Assignment – E11
DS- 203 program

200260018 – Gopal Parwani
22D1628 – Roomani Srivastava
22D1630-Kushal Kelshikar

Data

- Daily averaged values of observations of some parameters at a chemical processing plant
- 240 columns and 1025 rows. Each row corresponds to data for the date given in column c1
- Rest of the columns represent can be grouped into operating and controllable parameters
- Controllable parameters: c26, c27, c28, c29, c30, c31, c32, c33, c39, c139, c142, c143, c155, c156, c157, c158, c160, c161, c162, c163

Objectives

- c 51, c52, c53, c54 represent vibrations: need to be kept in limit
- c241 is specific energy or i.e. energy consumption per unit output: should be as minimum as possible
- All of them are non-controllable parameters i.e. can't control them directly
- Have to control them using controllable parameters

Our Job

Task 1: Reducing the vibrations

- Task 1.1 :
  - Create an ML model with operating + controllable parameters to predict HIGH and CRITICAL vibrations (c51, c52, c53, c54)
  - This model can be used to raise alerts and alarms
- Task 1.2:
  - Create an ML model with controllable parameters for c51, c52, c53, c54
  - From the model create a list of most important parameters to reduce vibrations

Task 2: Reducing the specific energy

- Task 2.1 :
  - Creating an ML model with operating + controllable parameters to predict specific energy, c241
  - From the model, create a list of most important parameters to reduce the energy consumption
- Task 2.2:
  - Find the minimum number of independent variables which can predict the energy consumption
  - Create an ML model with those variables

Data Pre-processing

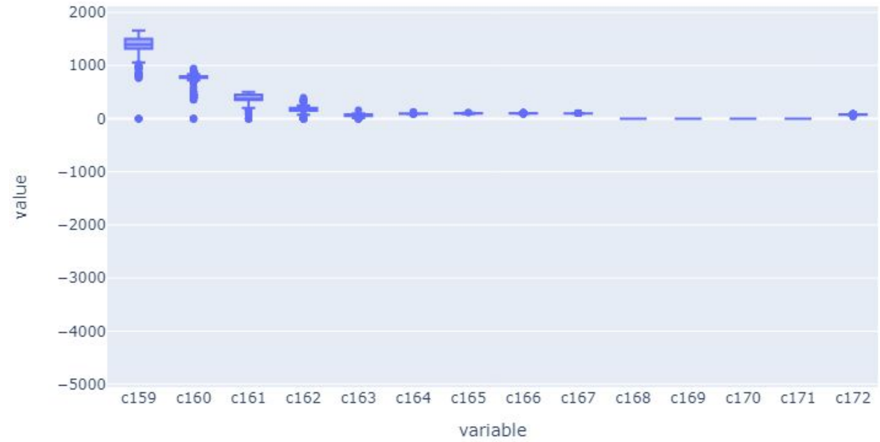
While eliminating features, we take care not to drop the vibration, specific energy & controllable parameter

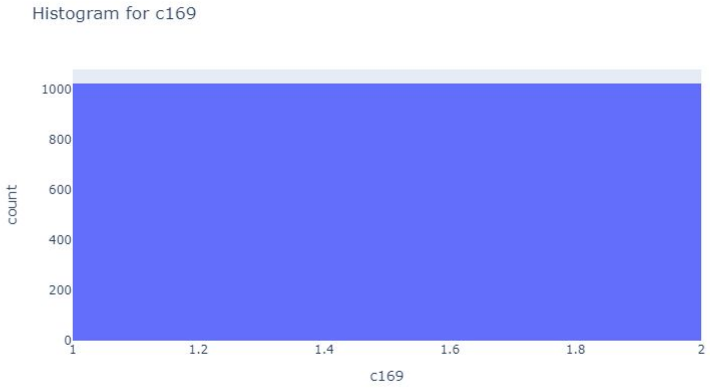
Eliminating Missing Features	Eliminating Const. Features	Imputing Outliers & Missing Values	Reducing Multicollinearity
Visualization Converted the data into dataframe df1 and dropped the columns which have data missing on > 50% of the days by visualization. We also drop the column c1 which is date.	**Box Plot and Histograms** Using plotly, created the box plot and found the variables using for which the spread was very less. If the variable was constant examined by count vs value histogram, dropped from df1	**Mean** Imputed the outliers and missing values in the columns of df1 using the mean of the available values	**VIF** Created a dataframe df2 after dropping the variables which had Variation Inflation Factor, VIF > 10 without changing df1 which has features before reducing multicollinearity

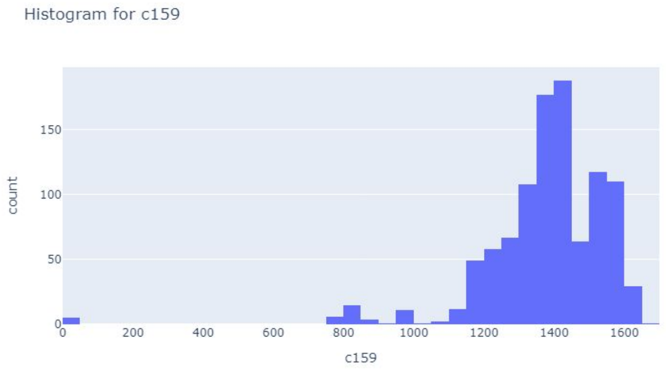
KCDH KOITA CENTRE FOR DIGITAL HEALTH

Data Pre-processing: Eliminating Constant Features

- A section of box plots for all variables



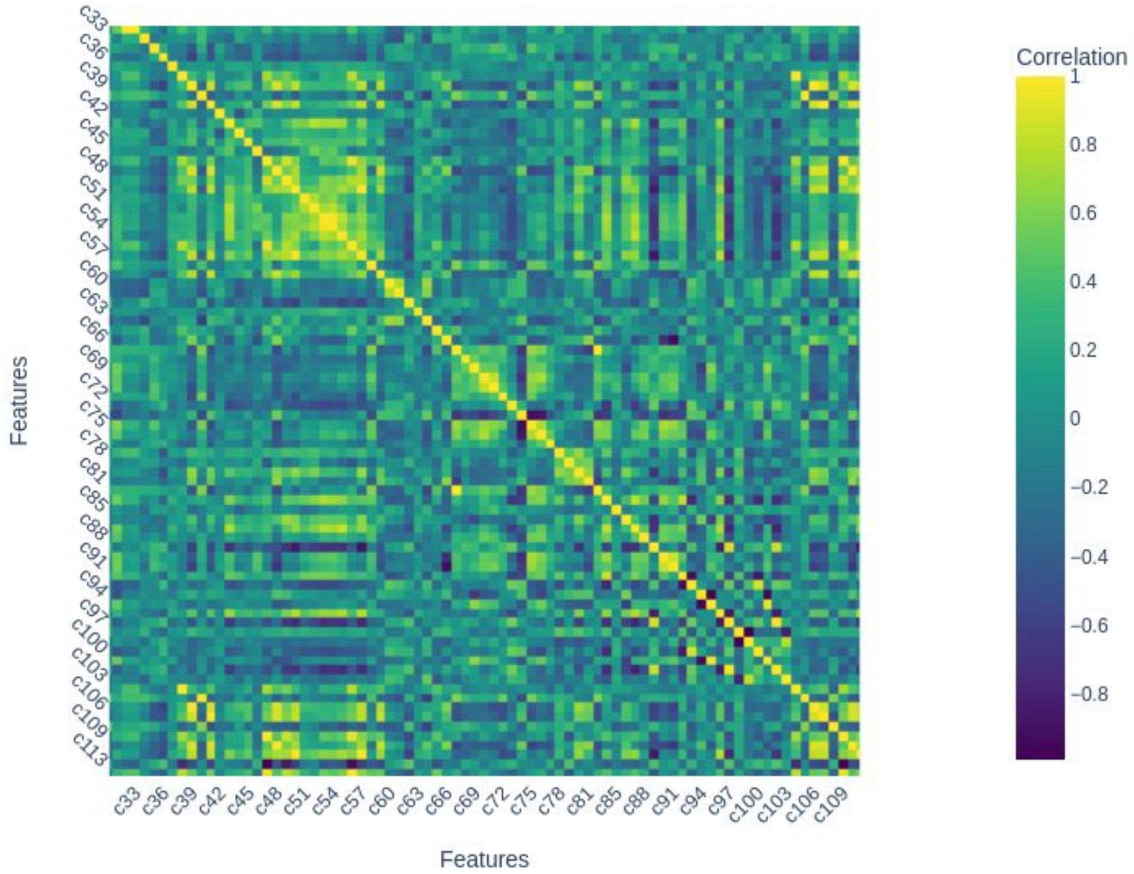




- c168, c169, c170, c171 look constant and on plotting histograms of such variables we find it to be the case and we such variables
- count vs value histogram of non-constant variables like c159 looks like the third figure and we keep such variables

Data Pre-processing: Reducing Multicollinearity

- Visualization using heat maps: In a section of heat map presented below, it can be seen that there is high collinearity between some variables as indicated by bright yellow and dark green squares, off the diagonal.

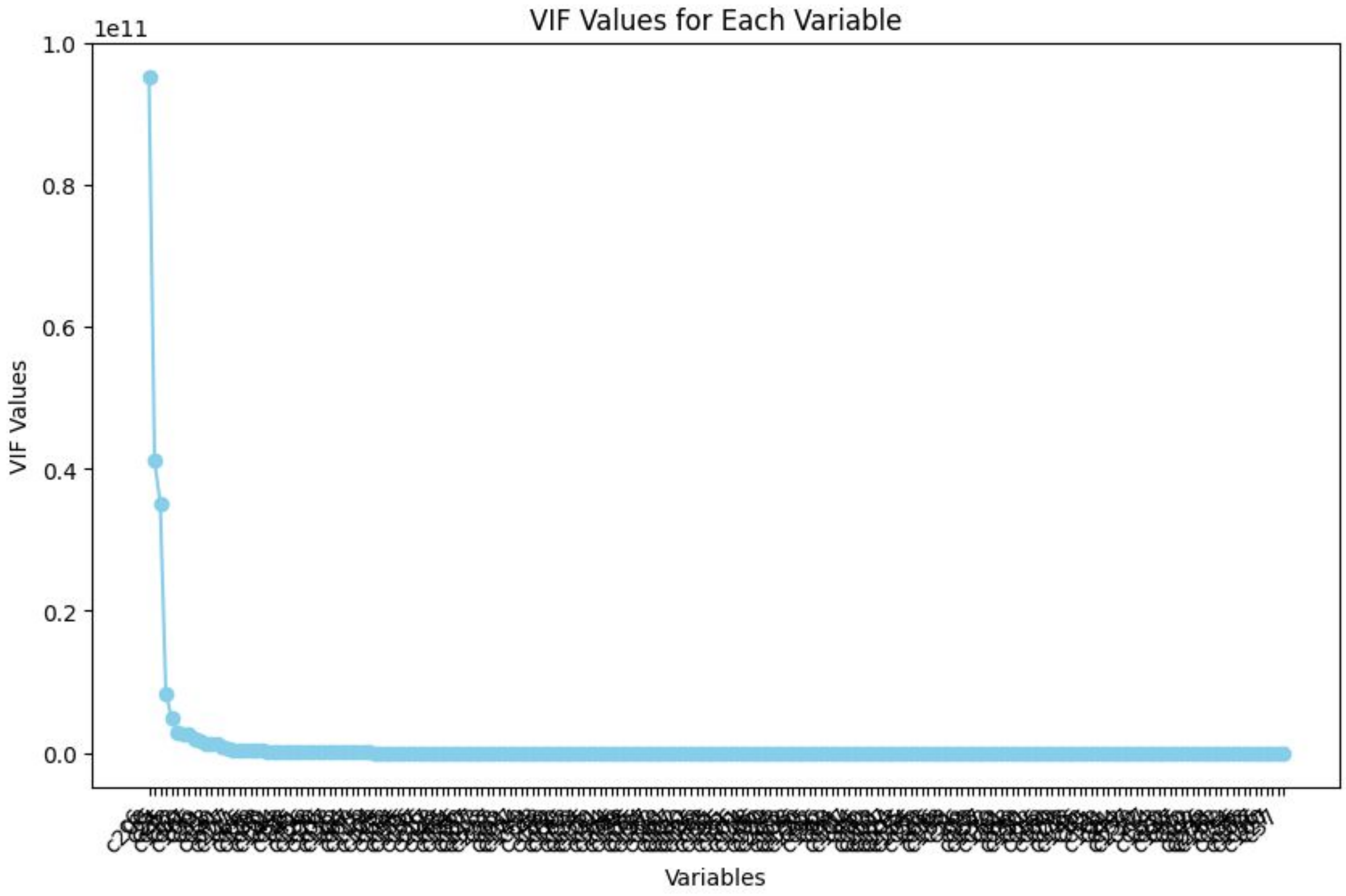


Data Pre-processing: Reducing Multicollinearity using VIF

VIF	Variance inflation factor of all variables were checked
Determining High VIF features	Features with VIF higher than 10 identified
Deleting High features	High VIF features deleted after checking if they were either ‘controllable parameters’ or ‘target variables’ in which case they were retained

Now we create a new df, df2 which has 53 columns after dropping hgh VIF features as opposed to df1 which has 202 columns after dropping redundant variables

Data Pre-processing: Plot showing VIF



Task 1: Reducing the vibrations

Task 1.1: Implementing ML to predict High and Critical Vibration Levels

- Creating model using df1 which contains variables with multicollinearity as well

Feature Transformation

We created a new target variable ‘overall risk’ using columns c51, c52, c53 & c54.
‘overall_risk’ takes values CRITICAL, HIGH, MODERATE, SAFE if any of the values from c51, c52, c53, c54 satisfy the criteria with ‘CRITICAL>HIGH>MODERATE>SAFE.

Label Encoding

We label encode overall_risk i.e.
CRITICAL = 0
HIGH = 1
MODERATE = 2
SAFE = 3
Interestingly, we don’t find any SAFE
Our input features are features in df1 other than c51, c52, c53 & c54 and target is overall_risk

Splitting Data & Normalization

We split the data into train and test with 80:20 ratio and then normalize them separately using min-max scaling. It is necessary to normalize the data after splitting so as to ensure the independence of training and test data.

Creating ML Model

This is a classification problem so we used **Logistic Regression model** as our baseline model which we trained on both dfs with high VIF features and without. We did not see much difference in the outputs. We also used a more sophisticated ML model which is SVM on df1 as models such as SVM automatically take care of multicolinearity

Task 1.1: Implementing ML to predict High and Critical Vibration Levels - Logistic Regression on unfiltered df1

```
Macro Average Recall: 0.9067537860283341
Confusion Matrix:
[[21  3  0]
 [ 2 82  5]
 [ 0  7 85]]
Classification Report:
              precision    recall  f1-score   support

    CRITICAL       0.91      0.88      0.89        24
        HIGH       0.89      0.92      0.91        89
    MODERATE       0.94      0.92      0.93        92

    accuracy                           0.92       205
   macro avg       0.92      0.91      0.91       205
weighted avg       0.92      0.92      0.92       205
```

Task 1.1: Implementing ML to predict High and Critical Vibration Levels
Logistic Regression on filtered data that is without features with High VIF

```
Macro Average Recall: 0.9341651739673233
Confusion Matrix:
[[23  1  0]
 [ 0 79 10]
 [ 0  4 88]]
Classification Report:
              precision    recall  f1-score   support

    CRITICAL       1.00      0.96      0.98        24
        HIGH       0.94      0.89      0.91        89
    MODERATE       0.90      0.96      0.93        92

    accuracy                           0.93       205
   macro avg       0.95      0.93      0.94       205
weighted avg       0.93      0.93      0.93       205
```

Alternative ML Model: SVM

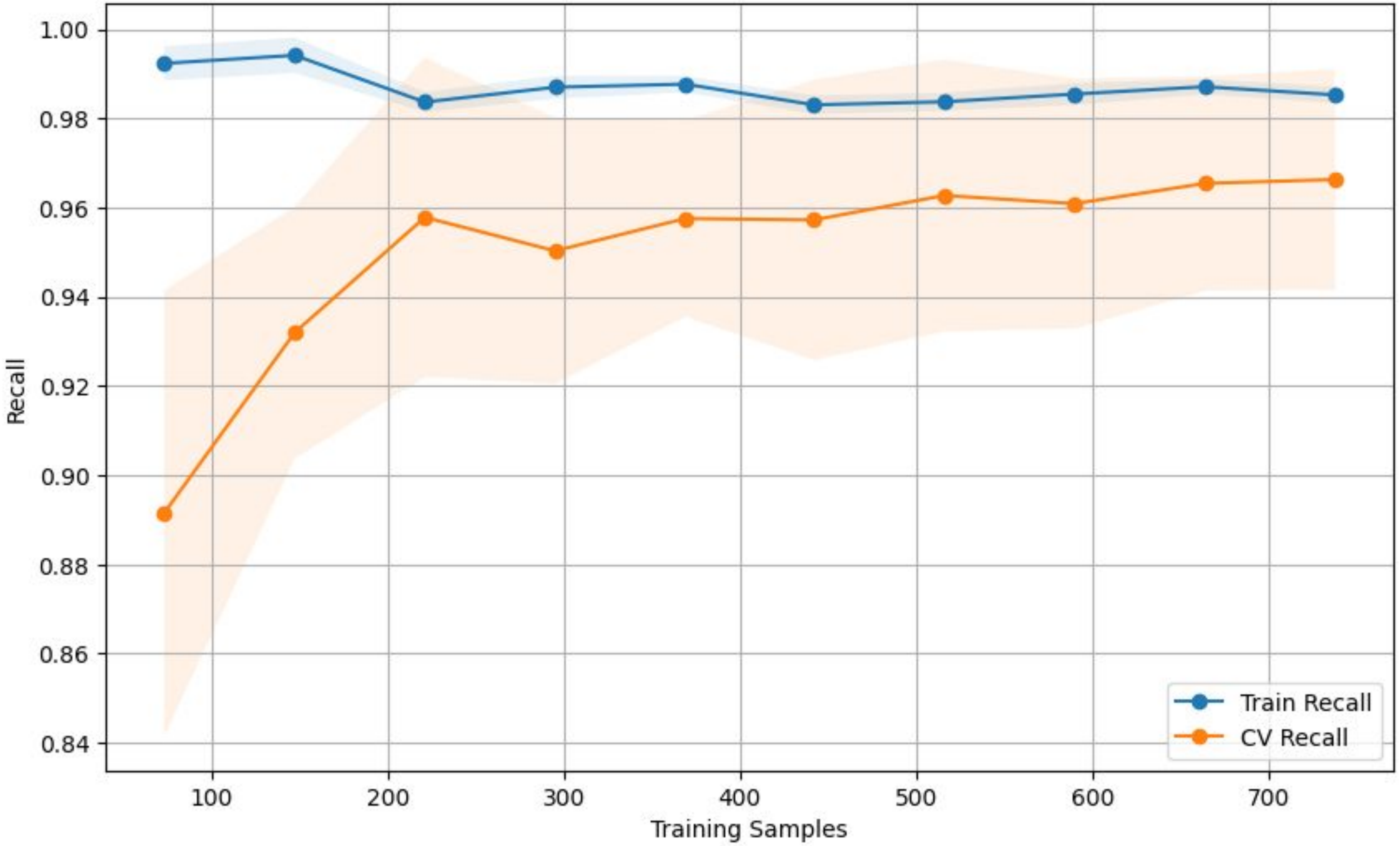
There are some machine learning models which take care of the multicolinearity of the features by using certain methods like regularization and can give good results with the right hyper parameter tuning even without using tools like VIF.

We demonstrate the use of one such ML model which Support Vector Classifier

Results of SVM

```
Best Parameters: {'C': 0.1, 'gamma': 'scale', 'kernel': 'poly'}
Best Score: 0.9667476422080437
Best Parameters: {'C': 0.1, 'gamma': 'scale', 'kernel': 'poly'}
Best Recall: 0.9667476422080437
Test Macro Average Recall SVM: 0.953013895673886


Confusion Matrix SVM:
[[23  1  0]
 [ 0 85  4]
 [ 0  5 87]]
Classification Report SVM:
              precision    recall  f1-score   support

    CRITICAL       1.00      0.96      0.98        24
        HIGH       0.93      0.96      0.94        89
    MODERATE       0.96      0.95      0.95        92

    accuracy                           0.95       205
   macro avg       0.96      0.95      0.96       205
weighted avg       0.95      0.95      0.95       205
```

Learning Curves of SVM



KOITA CENTRE FOR DIGITAL HEALTH

Task 1.1: Implementing ML to predict High and Critical Vibration Levels

As we can see SVM gave us better results, this could be due to the inherent features of sophisticated ML models such as SVM which take care of factors like multicolinearity.

ML Model for Controllable Parameters only

Task 1.2: Implementing ML to find Important Controllable Parameters

- We take the controllable parameters as x variables and 'overall_risk' as the y variable
- We then split the x variables into train and test data 80:20 and normalize using min-max scaling
- We train our model using Random Forest Classifier which has built-in feature importance function
- On fitting we find our performance of the model as follows:

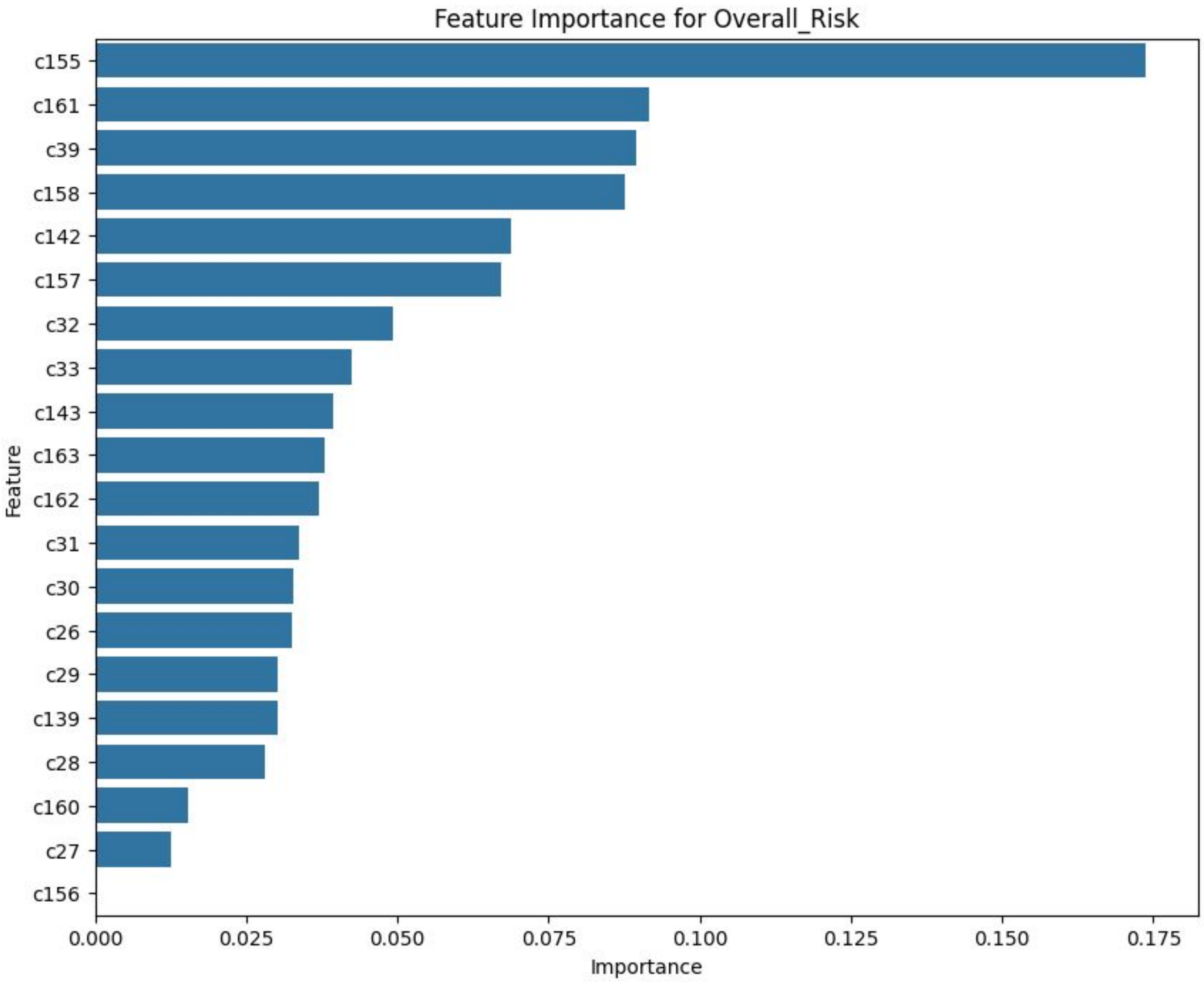
```
Accuracy RF: 0.9463414634146341
Macro Average Recall RF: 0.9596564077511806
Classification Report RF:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        24
           1       0.92      0.96      0.94        89
           2       0.96      0.92      0.94        92

    accuracy                           0.95       205
   macro avg       0.96      0.96      0.96       205
weighted avg       0.95      0.95      0.95       205
```

Task: List Important Features

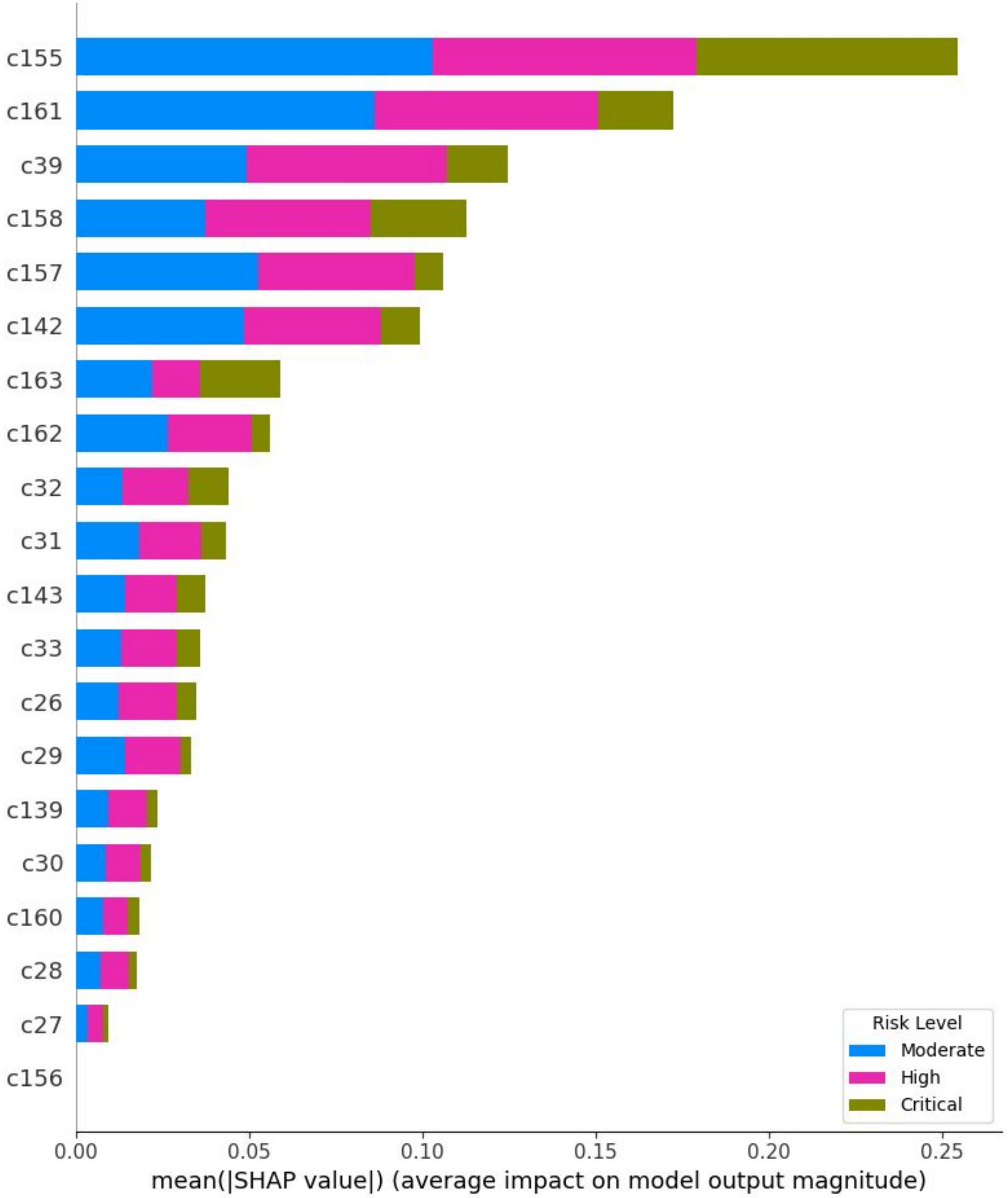
Two Approaches

- Inherent Feature Importance of Random Forest Classifier
- Model Agnostic feature Importance Method Such as SHapeley Additive exPlanations

Feature importance using Random Forest Classifier



Class wise Feature Importance using SHAP



KCDH KOITA CENTRE FOR DIGITAL HEALTH

Task 1.2: Important Features

Note that the important features remain the same more or less, but SHAP gives us added information of class wise feature importance.

KOITA CENTRE FOR DIGITAL HEALTH

Task 2: Reducing Specific Energy

Task 2.1: Implementing ML to Find Significant Features for Specific Energy Prediction

- We create a model using df2 which has no feature with high VIF
- Our target is c241 and the independent variables are all features in df2 other than c241

Splitting Data & Normalization	Addition of Const. Column	Creating ML Model	Important Features
We split the data into train and test with 80:20 ratio and then normalize them separately using min-max scaling. It is necessary to normalize the data after splitting so as to ensure the independence of training and test data.	We add the column of constant which we took as 1, to the features	This is a regression problem and we apply Ordinary Least Square (OLS) available in statsmodel.api It gives coefficients of each variable in it's summary dataframe	Since the features are normalized, the magnitude of the coefficients will be a representative of their importance in reducing the specific energy.

Task 2.1: Implementing ML to Find Significant Features for Specific Energy Prediction

- Truncated OLS results

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   c241   R-squared:                       0.258
Model:                            OLS   Adj. R-squared:                  0.218
Method:                 Least Squares   F-statistic:                     6.494
Date:                Wed, 28 Jan 2026   Prob (F-statistic):           4.70e-36
Time:                        19:18:59   Log-Likelihood:                -981.61
No. Observations:                1025   AIC:                             2069.
Df Residuals:                     972   BIC:                             2331.
Df Model:                          52
Covariance Type:            nonrobust
==============================================================================
```

Task 2.1: Implementing ML to Find Significant Features for Specific Energy Prediction

	Absolute Coefficient
const	4.017975
c155	2.072155
c39	1.365522
c139	1.036667
c26	0.957030
c32	0.913201
c22	0.801206
c143	0.799545
c137	0.681686
c72	0.601606
c158	0.591338
c7	0.558739
c42	0.551763
c53	0.542555
c157	0.516179
c20	0.509314
c21	0.500713
c28	0.401853
c33	0.393902
c73	0.392365
c31	0.372002
c147	0.348229
c29	0.310202
c85	0.305636
c45	0.303773
c133	0.273627
c35	0.259827
c160	0.258501
c12	0.250150
c14	0.205059
c34	0.204746
c239	0.199518
c27	0.197264
c30	0.193622
c142	0.192567
c51	0.157211
c163	0.144724
c54	0.143759
c44	0.125531
c36	0.123357
c11	0.120291
c238	0.109711
c10	0.107097
c176	0.077104
c177	0.071722
c8	0.071115
c161	0.066375
c37	0.044692
c63	0.031363
c162	0.019431
c156	0.017811
c52	0.014177
c146	0.004037

Task 2.2: Determining ‘minimum set’ of ‘independent feature’ utilized to only ‘predict not control’ c241

- Here our approach will be to use PCA with the original data set with all features (even ones with high VIF). The reasoning for this as follows:
  - PCA will help us identify those minimum number of components which explain the maximum variance in the dataset.
  - PCA makes us loose feature importance, so we know the principal components but we don’t know which features contributed to these principal components.
  - However, since our purpose is to only ‘predict’ and ‘not control’ the target, we can afford to loose explainability in lieu of better predictions.
  - Use of original df even with high VIF features is justified and PCA itself takes multicolinearity of the data set into consideration.

Task 2.2: Determining ‘minimum set’ of ‘independent feature’ utilized to only ‘predict not control’ c241

- Our approach:

Splitting Data & Normalization

We split the data into train and test with 80:20 ratio and then normalize them min-max scaling.

Performing PCA

We implement PCA such that number of components are enough to explain at least 90% of the variance of the dataset.

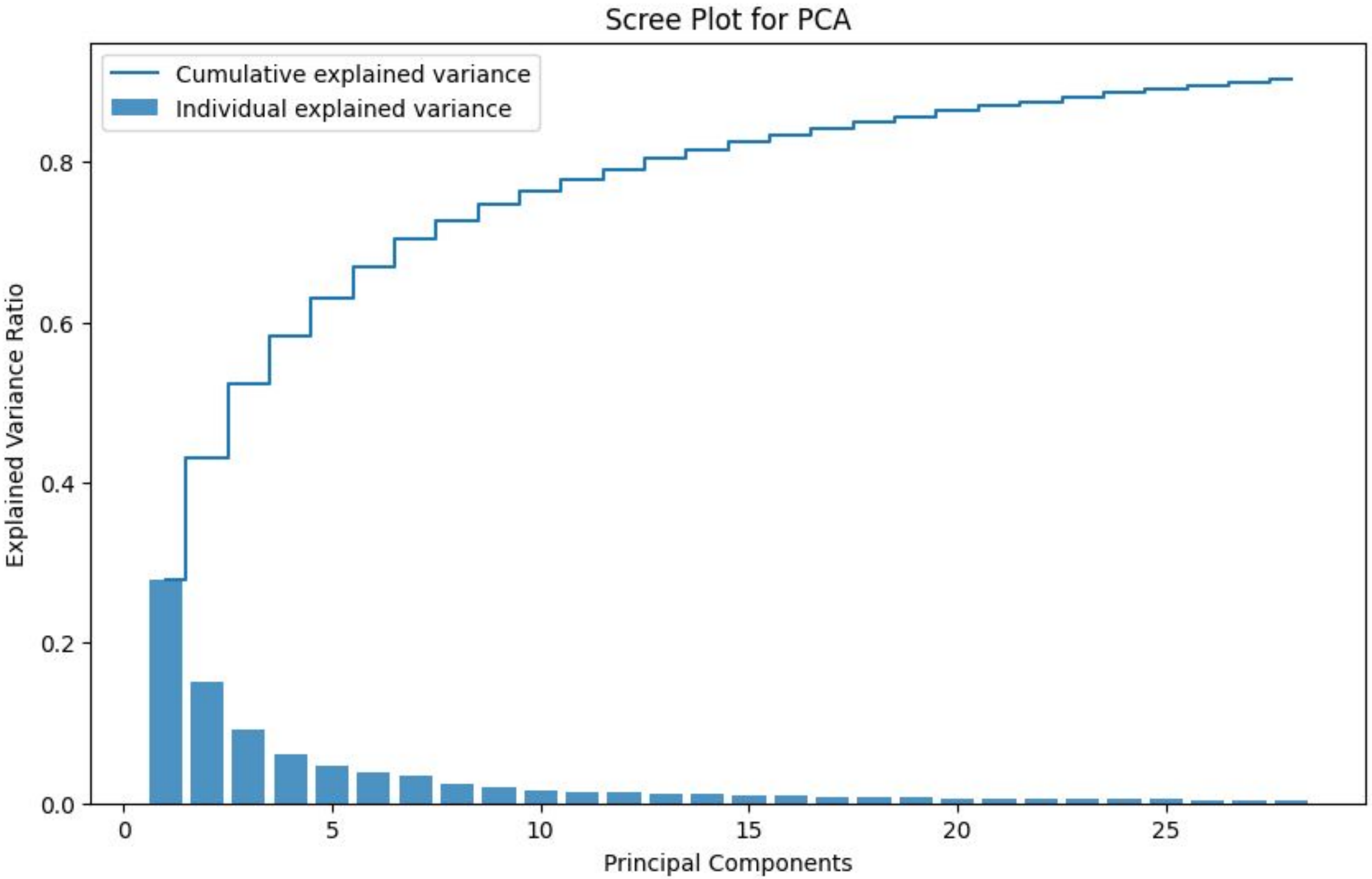
We plot the Scree plot and a scatter plot to visualise this output

Creating ML Model

We use the Principal components in the ML model (we used random forest regressor, we tried others too but rf gave the best results).

We used the functions fit_transform for training set and only transform for test set.

Scree plot for PCA



Results of Random Forest Model using PCA

```
Training MSE PCA RF: 0.08958498931367857
Training R2 PCA RF: 0.8654833745073147

Testing MSE PCA RF: 0.00178802658107922
Testing R2 PCA RF: 0.8590857093117211
```

Key Achievements

- We employed multiple methods of data pre processing for different problems in the data set.
  - Feature binning: For largely missing/ constant variables
  - Data imputation
  - Identifying outliers using IQR
  - Imputing outliers
  - Identifying multicolinearity in the data set and filtering the data set accordingly
- We were successfully able to build machine learning models for the various tasks while taking care of the problems in the dataset.
- For task 1 we created a single ML model and not multiple for c51, c52, c53, c54, by creating a new variable which depended on these four. We could do this as the purpose was to raise an alert if ‘any’ of these were *off.*
- For the feature importance task from among the controllable parameters we used ‘SHAP’ which goes a step ahead and gives us class wise feature importance, thus determining which features contribute heavily to predicting ‘CRITICAL’ level.

KCDH KOITA CENTRE FOR DIGITAL HEALTH

Challenges

- Deciding which machine learning algorithm is the best to use can be a challenge, we had to try multiple ones in our effort and then report the one with best results.
- Data being noisy was a challenge, which we tackled with a series of pre processing steps.

Recommendations

- Machine learning models thus created can be deployed to raise alerts/ alarms when levels reach HIgh/ Critical accordingly.
- Important controllable features identified can be monitored more closely for better outcomes
- In task 2.1 we identified ‘significant features’ responsible for the outcome ‘energy consumption’ these features can be adjusted when a rise in energy consumption is seen
- Finally a simple model created using only principal components can help predict/identify changes in energy consumption with even slight change in input variable.

Thank You