Problem Statement

01

Predicting vibrations

Prediction Of Vibration of certain equipments and classify and SAFE, MODERATE, HIGH. On the basis of all other controllable and operating parameters.

02

Important features for vibrations

Find the important controllable parameters for the prediction of the vibrations of the four equipments so that they can be used under control.

03

Important parameters for specific energy

An ML prediction model is required to understand which parameters (operating + controllable) significantly contribute to the ‘specific energy’

04

Predicting Specific Energy

Find out the minimum number of variables used to predict the specific energy consumption, and create a prediction model based on these variables.

Steps for solving

Data Preprocessing

- Removing NaN values and outliers.
- Finding relevant columns for ML prediction using PCA.

ML model for vibrations

Making an ML model for prediction of vibration using **Ordinary Least Square method**.

Finding Critical Factors

With the use of **Random Forest Regressor** finding the important features for the prediction of the specific energy

Predicting

With the use of **Principle Component Analysis (PCA)** we find the minimum features and then use Random Forest for prediction.

1. Data Preprocessing

Handling Empty Columns and ILL values:

- Our Dataset had a lot of ill values like #REF! , #N/A ,#VALUE! etc , these all values were first replaced/removed depending on their number.
- also there were some values whose type was ‘str’ rather than ‘float’ so we **converted them to numeric values**.

c206	c207	c208	c209	c210	c211	c212	c213
#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
#REF!	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A

```
dropcol=[]
j=0
for i in df.columns:
    if pd.isnull(df[i]).all()==True:
        dropcol.append(i)
        j=j+1
        df.drop(i,axis=1,inplace=True)
```

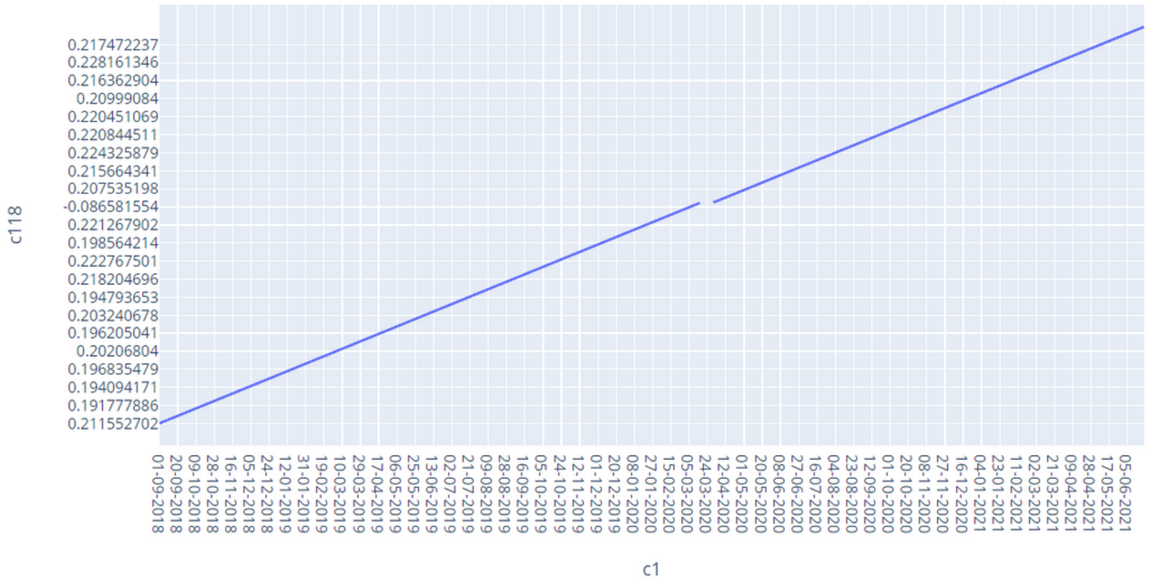
```
dropcol
```

```
['c199', 'c202', 'c204', 'c226', 'c229']
```

- We removed the columns containig all null values.
- Also, we had some columns containing over 800 null values/zeroes , we tried to manage these columns without dropping them immediately.

Handling Empty Columns and ILL values:

- After conversion of string to nan we also had to change the datatype of that column to numeric else datatype was object.
- It evaluated all the values as string rather than float and gave linear graph.




```
: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Columns: 240 entries, c1 to c241
dtypes: float64(213), int64(6), object(21)
memory usage: 1.9+ MB
```

```
: new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Columns: 235 entries, c1 to c241
dtypes: float64(233), int64(1), object(1)
memory usage: 1.8+ MB
```

- Thus we changed using pd.to_numeric method.

Handling Constant Columns:

We also noticed that:

- There were some columns which were constant throughout
- To make a good ML model we removed Those columns.
- There were around 8 such columns named Constant in the below code

```
constant=[]
for col in features:
    if(new_df[col].std()==0):
        constant.append(col)
        new_df.drop(col,axis=1,inplace=True)
print(constant)
print(new_df)
```

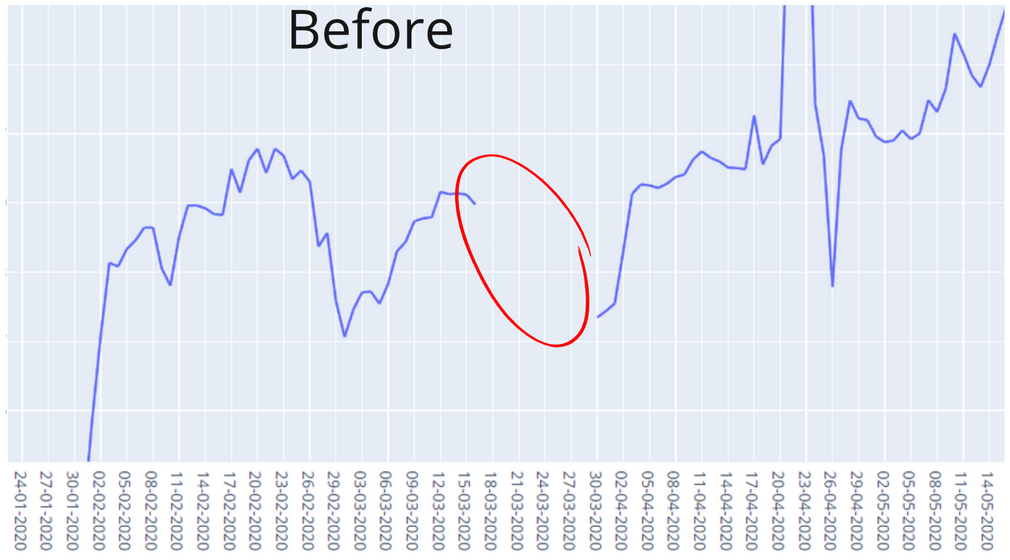
```
['c2', 'c82', 'c110', 'c156', 'c188', 'c189', 'c190', 'c206']
```

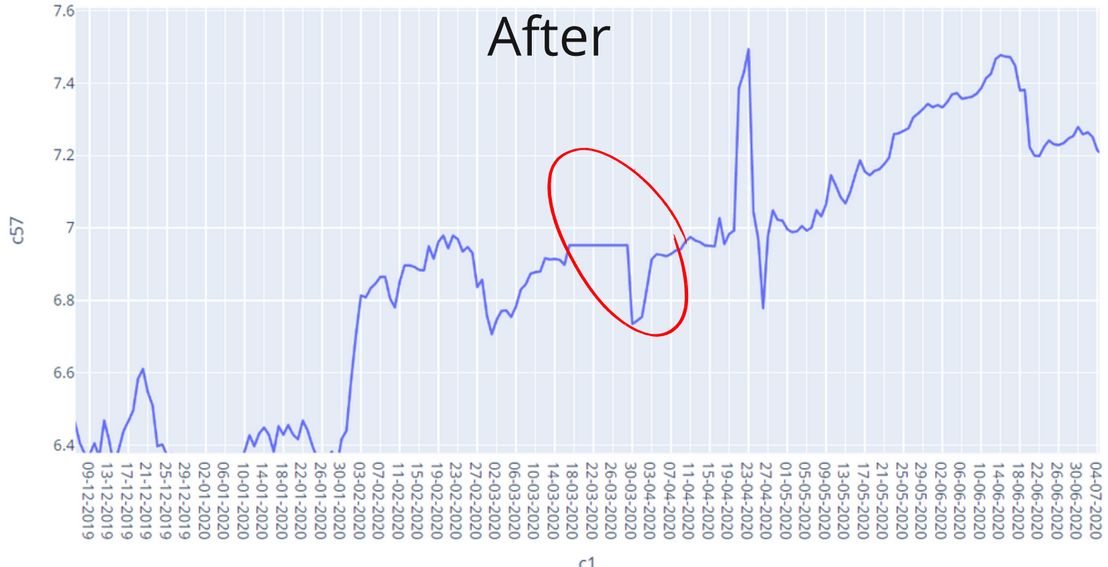
Strategy Used for Replacing NaN Values:

- Replacing all the Nan values with their **corresponding columns median.**

Reason For choosing Column Median:

- We choose median as it is **least affected by outliers** in the data than mean and maintains overall distribution of the dataset

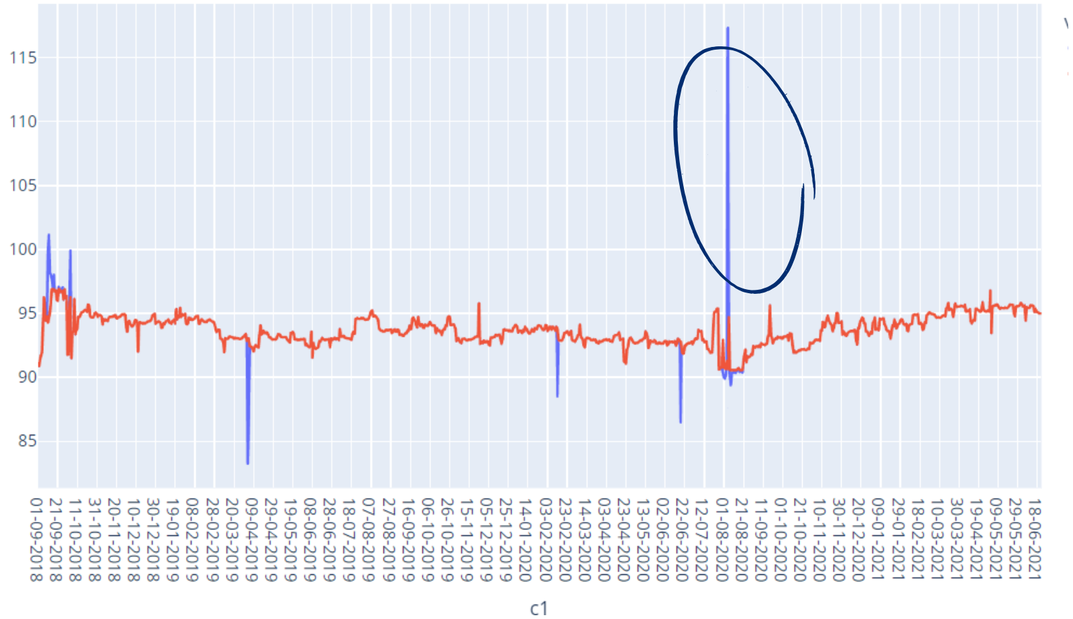




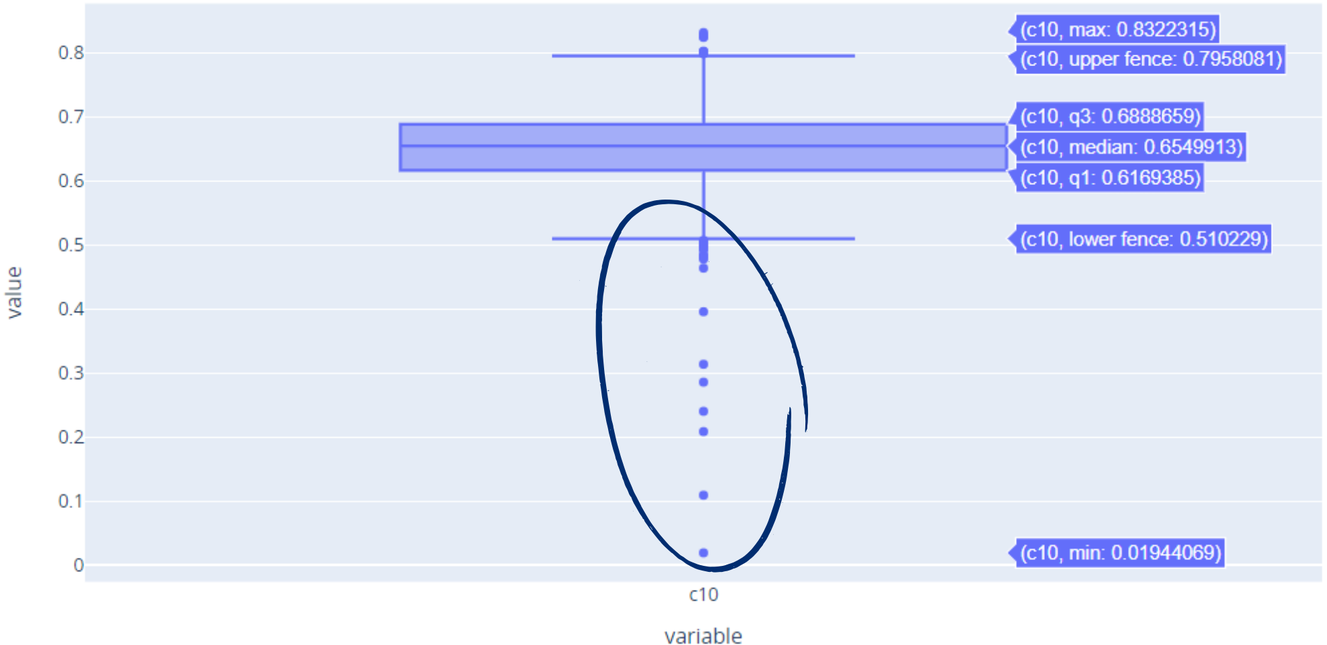
Outlier handling

- Handling Outliers:
- First we find out the outliers using the interquartile range.
- Values lying outside the IQR are replaced by rolling median considering window of size 10.

Outliers Through Line Plot

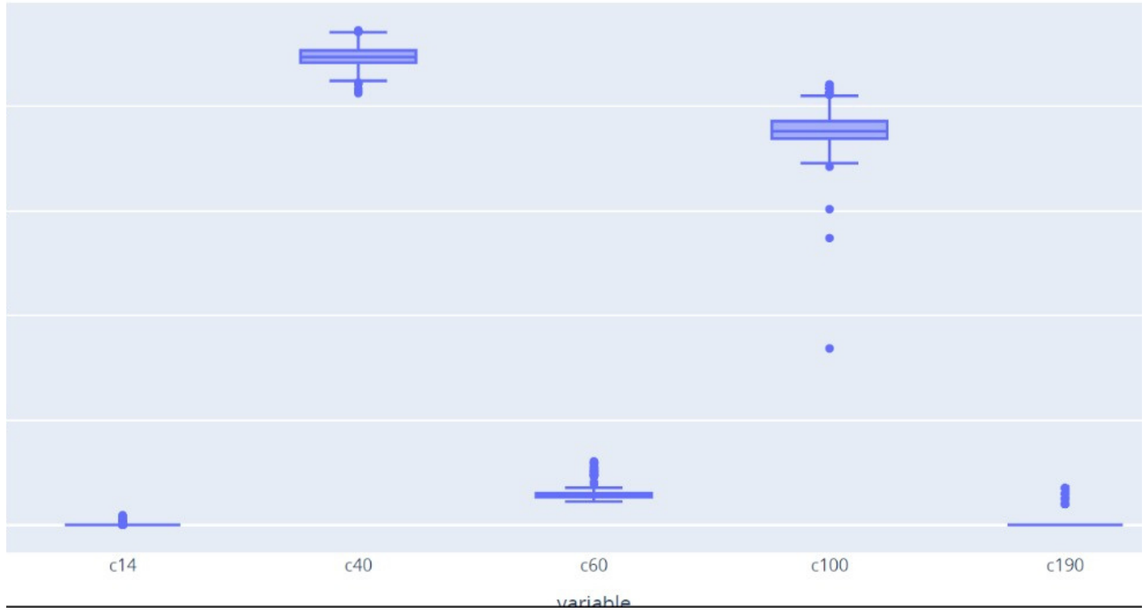


Outliers through Box Plot

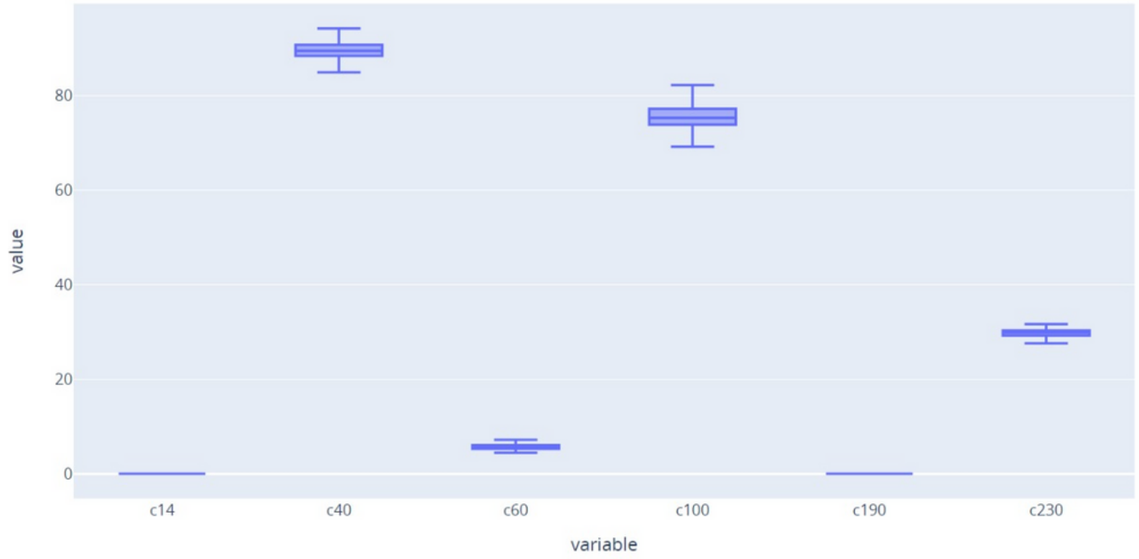


Outliers Handing

- Handling Outliers:



Before



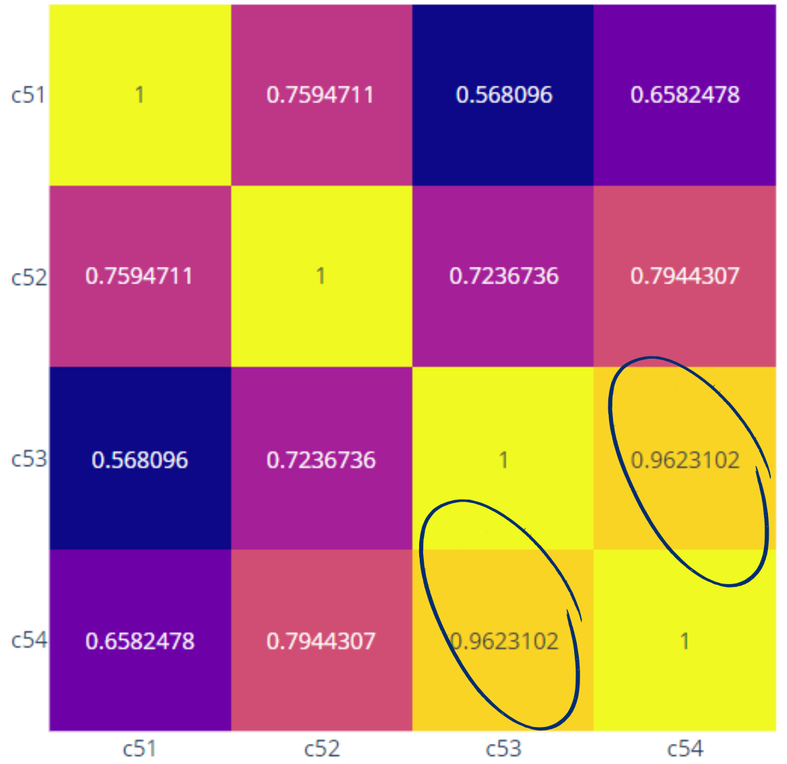
After

2. ML Models for Prediction

ML model for vibrations.

- In this model we have made classes:
  - 1-> Safe
  - 2-> moderate
  - 3-> high
  - 4-> Critical
- From the heat map we can see that c53 and c54 are highly correlated nad are linearly dependent.

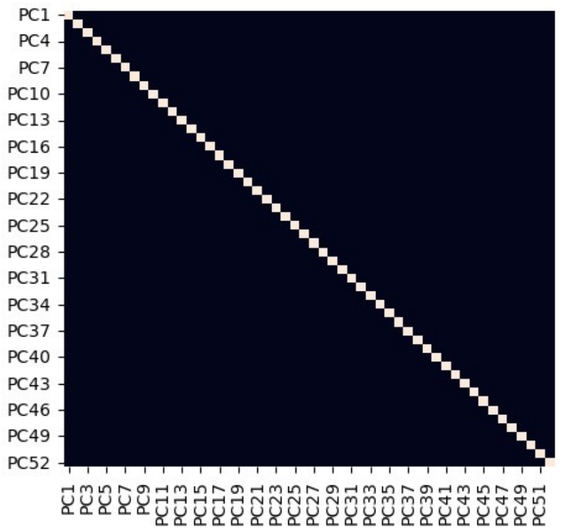
- Heat map of all 4 critical Vibrational Columns.



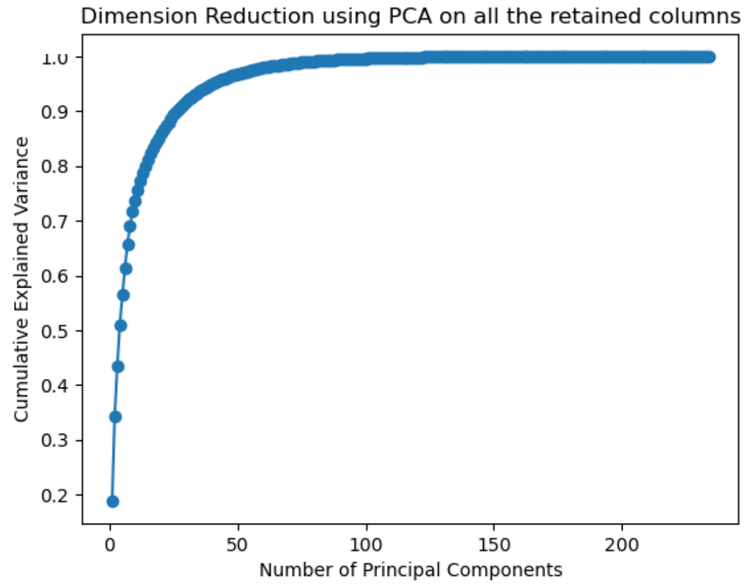
Principal Component Analysis

Intermixing of original columns

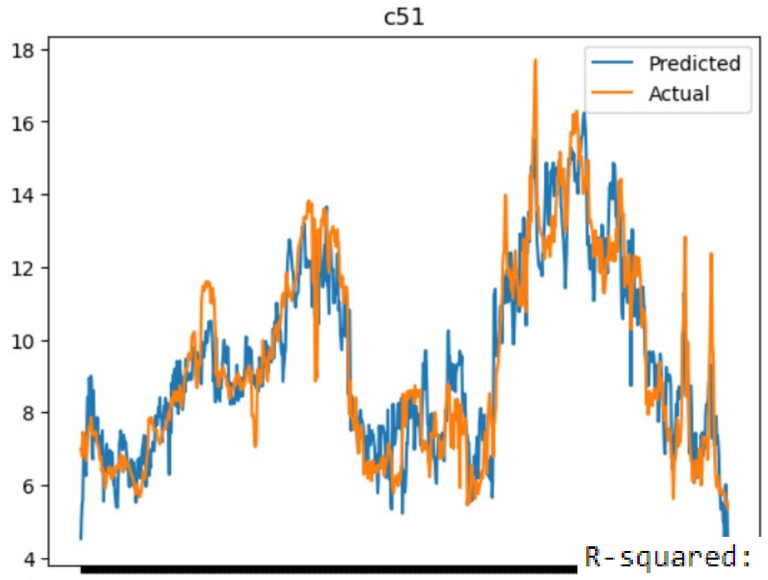
Got another set of independent and lesser columns



Heatmap Showing independence of new columns

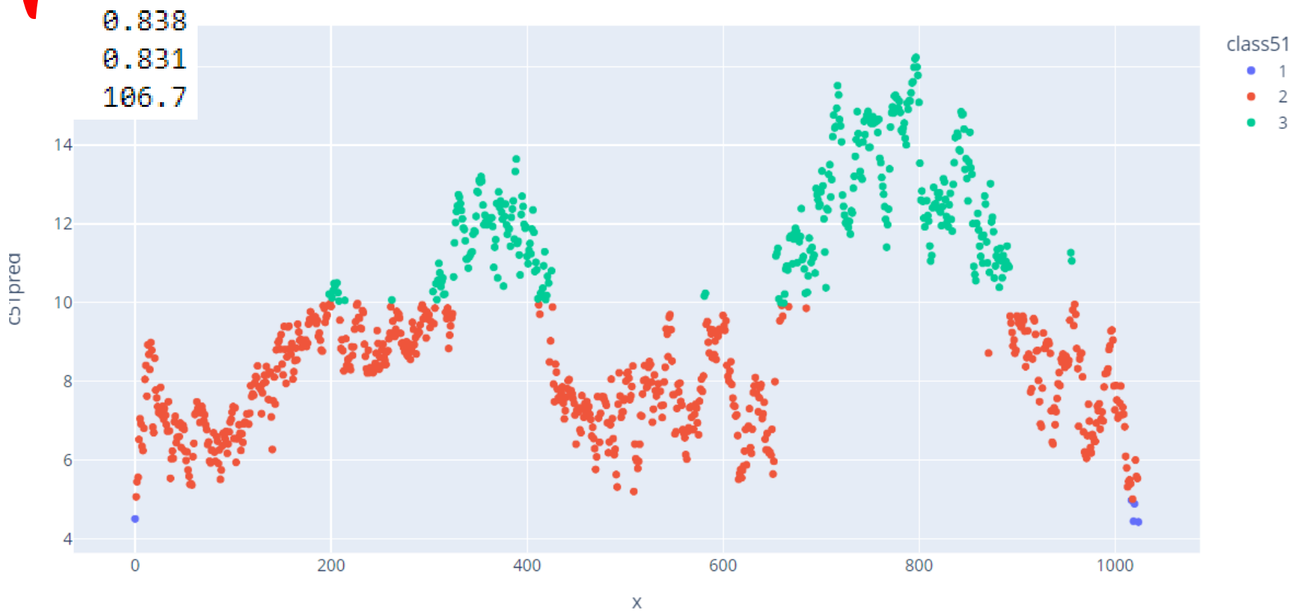


ML model for vibrations.



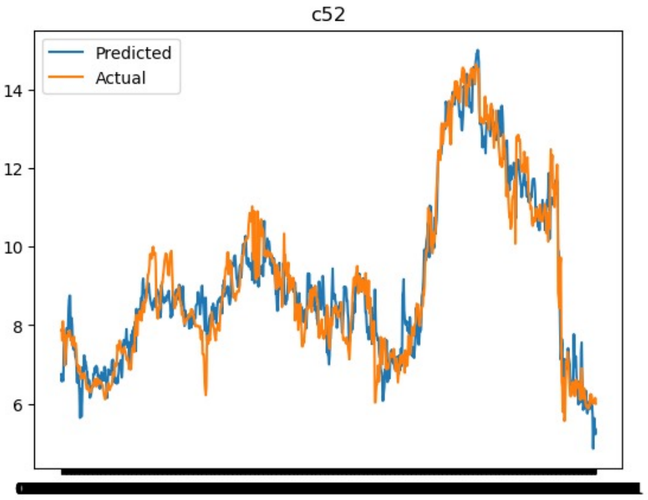
We use OLS library for the prediction of vibrations and divide them into classes.

Our final prediction model for c51 depended on just 38 parameters compared to 240 columns

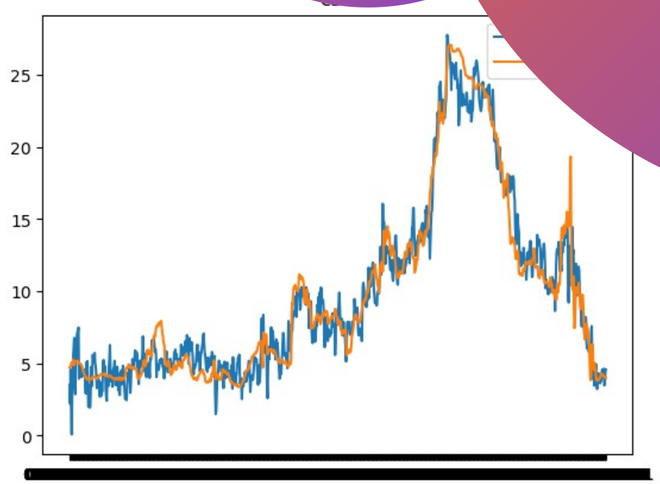


- First we standardise the dataset using the standard scaler library from sklearn.
- The we perform Principle Component Analysis (PCA) using a library to find out the most important columns for the prediction of the model.

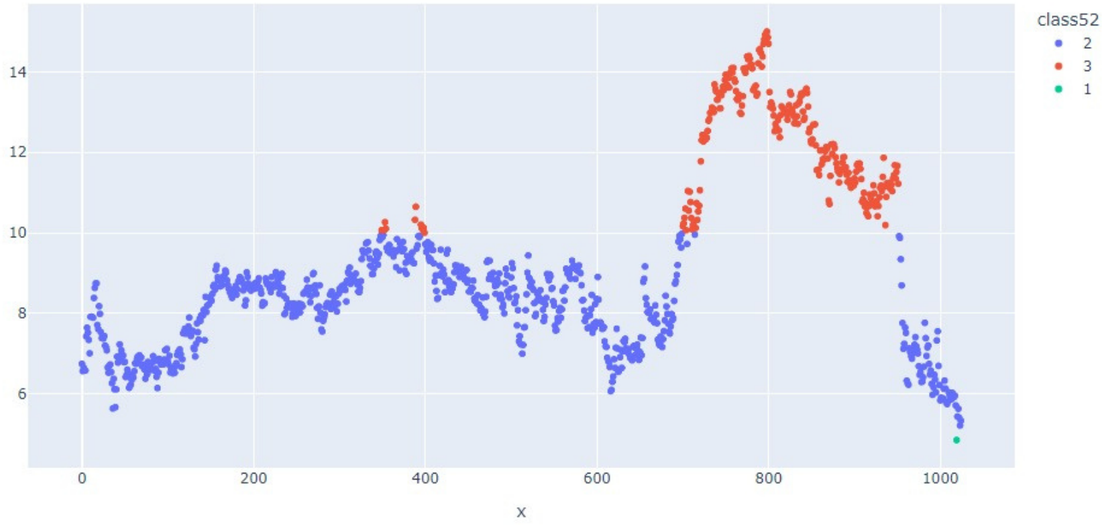
- Similarly we run the model for c52 and c53.

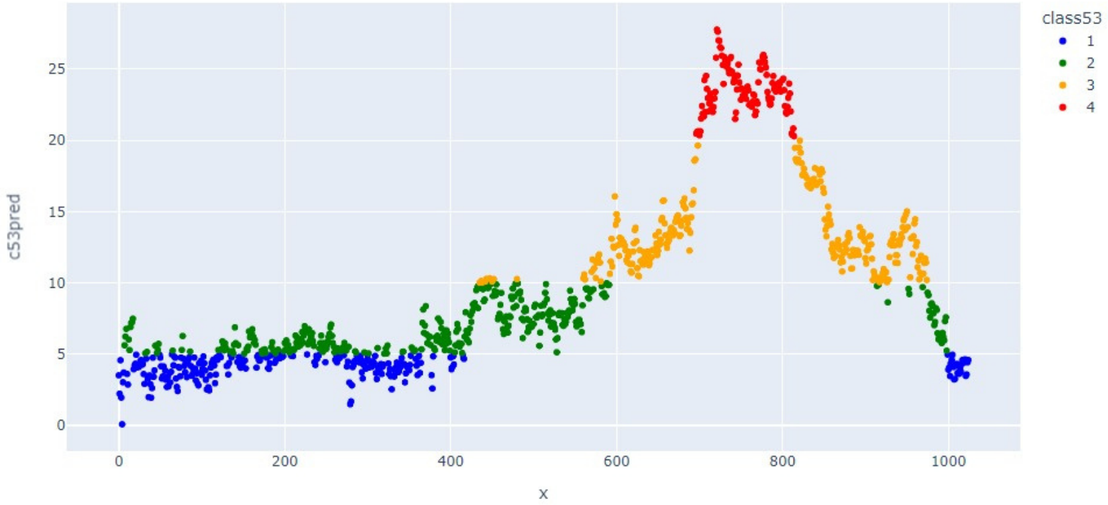



```
----------------------------------------
R-squared:                      0.921
Adj. R-squared:                 0.917
F-statistic:                    233.2
```

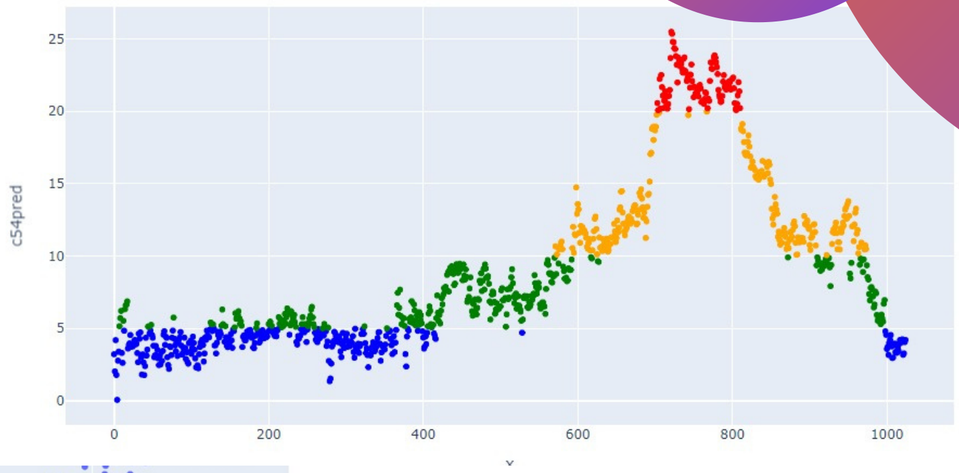


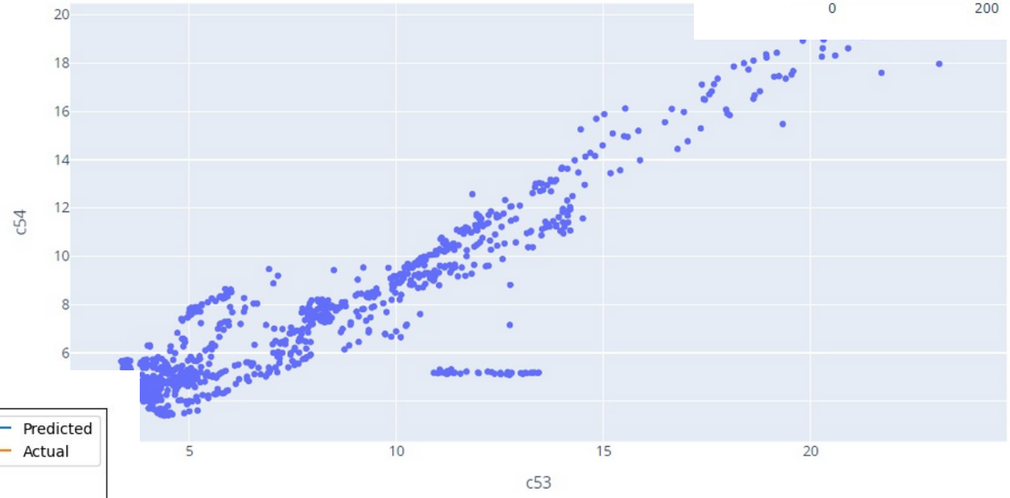

```
R-squared:                      0.954
Adj. R-squared:                 0.951
F-statistic:                    390.7
```

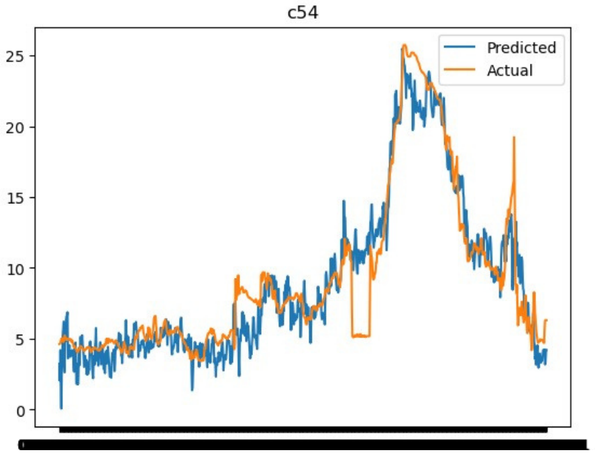




- For c54 we use the predicted c53 as the independent parameter for prediction of c54 , (linearly regressed c53 for c54)





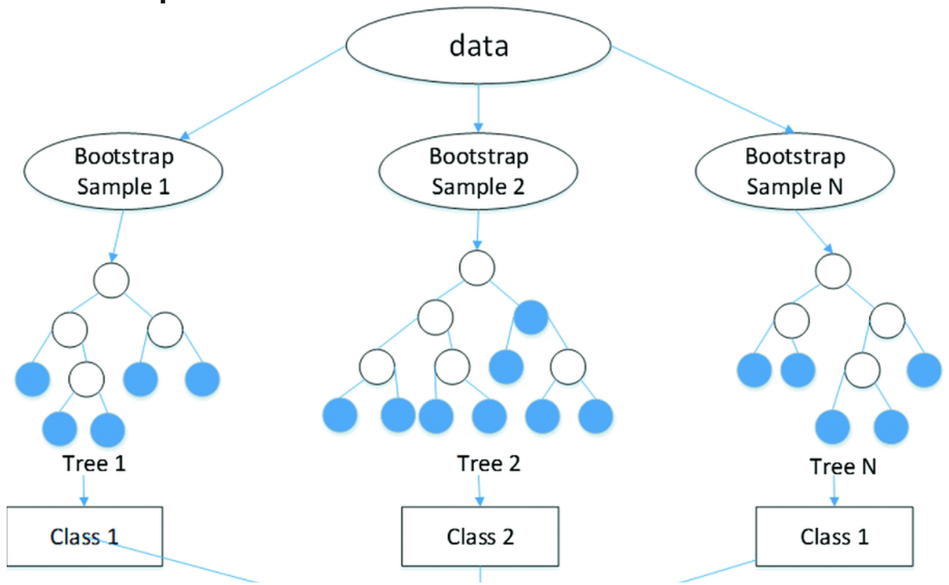


- We observed that the columns c53 and c54 were highly correlated as seen from this plot as well as from the heat map shown earlier.

3. Important Features Selection (RF model)

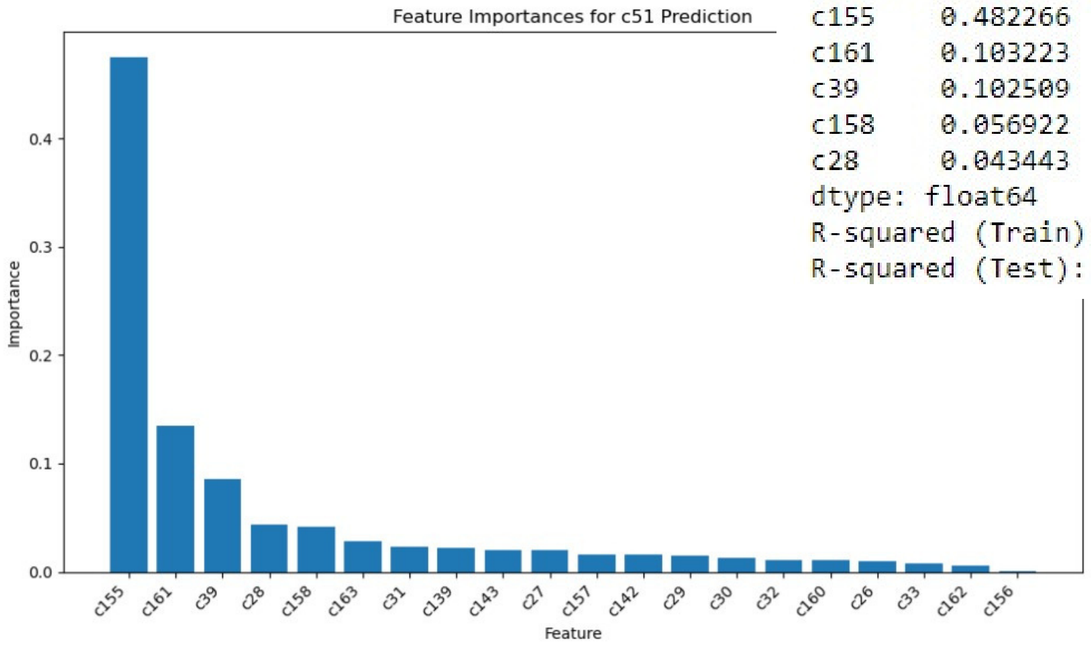
Why To Use Random Forest

- Random Forests are capable of **capturing non-linear relationships** between input features and the target variable.
- Random Forests provide a feature importance measure, which can help you identify the most influential parameters contributing to the dependent feature.
- Random Forests are an **ensemble learning method**, meaning they combine predictions from multiple individual models. This approach often leads to better generalization performance of the models

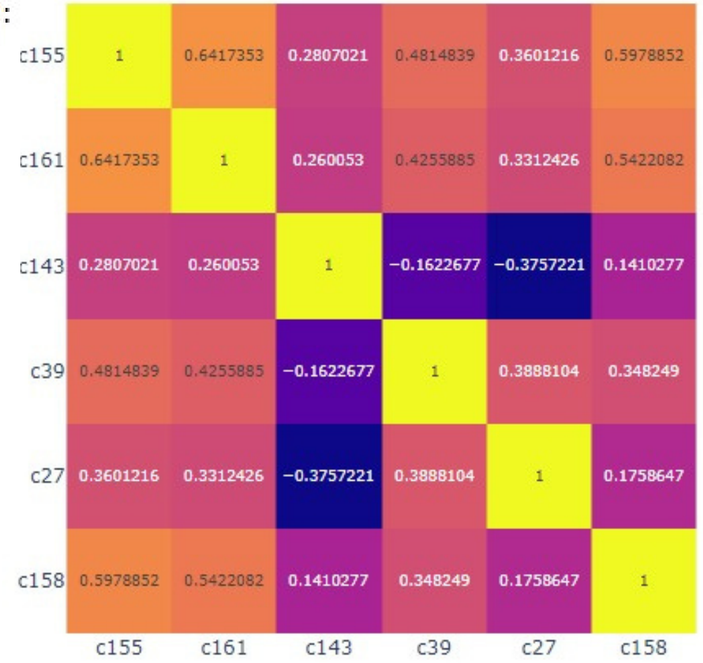


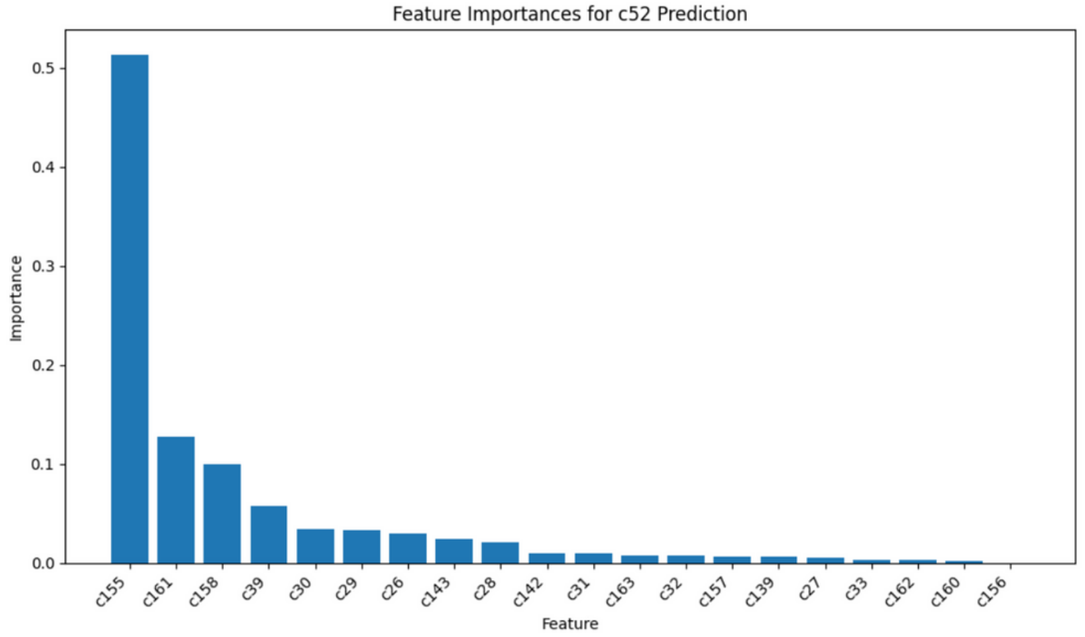
Finding important controllable features.

- We used **Random Forest Regressor** for finding the important features of c51.
- Following is the bar graph showing the importance of the controllable parameters
- The heat map indeed shows the independence of the important features as they have low correlation values.




```
Feature Importance (Descending Order):
c155    0.482266
c161    0.103223
c39     0.102509
c158    0.056922
c28     0.043443
dtype: float64
R-squared (Train): 0.99
R-squared (Test): 0.92
```



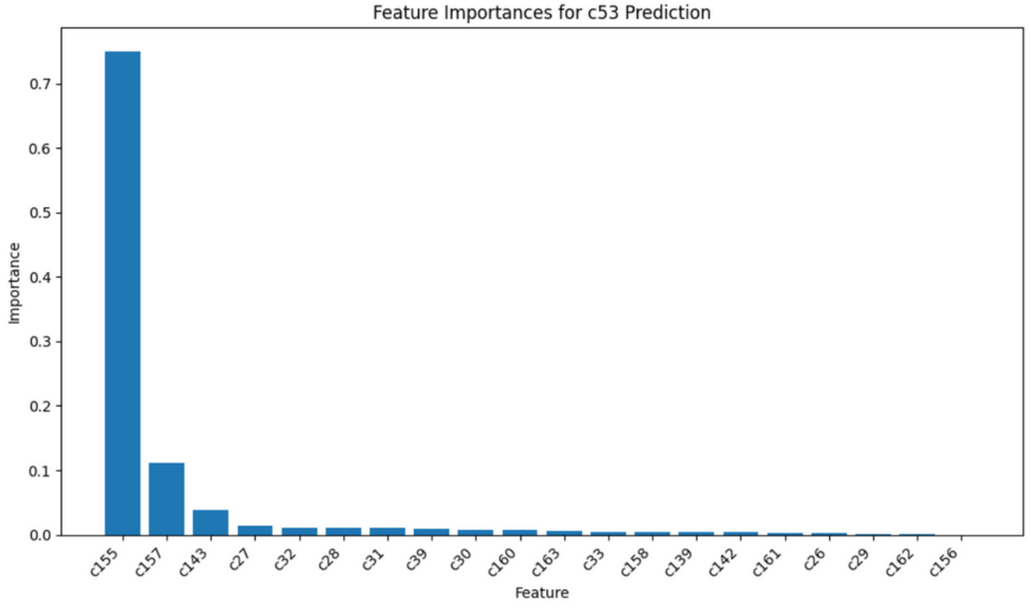


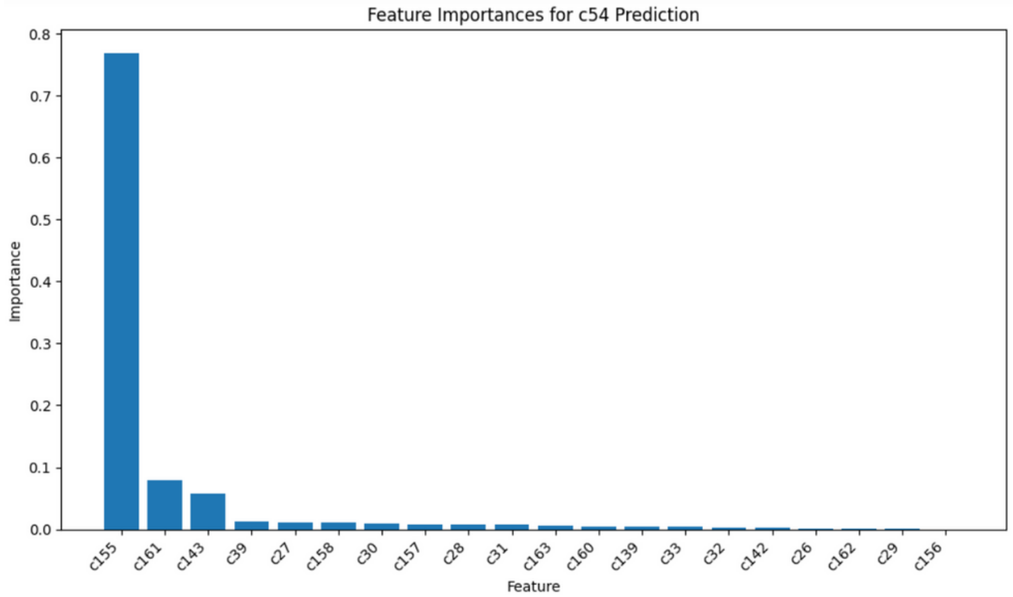

```
Feature Importance (Descending Order)
c155    0.457650
c161    0.161503
c158    0.119824
c39     0.058013
c143    0.034318
dtype: float64
R-squared (Train): 0.99
R-squared (Test): 0.97
```

c52.

c53.

```
Feature Importance (Descending Order):
c155    0.745427
c157    0.124698
c143    0.029893
c31     0.013184
c27     0.011819
dtype: float64
R-squared (Train): 1.00
R-squared (Test): 0.98
```






```
Feature Importance (Descending Order):
c155    0.773661
c161    0.079449
c143    0.055850
c39     0.014335
c27     0.011694
dtype: float64
R-squared (Train): 1.00
R-squared (Test): 0.98
```

c54.

- We can see that c155 is the most important feature for all four critical parameters.
- We came to know that on changing the top 5 parameters , our critical columns can be controlled , we also made and attempt to calculate how much these parameters need to be changed to actually safeguard the process from reaching the critical .

$$y = \beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n + \varepsilon$$

We used the above equation and considered the 5 most important features as parameters in it and created and MLR model out of it .

Differentiating the above equation ,assuming all Xi's to be independent we have

(change in y)/(coefficient of parameter)= (change to be done in parameter)

R-squared:	0.771
Adj. R-squared:	0.770
F-statistic:	857.2

We obtained a good R2 for 'c53' regression, and also the coefficients after dropping the higher p-values

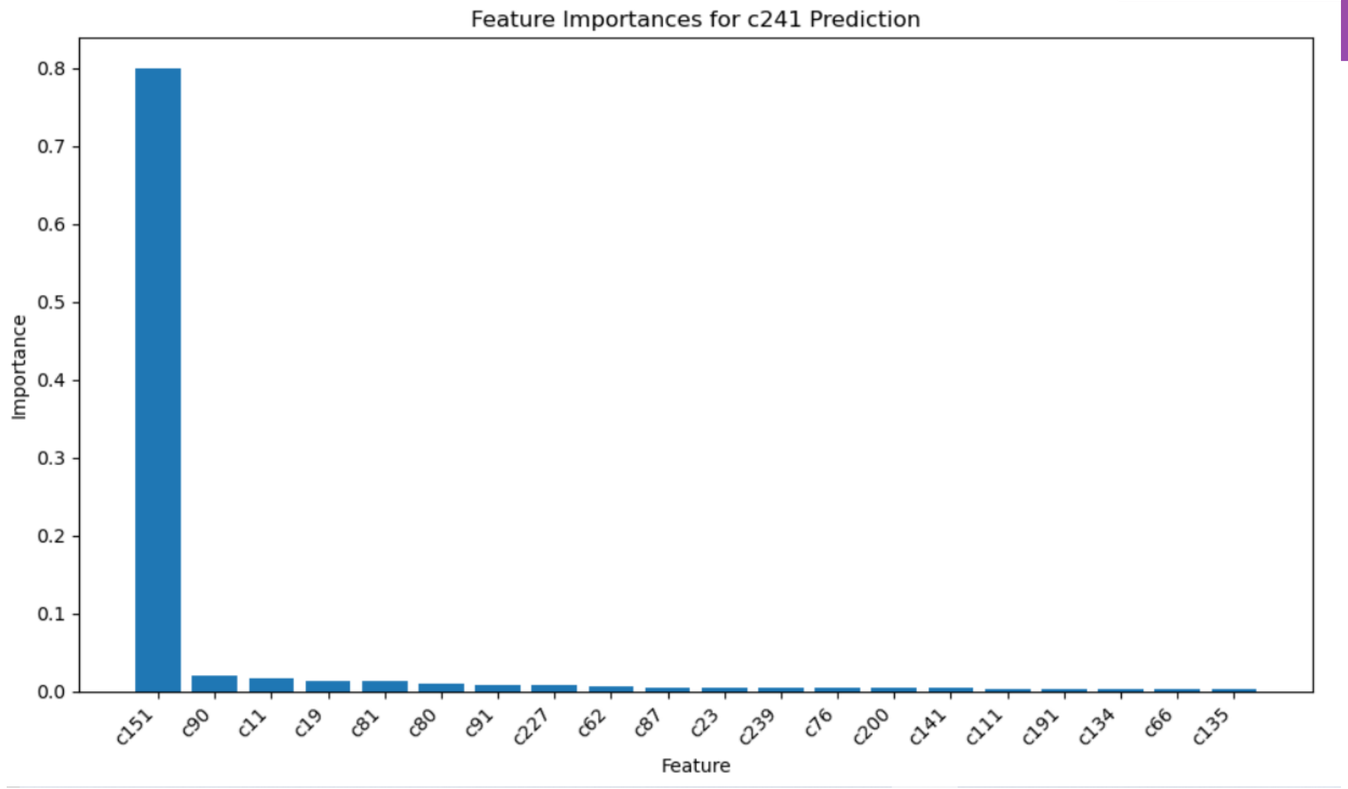
c155	0.670810
c157	-0.145593
c143	0.254542
c31	0.053576

```
10.015074855780838
 High ALERT!!! System Activated
Reduce the parameters
              0
c155   0.022473
c157  -0.103541
c143   0.059224
c31    0.281371
10.131191174623359
```

Finding important parameters for specific energy.

Feature Importances:

	Feature	Importance
134	c151	0.798811
77	c90	0.019817
8	c11	0.017025
16	c19	0.013610
69	c81	0.012728
68	c80	0.010425
78	c91	0.008089
194	c227	0.007187
50	c62	0.005662
74	c87	0.004779



From the above graphs we can see that Out of Controllable and Operating features, the Parameter which contributes most significantly to ‘specific energy’ is **c151**

Finding important parameters for specific energy.

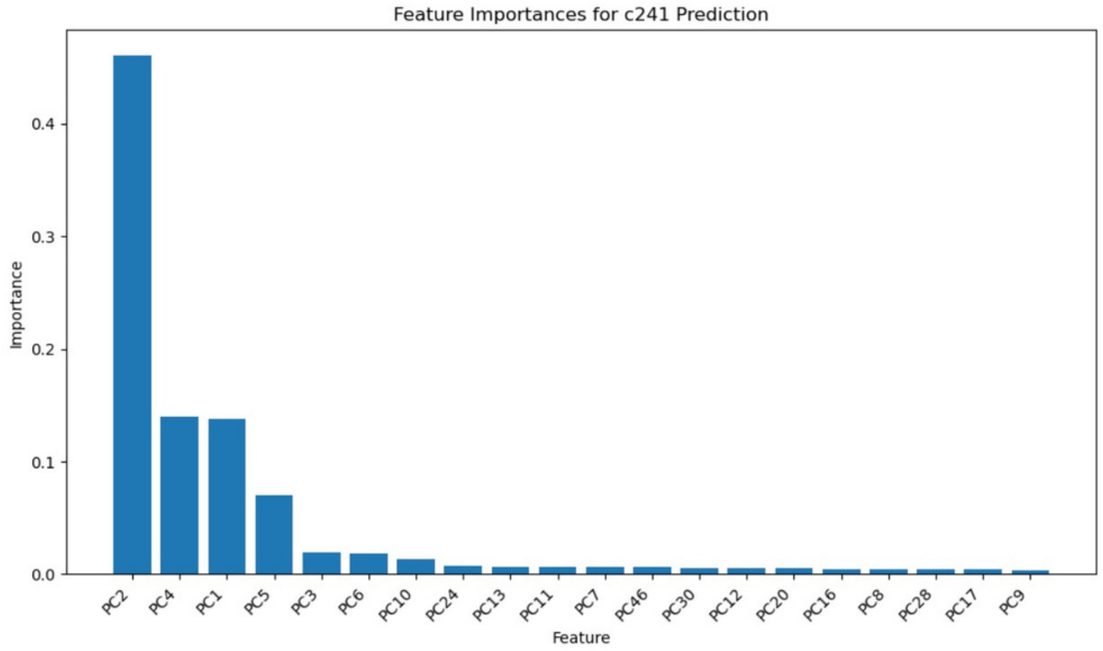
- This Heatmap shows the correlation between the Specific Energy and the other features
- we can see the highest correlation (0.957) with c151 followed by c90 (0.63)

Finding the independent variables for prediction.

- We use PCA for finding the minimum number of the independent features that can be used to predict the specific energy consumption. We got 55 such columns, but on analysing we found 7 such independent columns which can influence the specific energy.
- And then use Random Forest regressor to predict the specific energy consumption using that parameters.

Independent Feature Importances:

	Feature	Importance
1	PC2	0.460333
3	PC4	0.139753
0	PC1	0.138090
4	PC5	0.070350
2	PC3	0.019581
5	PC6	0.018275
9	PC10	0.013189

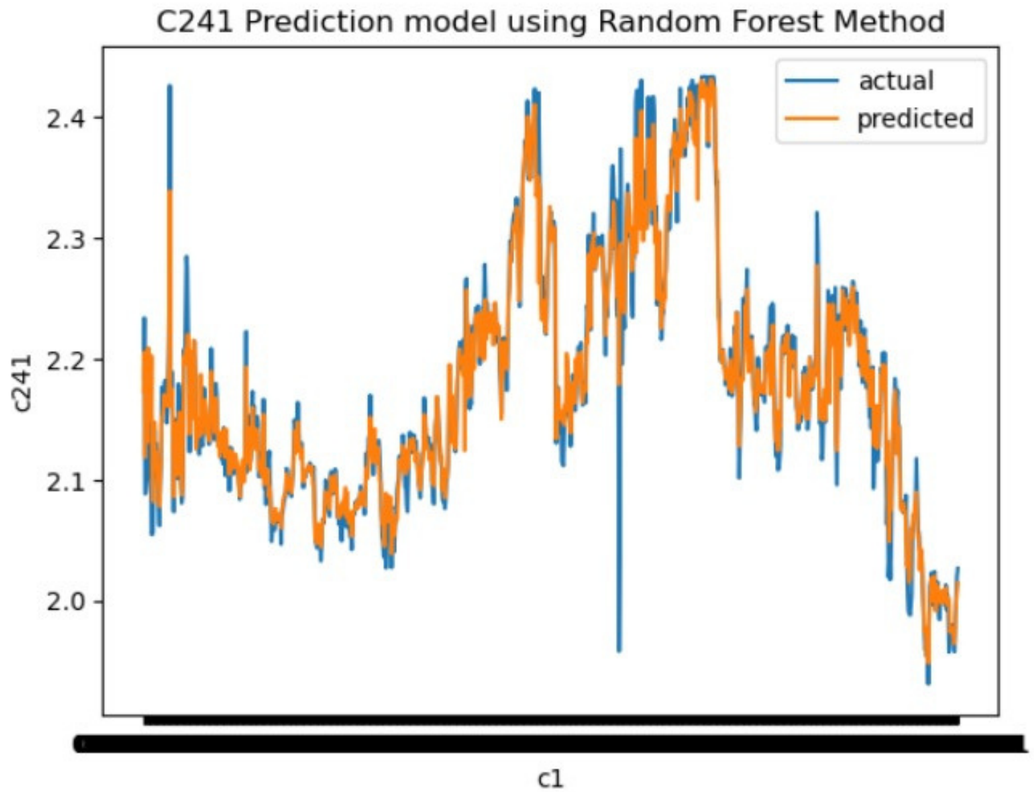


Finding the independent variables for prediction.

- Following is the output from the ML model used for the prediction of c241 feature
- We got R-squared value to be 0.859 from random forest model which is quite good.

```
R-squared
0.8597034112133396
                c1      c241  c241pred
0      01-09-2018  2.184083  2.173207
1      02-09-2018  2.233879  2.205219
2      03-09-2018  2.088296  2.118854
3      04-09-2018  2.089270  2.164127
4      05-09-2018  2.096676  2.146664
...           ...       ...       ...
1020   18-06-2021  1.957681  1.966358
1021   19-06-2021  1.986429  1.985633
1022   20-06-2021  2.006031  2.000001
1023   21-06-2021  2.020471  2.006775
1024   22-06-2021  2.026799  2.014767

[1025 rows x 3 columns]
```



4. Achievments and Challenges

ACHIEVEMENTS

OLS Regression Results

Dep. Variable:	c51	R-squared:	0.838
Model:	OLS	Adj. R-squared:	0.831
Method:	Least Squares	F-statistic:	106.7
Date:	Sun, 12 Nov 2023	Prob (F-statistic):	2.84e-280
Time:	01:12:22	Log-Likelihood:	-1235.2
No. Observations:	820	AIC:	2548.
Df Residuals:	781	BIC:	2732.
Df Model:	38		
Covariance Type:	nonrobust		

```
R-squared (OLS): 0.84
R-squared whole (OLS): 0.84
```

c51.

These high Rsq . values certainly take us to the risk of overfitting , but actually it is not , we got a similar Rsq values from the model when tested on test data for all the for vibrational columns.

- Following are the metrics for the OLS model of c51 and c52 with high R^2 value. We first used PCA and then created an MLR model out of it.

c52.

OLS Regression Results

Dep. Variable:	c52	R-squared:	0.921
Model:	OLS	Adj. R-squared:	0.917
Method:	Least Squares	F-statistic:	233.2
Date:	Sun, 12 Nov 2023	Prob (F-statistic):	0.00
Time:	01:16:16	Log-Likelihood:	-775.87
No. Observations:	820	AIC:	1632.
Df Residuals:	780	BIC:	1820.
Df Model:	39		
Covariance Type:	nonrobust		

ACHIEVEMENTS

- Following are the metrics for the OLS model of c53 and c54 with high R^2 value.

c53.

OLS Regression Results

Dep. Variable:	c53	R-squared:	0.954
Model:	OLS	Adj. R-squared:	0.951
Method:	Least Squares	F-statistic:	390.7
Date:	Sun, 12 Nov 2023	Prob (F-statistic):	0.00
Time:	01:20:38	Log-Likelihood:	-1430.5
No. Observations:	820	AIC:	2945.
Df Residuals:	778	BIC:	3143.
Df Model:	41		
Covariance Type:	nonrobust		

OLS Regression Results

Dep. Variable:	c54	R-squared (uncentered):	0.967
Model:	OLS	Adj. R-squared (uncentered):	0.967
Method:	Least Squares	F-statistic:	3.010e+04
Date:	Sun, 12 Nov 2023	Prob (F-statistic):	0.00
Time:	01:22:41	Log-Likelihood:	-2150.1
No. Observations:	1025	AIC:	4302.
Df Residuals:	1024	BIC:	4307.
Df Model:	1		
Covariance Type:	nonrobust		

c54.

ACHIEVEMENTS

- **Successfully made Automated Vibration Control And Reduction System.**
- To control vibrations we decrease the parameters having positive coefficients and increase having positive coefficients.
- In the following the model automatically detects if the vibrations are in critical or higher and then reduce the parameters to bring in safe region.

```
21.531973161593708
 CRITICAL ALERT!!! System Activated
Reduce the parameters
                0
c155    2.283767
c157  -10.522296
c143    6.018557
c31    28.594168
```

```
10.015074855780838
 High ALERT!!! System Activated
Reduce the parameters
                0
c155   0.022473
c157  -0.103541
c143   0.059224
c31    0.281371
```

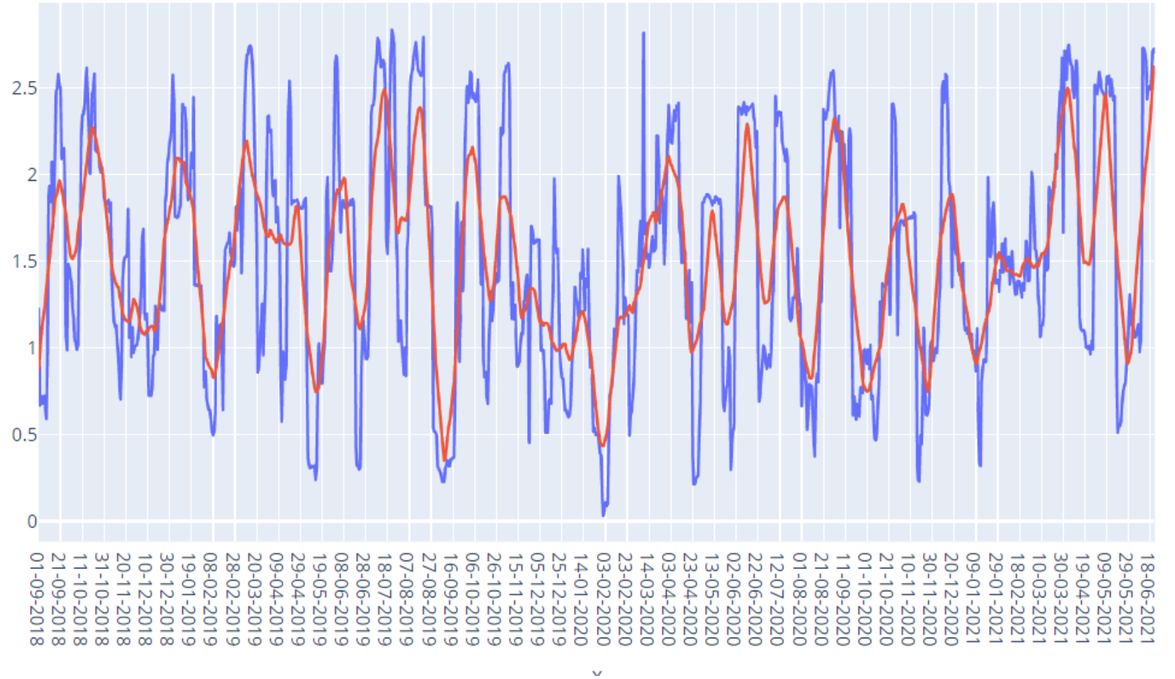
CHALLENGES WE FACED

- Handling the Nan Values were very difficult as it also included finding and replacing some of the text of excel errors.
- The Excel file also consists of some columns which were actually not of type “float” but of “string” ,their line plots were actually linear in nature, for our predictions we have to convert them to float.
- There were a lot of columns which had more than 900 zeroes in a column, thus a lot of data was missing in such columns.
- Also the ML model that we created for reduction of controllable parameters does not gave us a very good Rsq value.

CHALLENGES WE FACED

- Trying to smoothen the curve using rolling mean.

- Some fluctuations in the data are important but it was too smoothened for making the model.



Thank You