In [1]:

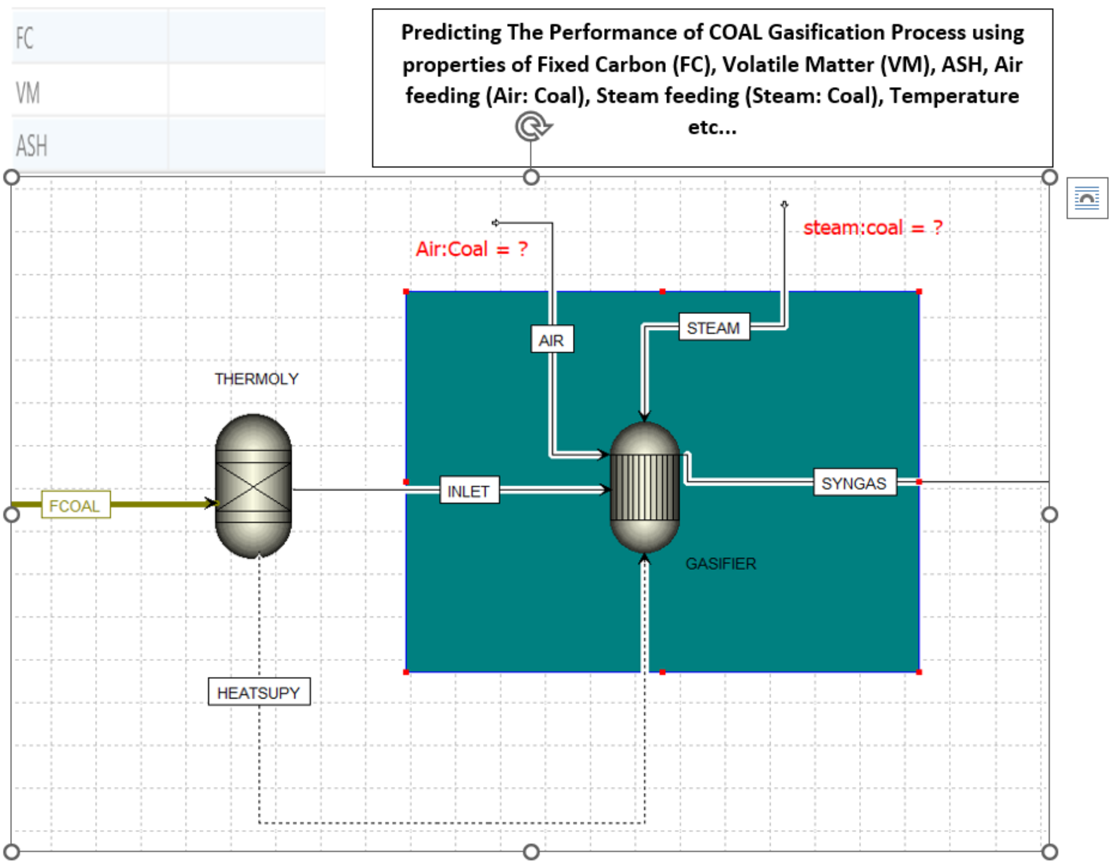
```
#import libraries
import numpy as np
import pandas as pd
import pycaret as pc
import matplotlib.pyplot as plt
import graphAnalysis #self made function
import countColumns #self made function
import seaborn as sns
import warnings
import IPython as ipy
import joblib
warnings.filterwarnings('ignore')
```

DIsplay of Case Study

In [2]:

```
ipy.display.Image(r'C:\Users\Joe\Desktop\coding\Matlab Revise\three.png')
```

Out[2]:



In [3]:

```
# dataset was compiled from the source
# Development of data-driven models for fluidized-bed coal gasification process
# link
# Features : Fixed carbon to Gas Produced
# Label class: Heat Value
df = pd.read_csv('dataset.csv')
df.head()
```

Out[3]:

	Fixed Carbon (FC)	Volatile Matter (VC)	ASH (MM)	Air Feeding (Nm^3/Kg)	Steam Feeding (kg/kg)	Temperature (C)	Heat Value (MJ/m^3)
0	26.89	33.20	39.91	1.93	0.35	904	4.14
1	26.89	33.20	39.91	1.98	0.35	910	4.70
2	26.89	33.20	39.91	2.09	0.44	888	3.96
3	26.89	33.20	39.91	2.10	0.37	907	4.42
4	27.31	49.56	23.14	2.41	0.43	912	4.23

In [4]:

```
countColumns.NumberOfColumns(df=df)
```

```
0 Fixed Carbon (FC)
1 Volatile Matter (VC)
2 ASH (MM)
3 Air Feeding (Nm^3/Kg)
4 Steam Feeding (kg/kg)
5 Temperature (C)
6 Heat Value (MJ/m^3)
The shape of the data is:  (106, 7)
```

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 106 entries, 0 to 105
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Fixed Carbon (FC)       106 non-null    float64
 1   Volatile Matter (VC)    106 non-null    float64
 2   ASH (MM)                106 non-null    float64
 3   Air Feeding (Nm^3/Kg)   106 non-null    float64
 4   Steam Feeding (kg/kg)   106 non-null    float64
 5   Temperature (C)         106 non-null    int64
 6   Heat Value (MJ/m^3)     106 non-null    float64
dtypes: float64(6), int64(1)
memory usage: 5.9 KB
```

In [6]:

```
df.describe()
```

Out[6]:

	Fixed Carbon (FC)	Volatile Matter (VC)	ASH (MM)	Air Feeding (Nm^3/Kg)	Steam Feeding (kg/kg)	Temperature (C)	Heat Value (MJ/m^3)
count	106.000000	106.000000	106.000000	106.000000	106.000000	106.000000	106.000000
mean	37.557736	32.821792	29.952075	2.345000	0.351698	858.801887	3.913774
std	11.864188	7.197448	12.024814	0.821092	0.112686	62.143064	0.943080
min	25.830000	21.290000	9.050000	1.340000	0.100000	720.000000	1.420000
25%	26.890000	25.760000	19.290000	1.712500	0.300000	825.000000	3.232500
50%	33.110000	33.200000	33.790000	2.130000	0.340000	850.000000	3.970000
75%	47.582500	36.480000	37.690000	2.535000	0.427500	900.000000	4.687500
max	61.000000	49.560000	49.980000	5.840000	0.630000	980.000000	5.500000

Using Pycaret Low Code Machine Learning

In [7]:

```
from pycaret.regression import *
```

In [8]:

```
exp = setup(data=df,target='Heat Value (MJ/m^3)', normalize=False,remove_outliers=True,sile
```

	Description	Value
0	session_id	6088
1	Target	Heat Value (MJ/m^3)
2	Original Data	(106, 7)
3	Missing Values	False
4	Numeric Features	6
5	Categorical Features	0
6	Ordinal Features	False
7	High Cardinality Features	False
8	High Cardinality Method	None
9	Transformed Train Set	(70, 6)
10	Transformed Test Set	(32, 6)

In [9]:

```
best = compare_models()
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
xgboost	Extreme Gradient Boosting	0.2464	0.0994	0.3063	0.7820	0.0627	0.0638	0.0340
et	Extra Trees Regressor	0.2641	0.1170	0.3266	0.7545	0.0682	0.0708	0.1210
gbr	Gradient Boosting Regressor	0.2604	0.1161	0.3322	0.7306	0.0676	0.0675	0.0220
ada	AdaBoost Regressor	0.2824	0.1273	0.3485	0.7101	0.0728	0.0758	0.0390
rf	Random Forest Regressor	0.2766	0.1353	0.3540	0.6920	0.0729	0.0736	0.1600
ridge	Ridge Regression	0.3465	0.2052	0.4275	0.6232	0.0889	0.0921	0.0140
lr	Linear Regression	0.3429	0.2108	0.4271	0.6197	0.0887	0.0908	1.2960
br	Bayesian Ridge	0.3497	0.2093	0.4312	0.6170	0.0898	0.0931	0.0120
huber	Huber Regressor	0.3583	0.2224	0.4447	0.5885	0.0924	0.0945	0.0200
lightgbm	Light Gradient Boosting Machine	0.3668	0.2285	0.4529	0.5558	0.0936	0.0966	0.0260
dt	Decision Tree Regressor	0.3736	0.2466	0.4610	0.4169	0.0947	0.0980	0.0130
knn	K Neighbors Regressor	0.4172	0.2944	0.5212	0.4106	0.1060	0.1095	0.0170
omp	Orthogonal Matching Pursuit	0.4588	0.3116	0.5390	0.3622	0.1128	0.1230	0.0130
en	Elastic Net	0.5043	0.3860	0.5917	0.2444	0.1221	0.1338	0.0150
lasso	Lasso Regression	0.5223	0.3954	0.6007	0.2042	0.1240	0.1387	0.0130
llar	Lasso Least Angle Regression	0.6634	0.5797	0.7407	-0.1396	0.1505	0.1756	0.0110
dummy	Dummy Regressor	0.6634	0.5797	0.7407	-0.1396	0.1505	0.1756	0.0110
lar	Least Angle Regression	0.5606	1.4593	0.7393	-1.6481	0.1350	0.1411	0.0130
par	Passive Aggressive Regressor	1.0037	1.4669	1.1656	-3.1239	0.2311	0.2541	0.0140

In [10]:

```
model = create_model(best)
print(model)
```

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.1247	0.0224	0.1496	0.9168	0.0289	0.0306
1	0.2285	0.0725	0.2692	0.7229	0.0586	0.0626
2	0.1837	0.0418	0.2044	0.9157	0.0419	0.0444
3	0.2275	0.0914	0.3024	0.7133	0.0578	0.0547
4	0.2978	0.1637	0.4046	0.8280	0.0911	0.0882
5	0.3226	0.1269	0.3563	0.7694	0.0656	0.0726
6	0.2663	0.1411	0.3757	0.8306	0.0750	0.0670
7	0.2500	0.1113	0.3336	0.4885	0.0637	0.0612
8	0.3081	0.1247	0.3531	0.8165	0.0808	0.0904
9	0.2546	0.0986	0.3140	0.8180	0.0636	0.0659
Mean	0.2464	0.0994	0.3063	0.7820	0.0627	0.0638
Std	0.0567	0.0418	0.0750	0.1173	0.0171	0.0172

```
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, feature_types=None, gamma=0, gpu_id=-1,
             grow_policy='depthwise', importance_type=None,
             interaction_constraints='', learning_rate=0.300000012, max_bin=
256,
             max_cat_threshold=64, max_cat_to_onehot=4, max_delta_step=0,
             max_depth=6, max_leaves=0, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=100, n_jobs=-1,
             num_parallel_tree=1, objective='reg:squarederror',
             predictor='auto', ...)
```

In [11]:

```
#Extracting parameters from pycaret's experiment
X = get_config(variable='X')
#X.head()
X_test = get_config(variable='X_test')
X_train = get_config(variable='X_train')
y_train= get_config(variable='y_train')
y_trained= get_config(variable='y_train')
y_test= get_config(variable='y_test')
#X_test.head()
y_test = pd.DataFrame(y_test).reset_index(drop=True)
y_trained = pd.DataFrame(y_trained).reset_index(drop=True)
#y_test
```

In [14]:

```
from sklearn.model_selection import cross_val_score
cross_val_score = cross_val_score(model,X_train,y_train)
cross_val_score.mean() # answer is above average ~80%
```

Out[14]:

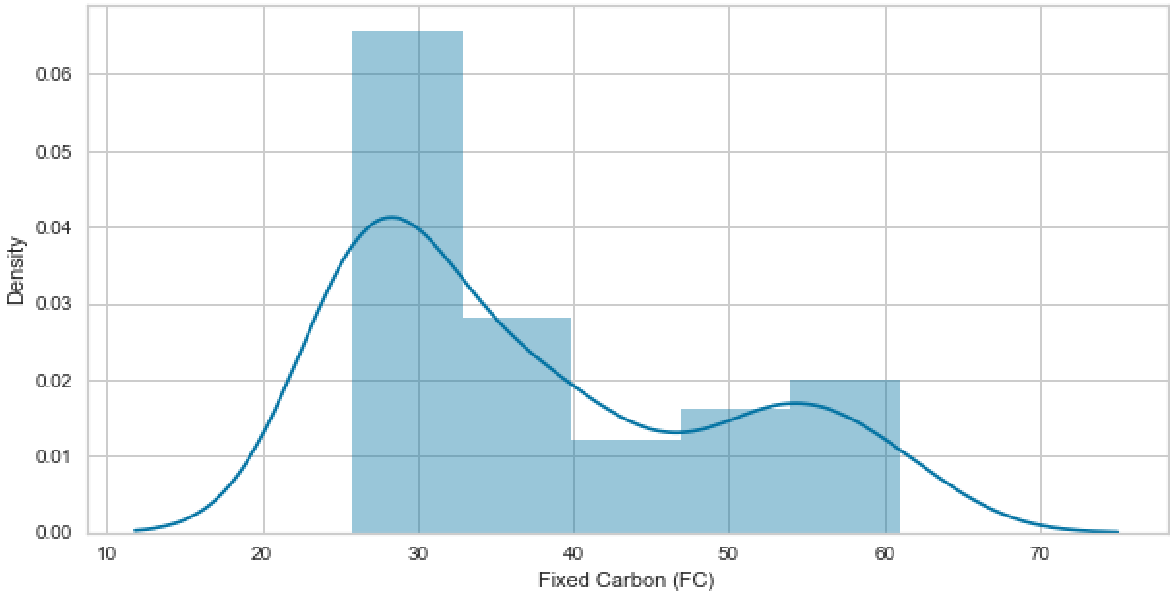
```
0.7978971388161538
```

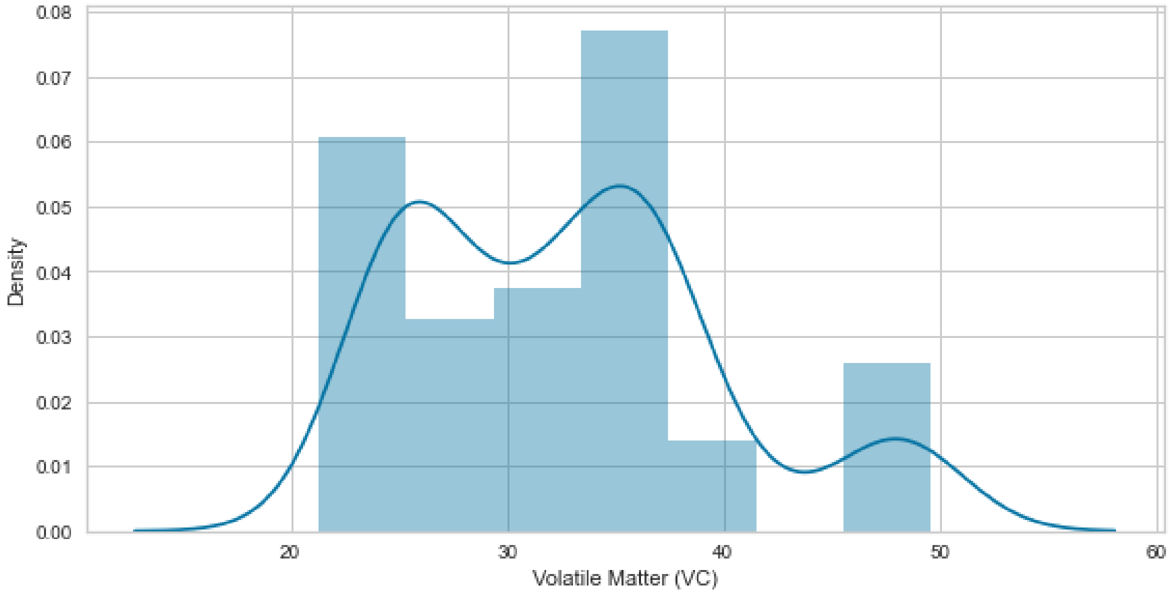
In [15]:

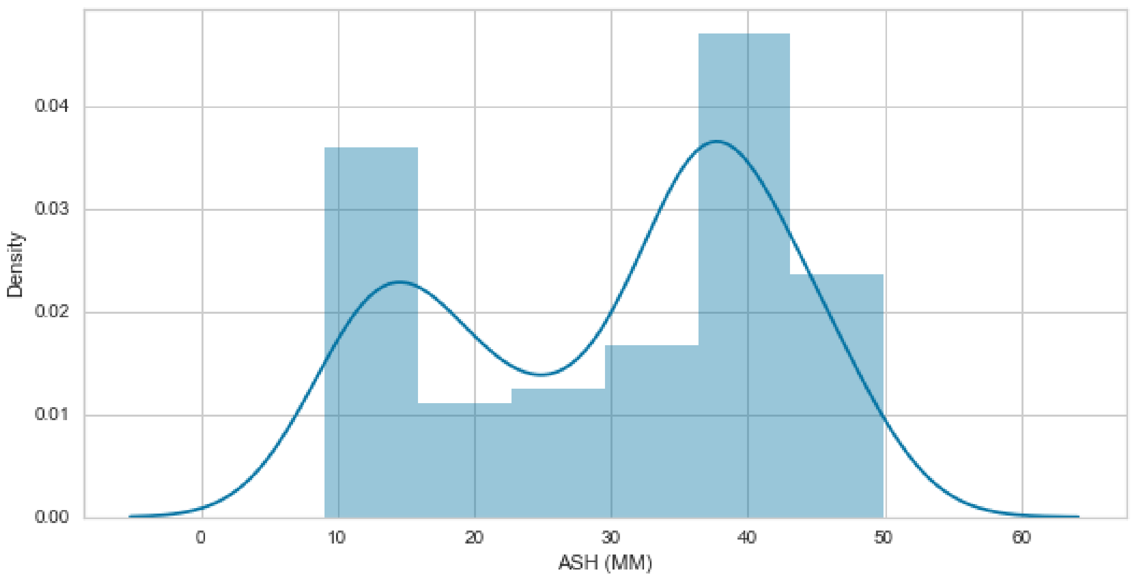
```
# Function to display box plot and dispplot
def distp_plot(df):
    for col in df.columns:
        plt.figure(figsize=(10,5))
        plt.subplot(1,1,1)
        sns.distplot(df[col])
        plt.show()
def box_plot(df):
    for col in df.columns:
        plt.figure(figsize=(10,5))
        plt.subplot(1,1,1)
        sns.boxplot(df[col])
        plt.show()
```

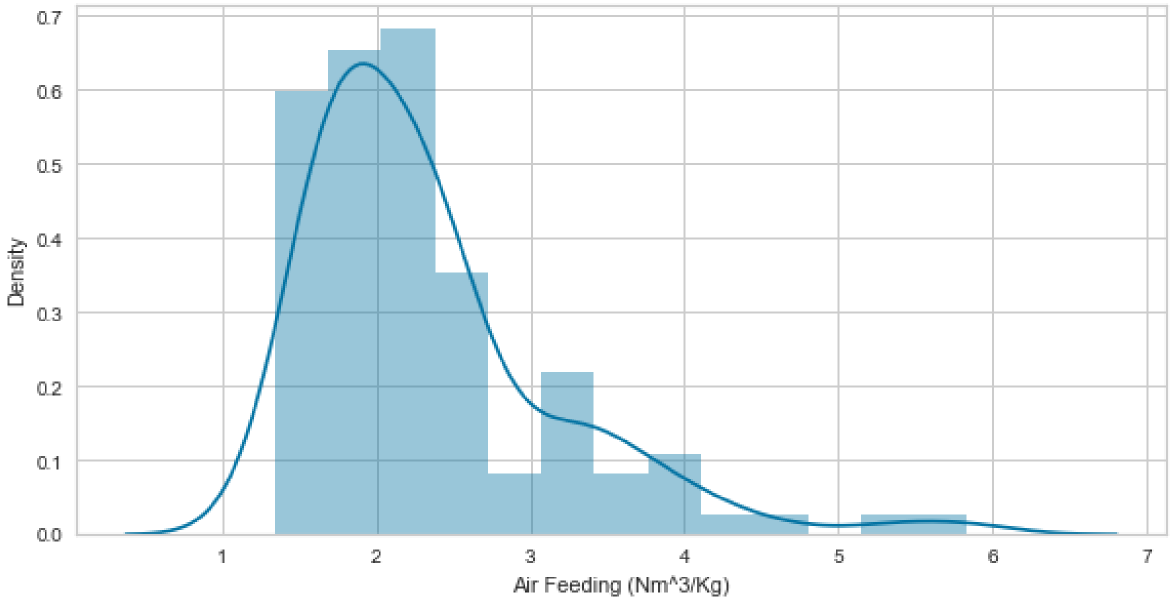
In [16]:

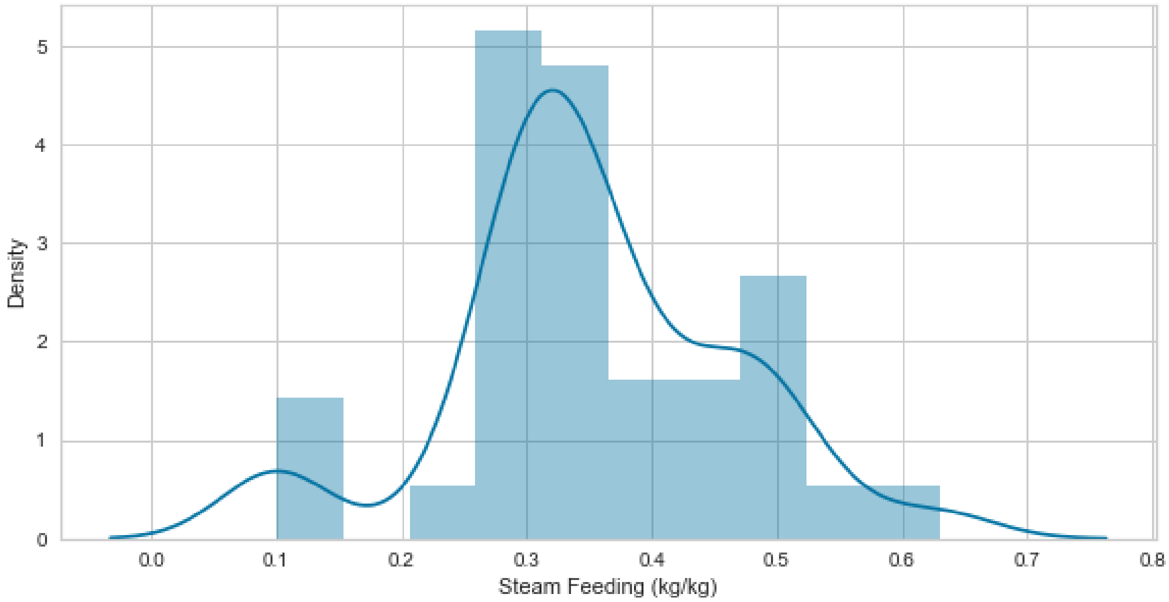
```
distp_plot(df=df)
```

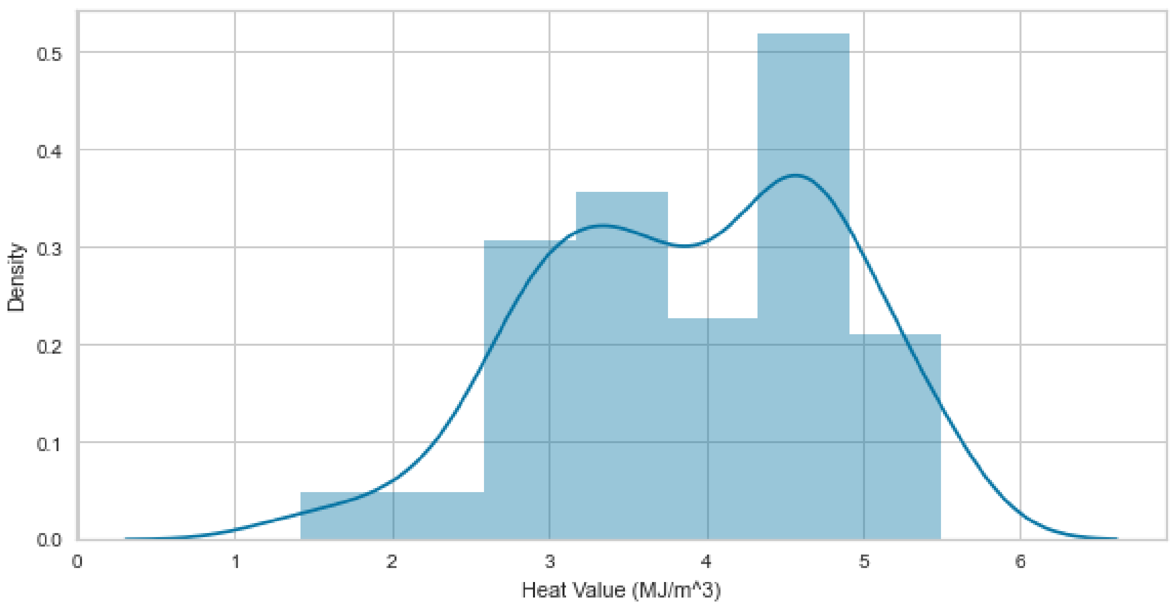







0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Density
1
2
3
4
5
6
7
Air Feeding (Nm^3/Kg)

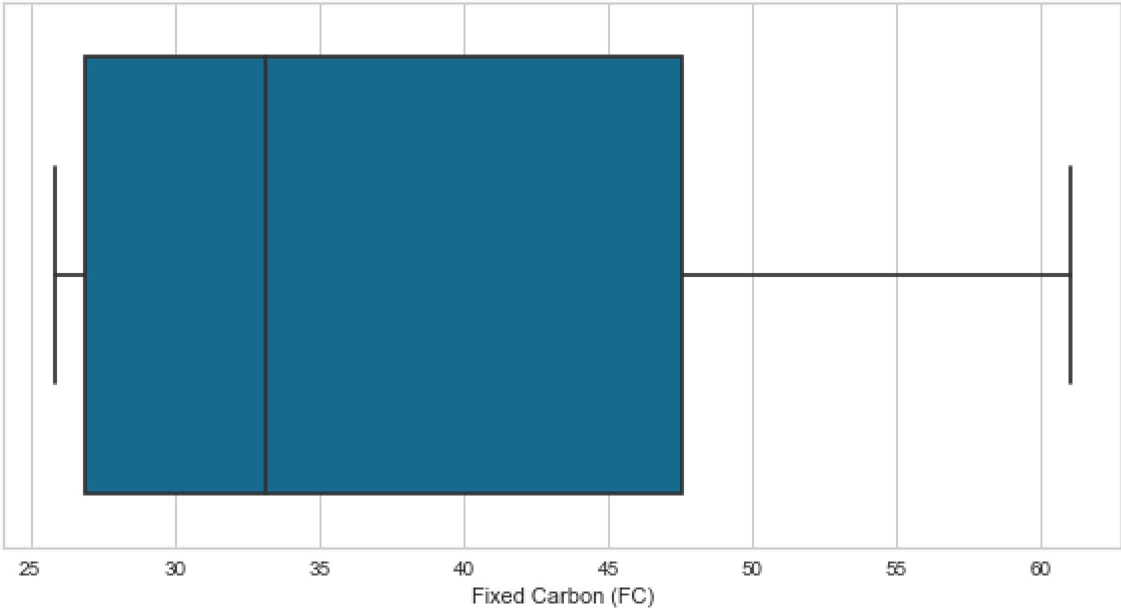

5
4
3
2
1
0
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Steam Feeding (kg/kg)

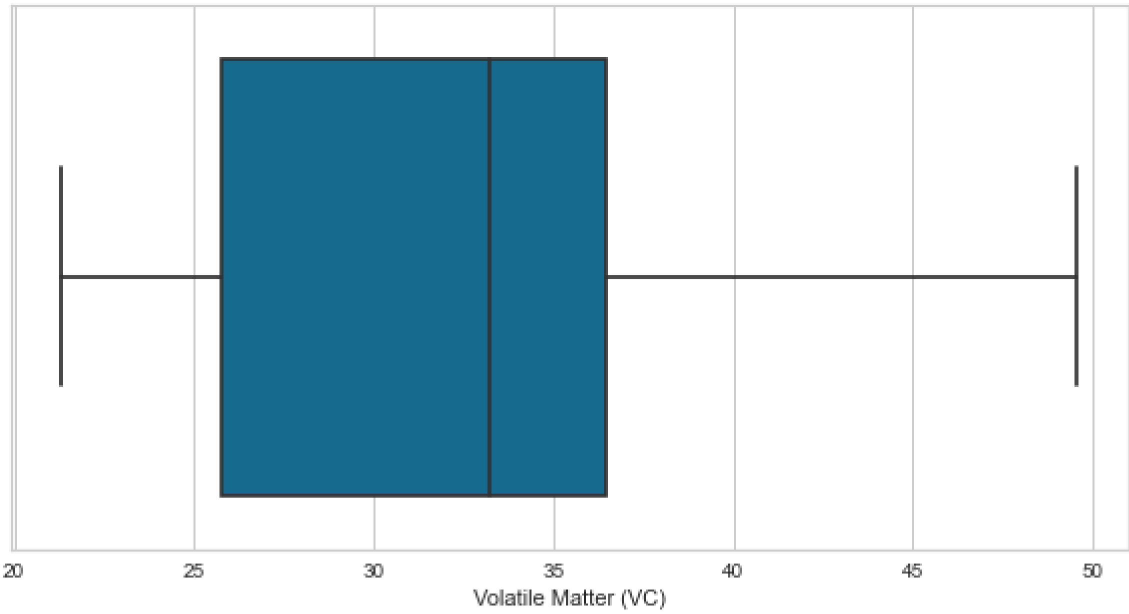


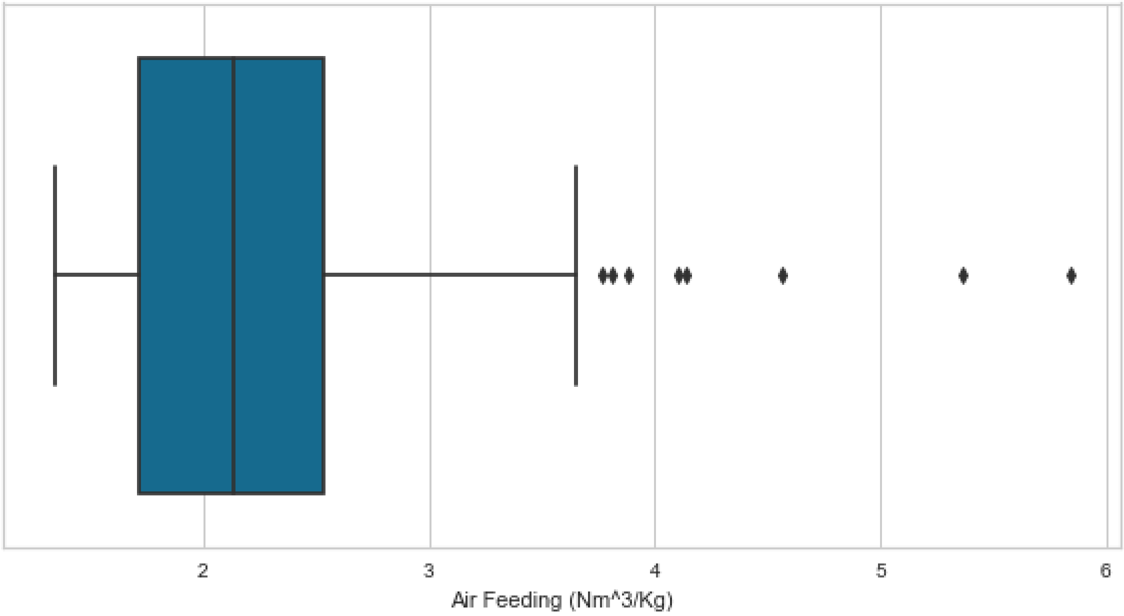
0.5
0.4
0.3
0.2
0.1
0.0
Density
0
1
2
3
4
5
6
Heat Value (MJ/m^3)

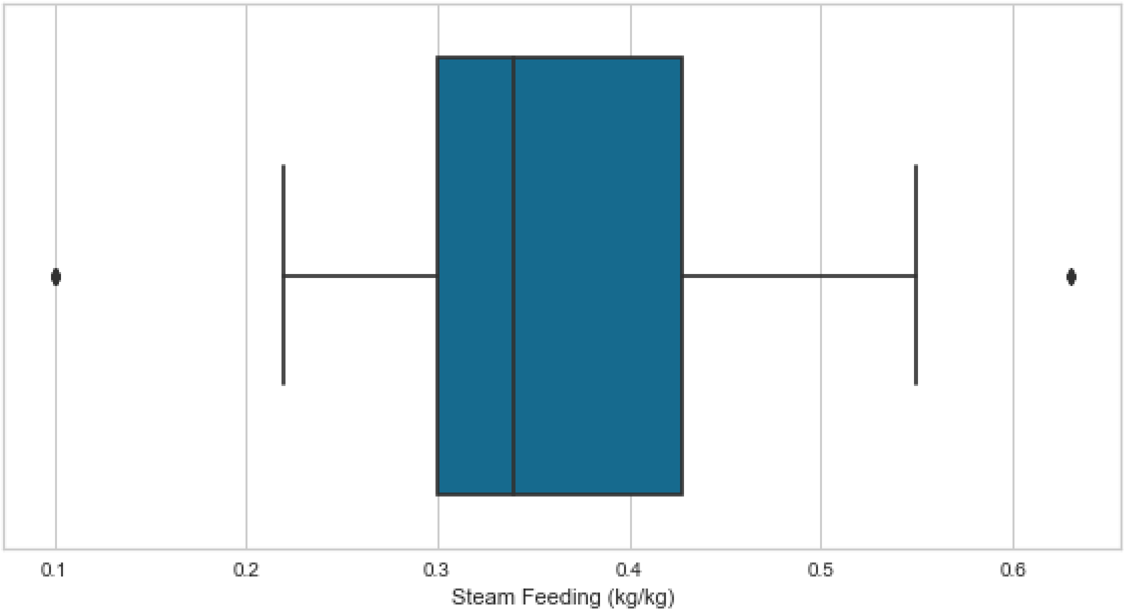
In [17]:

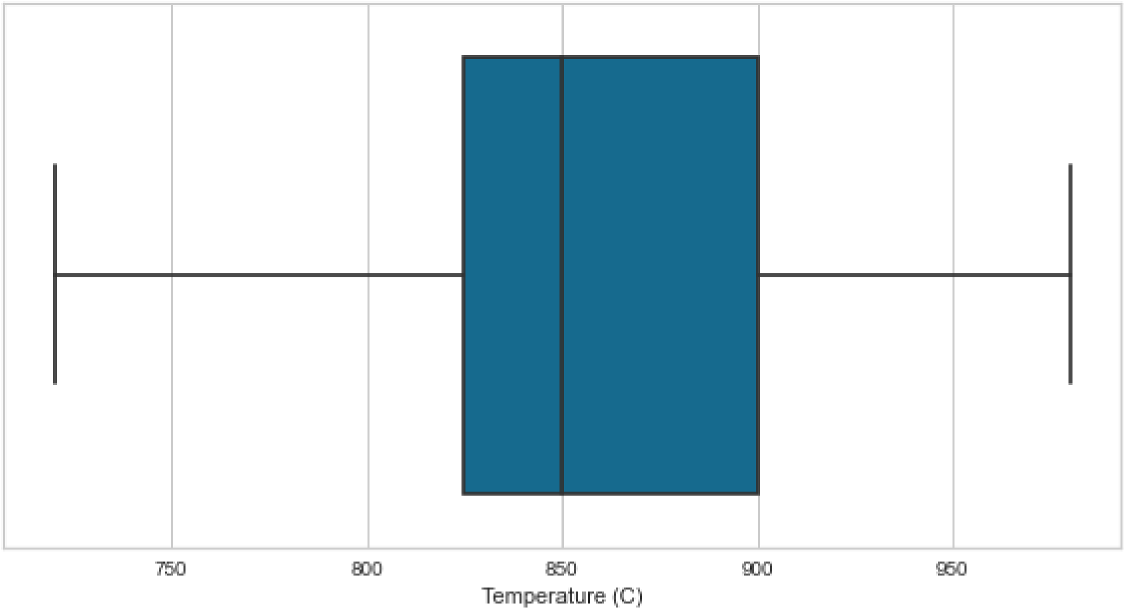
```
box_plot(df=df)
```

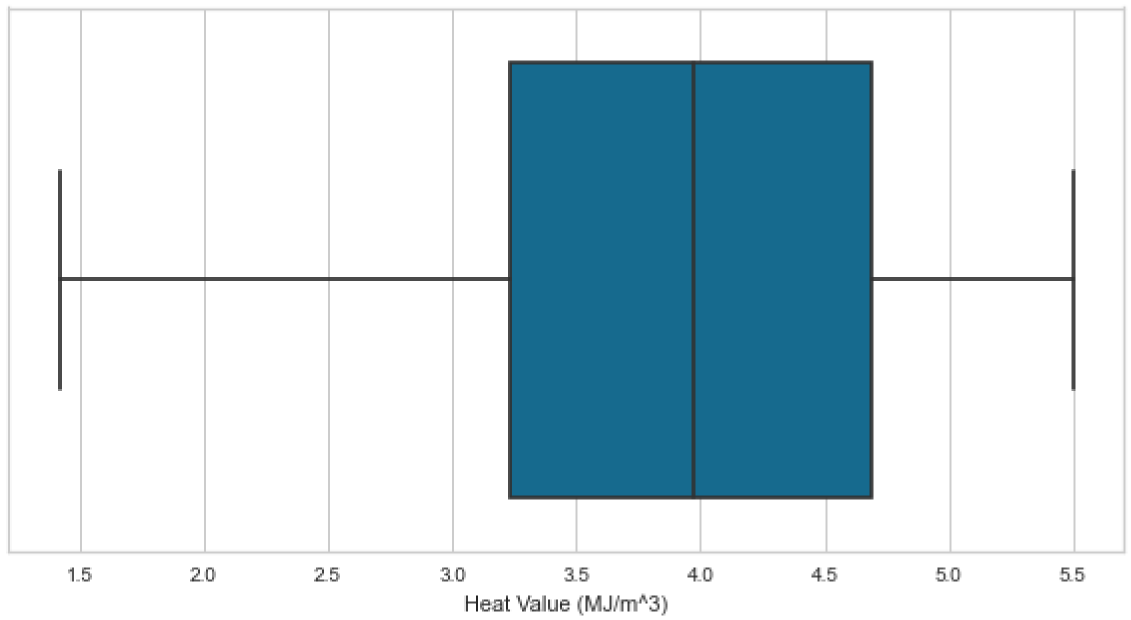





2
3
4
5
6
Air Feeding (Nm^3/Kg)


0.1
0.2
0.3
0.4
0.5
0.6
Steam Feeding (kg/kg)

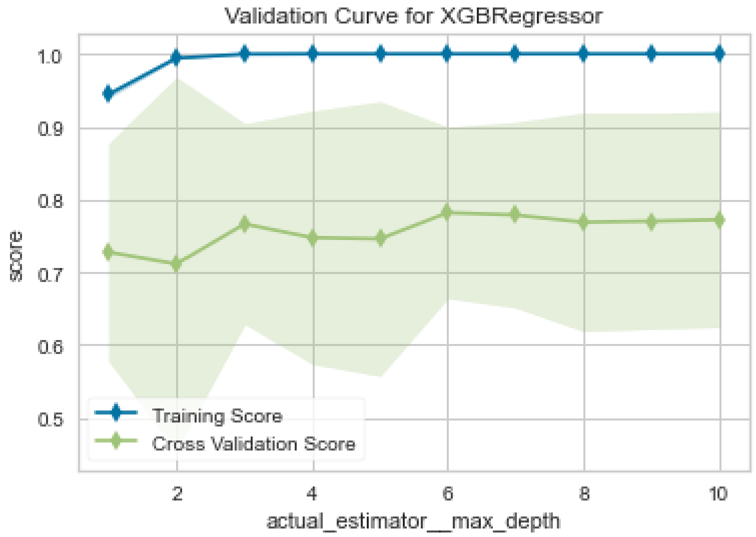

750
800
850
900
950
Temperature (C)



Various Visualization Plots

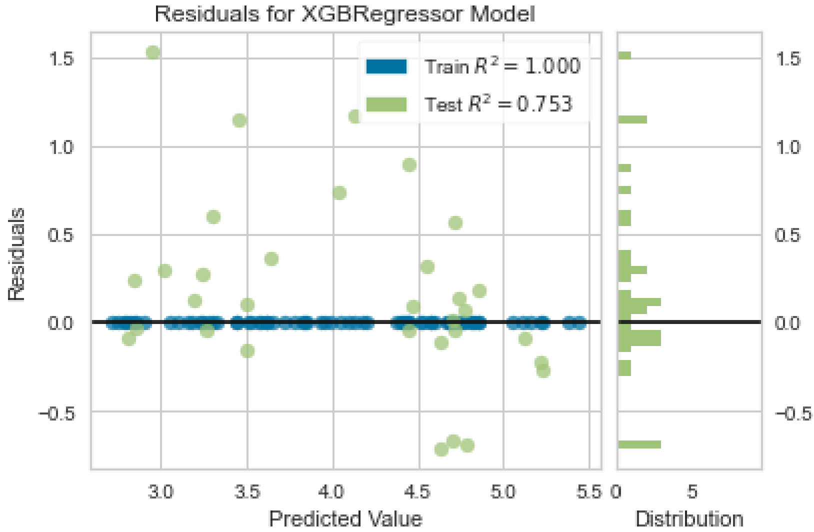
In [18]:

```
#Trianing score vs cross val score
plot_model(model,plot='vc')
```



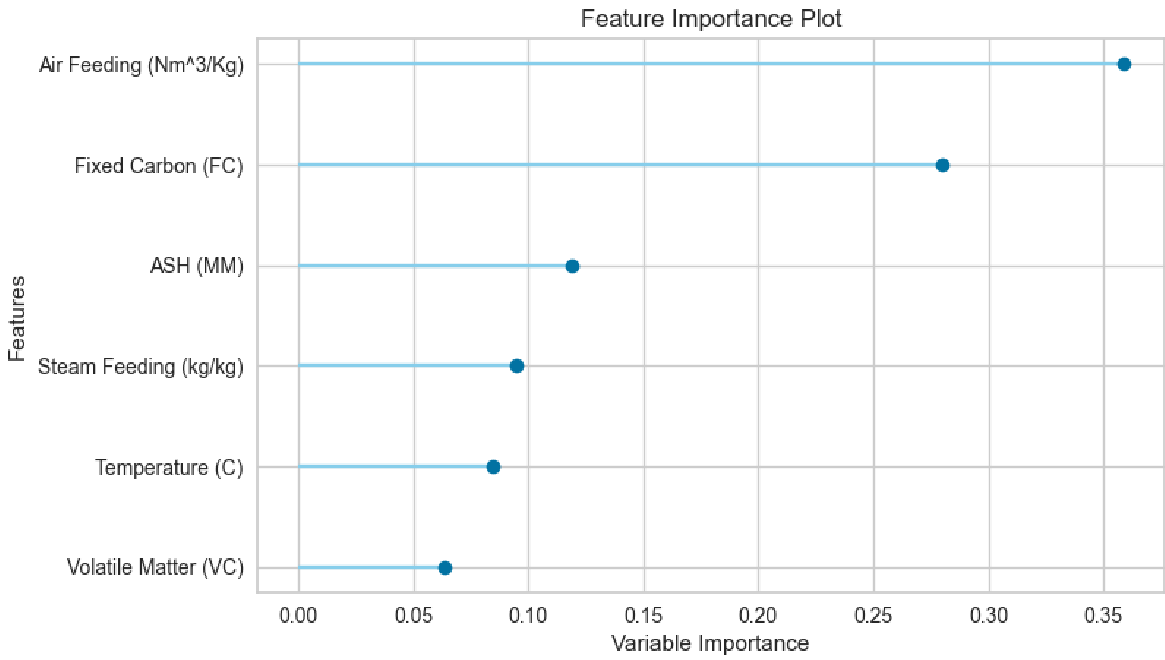
In [19]:

```
#Training and Test Rsquared
plot_model(model,plot='residuals')
```



In [20]:

```
#Feature Importance
plot_model(model,plot='feature')
```



Predicting values(x_test) using the model

In [21]:

```
predict_model(model,drift_report=False)
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Extreme Gradient Boosting	0.3759	0.2870	0.5357	0.7526	0.1320	0.1278

Out[21]:

	Fixed Carbon (FC)	Volatile Matter (VC)	ASH (MM)	Air Feeding (Nm^3/Kg)	Steam Feeding (kg/kg)	Temperature (C)	Heat Value (MJ/m^3)	Label
0	33.110001	47.580002	19.290001	2.98	0.10	840.0	2.90	2.813571
1	53.599998	33.910000	12.470000	1.42	0.30	850.0	5.47	4.785153
2	26.889999	33.200001	39.910000	1.98	0.35	910.0	4.70	4.707475
3	37.220001	25.760000	37.020000	1.68	0.30	800.0	4.76	4.715407
4	38.290001	27.920000	33.790001	1.81	0.34	765.0	5.21	5.120207
5	40.610001	31.590000	27.799999	1.86	0.35	750.0	5.44	5.217225
6	53.599998	33.910000	12.470000	2.26	0.30	850.0	3.55	4.440182
7	26.889999	33.200001	39.910000	1.93	0.35	904.0	4.14	4.710785
8	25.830000	36.480000	37.689999	2.28	0.49	937.0	3.66	3.501238
9	25.830000	36.480000	37.689999	2.54	0.46	935.0	2.97	3.246236
10	25.830000	36.480000	37.689999	2.28	0.49	941.0	3.40	3.501238
11	53.599998	33.910000	12.470000	2.61	0.30	850.0	3.31	4.041566
12	29.750000	25.049999	45.209999	1.67	0.28	860.0	4.75	4.633142
13	27.309999	49.560001	23.139999	2.41	0.43	912.0	4.23	4.549460
14	37.220001	25.760000	37.020000	1.70	0.30	800.0	4.60	4.739041
15	38.290001	27.920000	33.790001	2.15	0.40	725.0	5.34	4.628722
16	60.450001	37.990002	11.040000	5.84	0.63	850.0	1.42	2.947119
17	54.889999	31.020000	15.080000	3.35	0.38	900.0	2.88	2.851550
18	33.110001	47.580002	19.290001	2.19	0.10	750.0	2.70	3.301872
19	33.110001	47.580002	19.290001	2.57	0.10	810.0	2.60	2.843458
20	29.750000	25.049999	45.209999	1.70	0.34	800.0	4.70	4.766938
21	29.750000	25.049999	45.209999	1.67	0.30	950.0	4.48	4.439998
22	54.889999	31.020000	15.080000	3.35	0.30	900.0	2.72	3.020860
23	49.880001	41.060001	9.050000	4.14	0.42	850.0	2.31	3.452522
24	25.830000	36.480000	37.689999	2.39	0.32	890.0	3.27	3.639143
25	37.220001	25.760000	37.020000	1.50	0.28	800.0	4.67	4.851515
26	25.830000	36.480000	37.689999	2.30	0.51	958.0	3.06	3.191103
27	53.599998	33.910000	12.470000	2.32	0.29	900.0	2.96	4.128601
28	33.110001	47.580002	19.290001	2.35	0.10	810.0	3.30	3.259234
29	38.290001	27.920000	33.790001	1.57	0.34	740.0	5.37	4.700952

	Fixed Carbon (FC)	Volatile Matter (VC)	ASH (MM)	Air Feeding (Nm^3/Kg)	Steam Feeding (kg/kg)	Temperature (C)	Heat Value (MJ/m^3)	Label
30	29.750000	25.049999	45.209999	1.63	0.29	900.0	4.37	4.462823
31	40.610001	31.590000	27.799999	1.82	0.34	750.0	5.50	5.225820

Saving the model with pipeline

In [22]:

```
save_model(model,'savedmodel')
```

```
Transformation Pipeline and Model Successfully Saved
```

Out[22]:

```
(Pipeline(memory=None,
          steps=[('dtypes',
                  DataTypes_Auto_infer(categorical_features=[],
                                       display_types=False, features_todrop=
[],
                                       id_columns=[], ml_usecase='regressio
n',
                                       numerical_features=[],
                                       target='Heat Value (MJ/m^3)',
                                       time_features=[])),
                 ('imputer',
                  Simple_Imputer(categorical_strategy='not_available',
                                 fill_value_categorical=None,
                                 fill_value_numerical=None,
                                 numer...
                               grow_policy='depthwise', importance_type=Non
e,
                               interaction_constraints='',
                               learning_rate=0.300000012, max_bin=256,
                               max_cat_threshold=64, max_cat_to_onehot=4,
                               max_delta_step=0, max_depth=6, max_leaves=0,
                               min_child_weight=1, missing=nan,
                               monotone_constraints='()', n_estimators=100,
                               n_jobs=-1, num_parallel_tree=1,
                               objective='reg:squarederror', predictor='aut
o', ...)]],
          verbose=False),
 'savedmodel.pkl')
```

Saving just the model

In [23]:

```
joblib.dump(model,'gasificationmodel.pkl')
```

Out[23]:

```
['gasificationmodel.pkl']
```

Load the model

In [24]:

```
loaded_model = load_model('savedmodel')
```

```
Transformation Pipeline and Model Successfully Loaded
```

In [25]:

```
model_gasification = joblib.load('gasificationmodel.pkl')
model_gasification.fit(X_train.values,y_train.values)
model_gasification.score(X_test,y_test)
```

Out[25]:

```
0.7526325211298037
```

In [26]:

```
####Testing loaded_model with extracted x_test from pycaret
#X_test
y_predicted = loaded_model.predict(X_test)
Predicted_Heat_Value  = pd.DataFrame(y_predicted,columns=['Predicted Heat Value (MJ/m^3)'])
Predicted_Heat_Value.head()
y_test_df = y_test
```

In [27]:

```
#Merging y_test and y_predicted
merged = pd.concat([y_test_df,Predicted_Heat_Value],axis=1)
merged
```

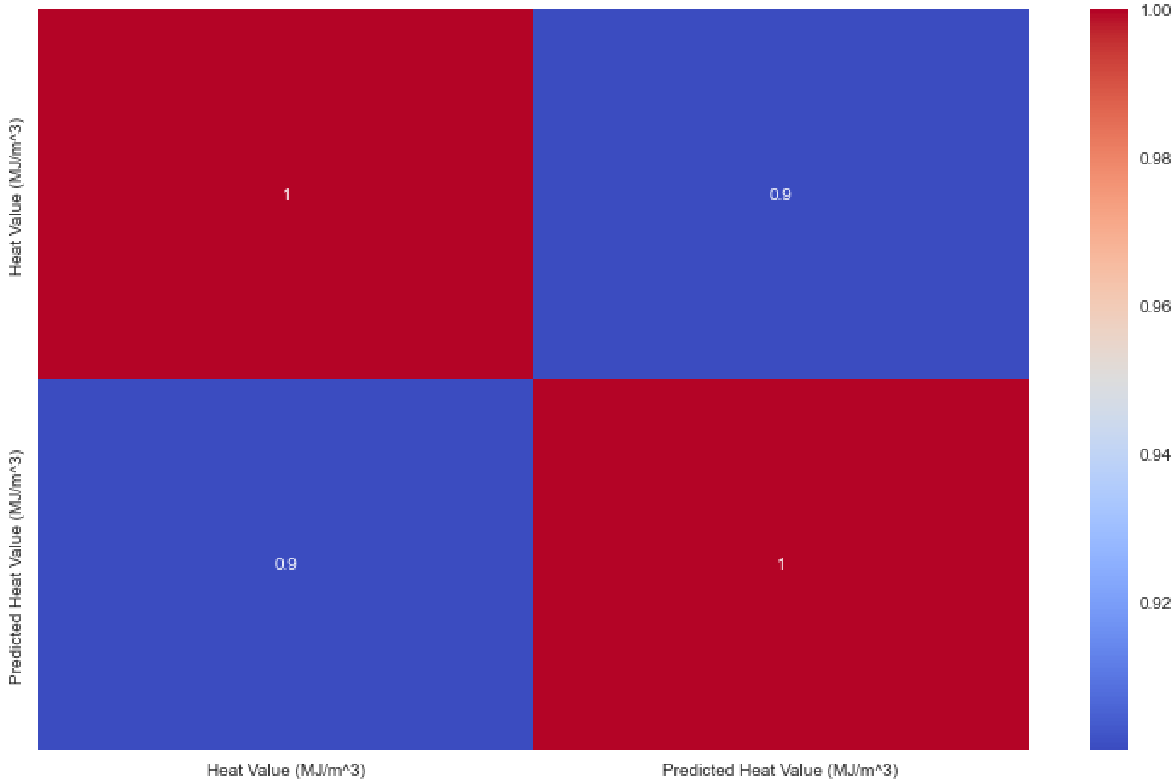
Out[27]:

	Heat Value (MJ/m^3)	Predicted Heat Value (MJ/m^3)
0	2.90	2.813571
1	5.47	4.785153
2	4.70	4.707475
3	4.76	4.715407
4	5.21	5.120207
5	5.44	5.217225
6	3.55	4.440182
7	4.14	4.710785
8	3.66	3.501238
9	2.97	3.246236
10	3.40	3.501238
11	3.31	4.041566
12	4.75	4.633142
13	4.23	4.549460
14	4.60	4.739041
15	5.34	4.628722
16	1.42	2.947119
17	2.88	2.851550
18	2.70	3.301872
19	2.60	2.843458
20	4.70	4.766938
21	4.48	4.439998
22	2.72	3.020860
23	2.31	3.452522
24	3.27	3.639143
25	4.67	4.851515
26	3.06	3.191103
27	2.96	4.128601
28	3.30	3.259234
29	5.37	4.700952
30	4.37	4.462823
31	5.50	5.225820

Visualizing Actual Value versus Prediced Value

In [28]:

```
#Correlation plot
plt.figure(figsize=(13,8))
sns.heatmap(merged.corr(), cmap='coolwarm', annot=True)
plt.show()
```



In [29]:

```
#Manual submission of imputs for predictions
#Recall columns
features = countColumns.NumberOfColumns(df=df)
```

```
0 Fixed Carbon (FC)
1 Volatile Matter (VC)
2 ASH (MM)
3 Air Feeding (Nm^3/Kg)
4 Steam Feeding (kg/kg)
5 Temperature (C)
6 Heat Value (MJ/m^3)
The shape of the data is:  (106, 7)
```

In [30]:

```
X_train
```

Out[30]:

	Fixed Carbon (FC)	Volatile Matter (VC)	ASH (MM)	Air Feeding (Nm^3/Kg)	Steam Feeding (kg/kg)	Temperature (C)
14	29.750000	25.049999	45.209999	1.66	0.30	950.0
19	29.750000	25.049999	45.209999	1.67	0.37	725.0
91	25.830000	36.480000	37.689999	2.11	0.30	896.0
22	29.750000	25.049999	45.209999	1.68	0.37	850.0
15	29.750000	25.049999	45.209999	1.67	0.32	810.0
...	...	...	...	...	...	...
16	29.750000	25.049999	45.209999	1.68	0.32	850.0
3	26.889999	33.200001	39.910000	2.10	0.37	907.0
94	25.830000	36.480000	37.689999	2.25	0.54	935.0
49	44.750000	24.040001	31.200001	1.73	0.32	950.0
63	53.599998	33.910000	12.470000	2.75	0.30	850.0

70 rows × 6 columns

Note: Sum of Fixed Carbon,Volatile Matter and ASH =100%

In [34]:

```
#input parameters
inputs = [[60,20,20,2.4,0.24,1000]]
```

In [35]:

```
def predict(data):
    mode = model_gasification.predict(data)
    ans =  mode[0]
    fin = round((ans),3)
    print(f'The Predicted Heating Value is ' + str(fin) )
```

In [36]:

```
predict(data=inputs)
```

```
The Predicted Heating Value is 3.911
```