr.T.Ruban Deva Prakash, "A Simple Approach for Selecting the Best Machine Learning Algorithm",International Journal of Scientific & Engineering Research, ISSN 2229-5518, Volume 9, Issue 12, September-2021.
[5] G.Justin Sunil Dhas,Dr.T.Ruban Deva Prakash, "Two Neuron Model for Voltage Flicker Mitigation Using Generalized Unified Power Flow Controller", International Journal of Current Engineering and Technology (ISSN 2277 – 4106), June, 2012
[6] G.Justin Sunil Dhas, Dr.T.Ruban Deva Prakash, "Voltage Sag Source Location Using Artificial Neural Network", International Journal of Current Engineering and Technology (ISSN 2277 - 4106), March, 2012
[7] T.Ruban Deva Prakash,Dr.N.KesavanNair"A New Control Algorithm for Harmonic Elimination Using Shunt Active Filter", Journal on Electrical Engineering (ISSN :2230 – 7176), July, 2007
[8] Thaha H. S,Dr.T.Ruban Deva Prakash, "Use of neural network based DVR for the reduction of power quality issues in composite micro-grid", Journal of Ambient Intelligence and Humanized Computing, ISSN:1868-5137, June 2020
[9] Thaha H. S, Dr.T.Ruban Deva Prakash, "Reduction of Power Quality Issues in Micro-Grid using Neural Network Based DVR", International Journal of Innovative Technology and Exploring Engineering (IJITEE), ISSN: 2278-3075, Volume-9 Issue-6, April 2020