

main\_new\_4\_2\_2\_0

August 28, 2021

## 1 IMPORTANT LIBRARIES

```
[1]: # Warning Libraries :  
import warnings  
warnings.filterwarnings("ignore")  
  
[2]: # Scientific and Data Manipulation Libraries :  
import pandas as pd  
import numpy as np  
from numpy import percentile  
import math  
import os  
from sklearn.model_selection import train_test_split  
  
[3]: # Data Visualization Libraries :  
%matplotlib inline  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
[4]: #pip install lasio  
  
[5]: #Libraries to convert .las files to .csv and merge  
  
import lasio  
from sys import stdout  
import glob ##For merging csv files  
  
[6]: #DATA IMPUTATION LIBRARY  
from sklearn.experimental import enable_iterative_imputer  
from sklearn.impute import IterativeImputer  
from sklearn.impute import KNNImputer  
from sklearn.linear_model import LinearRegression  
  
[7]: #Feature Selection Libraries  
from sklearn.feature_selection import VarianceThreshold  
from sklearn.feature_selection import mutual_info_classif  
from sklearn.feature_selection import SelectKBest
```

```
[8]: #SCALING LIBRARIES
from sklearn.preprocessing import StandardScaler, MinMaxScaler, Normalizer,
↳RobustScaler, MaxAbsScaler

[9]: #pip install catboost

[10]: #MODEL TRAINING LIBRARIES
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from catboost import CatBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import VotingClassifier
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from sklearn.ensemble import RandomForestClassifier

[11]: #MODEL ACCURACY LIBRARIES
from sklearn import metrics
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

[12]: #grid searching key hyperparametres for logistic regression
from sklearn.datasets import make_blobs
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import GridSearchCV

[13]: path='/media/mr-robot/Local Disk/summer_training/Train'
os.chdir(path)
```

## 2 LAS TO CSV

```
[14]: # Converting all las files in csv by iterating using lasio
for file in os.listdir():
    if file.endswith(".las"):
        file_path = f"{path}/{file}"
        las=lasio.read(file_path)
        size=len(file_path)
        filepath1=file_path[:size-3]
        las.to_csv(filepath1+'csv', units=False)

[15]: # Adding Well name to easily identify
for file in os.listdir():
    if file.endswith(".csv"):
        s=pd.read_csv(file)
        size=len(file)
        dict=[]
        filename= file[:size-4]
```

```

t=s.shape[0]
for i in range(t):
    dict.append(filename)
s['WELL']=dict
s.to_csv(filename+'.csv',index=False)

```

```

[16]: ## To avoid furthur merging data and redundancy
if(os.path.isfile('./merged_data.csv')):
    os.remove("merged_data.csv")

if(os.path.isfile('./FACIES_imputed.csv')):
    os.remove("FACIES_imputed.csv")

if(os.path.isfile('./FACIES_TRAIN.csv')):
    os.remove("FACIES_TRAIN.csv")

```

```

[17]: # Merging all Well Log using Glob
filenames = glob.glob(path + "/*.csv")
dfs = []
for filename in filenames:
    dfs.append(pd.read_csv(filename))
big_frame = pd.concat(dfs, ignore_index=True)
big_frame.to_csv('merged_data.csv',index=False)

```

### 3 IMPUTATION

```

[18]: df = pd.read_csv('merged_data.csv')
df

```

```

[18]:
      DEPTH  ACOUSTICIMPEDANCE1      AI  AVG_PIGN  CALI  \
0    1275.0552      12875.0811  12875081.0      NaN  9.7141
1    1275.2076      12854.2256  12854226.0      NaN  9.7848
2    1275.3600      13024.1377  13024138.0      NaN  9.8300
3    1275.5124      13093.3428  13093343.0      NaN  9.8587
4    1275.6648      13169.9307  13169931.0      NaN  9.8756
...      ...      ...      ...      ...      ...
58494  1622.6028      6069.1309  6069130.5      NaN  8.5257
58495  1622.7552      6067.8120  6067812.0      NaN  8.5282
58496  1622.9076      6105.7729  6105773.0      NaN  8.5313
58497  1623.0600      6152.9897  6152977.5      NaN  8.5331
58498  1623.2124      6157.8291  6157829.5      NaN  8.5338

      CALI[DERIVED]1      DT  FACIES  FLD1      GR  ...  CALI_1  NPHI_1  \
0           9.7141  50.2544      NaN  NaN  50.2128  ...      NaN      NaN
1           9.7848  50.3881      NaN  NaN  49.7509  ...      NaN      NaN
2           9.8300  49.8852      NaN  NaN  48.2513  ...      NaN      NaN
3           9.8587  49.9032      NaN  NaN  46.8212  ...      NaN      NaN

```

4	9.8756	50.0157	NaN	NaN	45.3463	...	NaN	NaN
...	...	...	...	...	...	...	...	...
58494	NaN	123.7404	NaN	NaN	NaN	...	NaN	0.4993
58495	NaN	123.8728	NaN	NaN	NaN	...	NaN	0.5313
58496	NaN	123.3722	NaN	NaN	NaN	...	NaN	0.5448
58497	NaN	122.6038	NaN	NaN	NaN	...	NaN	0.5364
58498	NaN	122.3045	NaN	NaN	NaN	...	NaN	0.5331

	ZCOR	RHOB_1	RXO	SPDH	DTDS	M2R1	TH	U
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...
58494	NaN	2.4639	NaN	NaN	123.7404	1.5970	NaN	NaN
58495	NaN	2.4660	NaN	NaN	123.8728	1.6128	NaN	NaN
58496	NaN	2.4714	NaN	NaN	123.3722	1.7043	NaN	NaN
58497	NaN	2.4750	NaN	NaN	122.6038	1.8375	NaN	NaN
58498	NaN	2.4709	NaN	NaN	122.3045	1.9363	NaN	NaN

[58499 rows x 67 columns]

```
[19]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 58499 entries, 0 to 58498
Data columns (total 67 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   DEPTH                                58499 non-null  float64
1   ACOUSTICIMPEDANCE1                  58499 non-null  float64
2   AI                                  55259 non-null  float64
3   AVG_PIGN                             323 non-null    float64
4   CALI                                54981 non-null  float64
5   CALI[DERIVED]1                      44090 non-null  float64
6   DT                                  58499 non-null  float64
7   FACIES                              52641 non-null  float64
8   FLD1                                3963 non-null   float64
9   GR                                  58379 non-null  float64
10  LLD                                 44942 non-null  float64
11  LLS                                 27394 non-null  float64
12  DEPTH_1                             50885 non-null  float64
13  NPHI                                58172 non-null  float64
14  ONE-WAYTIME1                        15713 non-null  float64
15  PIGN_MODELLING                      51101 non-null  float64
16  PIMP                                55259 non-null  float64
17  RHOB                                58499 non-null  float64
```

18	RT_MODELLING	53629	non-null	float64
19	SP	55992	non-null	float64
20	SUWI_MODELLING	51099	non-null	float64
21	TDVSS	58437	non-null	float64
22	ZLT	44562	non-null	float64
23	WELL	58499	non-null	object
24	DFL	23458	non-null	float64
25	HDRS	26951	non-null	float64
26	HMRS	26951	non-null	float64
27	PERF_INT	1569	non-null	float64
28	PERMEABILITY	28149	non-null	float64
29	PIGN	46949	non-null	float64
30	RT_POWER	51379	non-null	float64
31	SUWI	46947	non-null	float64
32	VCL	46947	non-null	float64
33	WATER_VOL	43735	non-null	float64
34	LL3	12373	non-null	float64
35	BS	6706	non-null	float64
36	CALI1	2389	non-null	float64
37	DEVI	10283	non-null	float64
38	DT1	6130	non-null	float64
39	PHIT	16532	non-null	float64
40	PIGE	5245	non-null	float64
41	LLD_1	9518	non-null	float64
42	SXWI	27938	non-null	float64
43	PEF	19419	non-null	float64
44	AZI1	2487	non-null	float64
45	TEMP	14514	non-null	float64
46	DRES	2765	non-null	float64
47	DT2	2765	non-null	float64
48	DT4P	5854	non-null	float64
49	GR_EDTC	2765	non-null	float64
50	M2R2	8568	non-null	float64
51	LLS_1	238	non-null	float64
52	MSFL	2765	non-null	float64
53	PR	2757	non-null	float64
54	TENS	2765	non-null	float64
55	VPVS	2757	non-null	float64
56	BIT	5553	non-null	float64
57	CALI_1	2999	non-null	float64
58	NPHI_1	10811	non-null	float64
59	ZCOR	2998	non-null	float64
60	RHOB_1	10899	non-null	float64
61	RXO	1552	non-null	float64
62	SPDH	3069	non-null	float64
63	DTDS	2546	non-null	float64
64	M2R1	2546	non-null	float64
65	TH	2509	non-null	float64

```
66 U                2509 non-null    float64
dtypes: float64(66), object(1)
memory usage: 29.9+ MB
```

```
[20]: df.shape[1]
```

```
[20]: 67
```

```
[21]: obj = df.isnull().sum()
      for key,value in obj.iteritems():
          print(key,",",value)
```

```
DEPTH , 0
ACOUSTICIMPEDANCE1 , 0
AI , 3240
AVG_PIGN , 58176
CALI , 3518
CALI[DERIVED]1 , 14409
DT , 0
FACIES , 5858
FLD1 , 54536
GR , 120
LLD , 13557
LLS , 31105
DEPTH_1 , 7614
NPHI , 327
ONE-WAYTIME1 , 42786
PIGN_MODELLING , 7398
PIMP , 3240
RHOB , 0
RT_MODELLING , 4870
SP , 2507
SUWI_MODELLING , 7400
TDVSS , 62
ZLT , 13937
WELL , 0
DFL , 35041
HDRS , 31548
HMRS , 31548
PERF_INT , 56930
PERMEABILITY , 30350
PIGN , 11550
RT_POWER , 7120
SUWI , 11552
VCL , 11552
WATER_VOL , 14764
LL3 , 46126
BS , 51793
```

```

CALI1 , 56110
DEVI , 48216
DT1 , 52369
PHIT , 41967
PIGE , 53254
LLD_1 , 48981
SXWI , 30561
PEF , 39080
AZI1 , 56012
TEMP , 43985
DRES , 55734
DT2 , 55734
DT4P , 52645
GR_EDTC , 55734
M2R2 , 49931
LLS_1 , 58261
MSFL , 55734
PR , 55742
TENS , 55734
VPVS , 55742
BIT , 52946
CALI_1 , 55500
NPHI_1 , 47688
ZCOR , 55501
RHOB_1 , 47600
RXO , 56947
SPDH , 55430
DTDS , 55953
M2R1 , 55953
TH , 55990
U , 55990

```

```

[22]: #Selecting required feature
df=df[["DT","GR","NPHI","RHOB","FACIES"]]

```

```

[23]: df

```

```

[23]:
      DT      GR      NPHI      RHOB      FACIES
0  50.2544  50.2128  0.5340  2.1228      NaN
1  50.3881  49.7509  0.5316  2.1250      NaN
2  49.8852  48.2513  0.5126  2.1316      NaN
3  49.9032  46.8212  0.5137  2.1437      NaN
4  50.0157  45.3463  0.5472  2.1611      NaN
...
58494  123.7404      NaN  0.4993  2.4639      NaN
58495  123.8728      NaN  0.5313  2.4660      NaN
58496  123.3722      NaN  0.5448  2.4714      NaN

```

```
58497 122.6038      NaN 0.5364 2.4750      NaN
58498 122.3045      NaN 0.5331 2.4709      NaN
```

```
[58499 rows x 5 columns]
```

```
[24]: df.isnull().sum()
```

```
[24]: DT          0
      GR         120
      NPFI        327
      RHOB         0
      FACIES      5858
      dtype: int64
```

```
[25]: #Exporting required features to csv
      df.to_csv("FACIES_TRAIN.csv",index=False)
```

```
[26]: df=pd.read_csv("FACIES_TRAIN.csv")
```

```
[27]: df.head(20)
```

```
[27]:
```

	DT	GR	NPFI	RHOB	FACIES
0	50.2544	50.2128	0.5340	2.1228	NaN
1	50.3881	49.7509	0.5316	2.1250	NaN
2	49.8852	48.2513	0.5126	2.1316	NaN
3	49.9032	46.8212	0.5137	2.1437	NaN
4	50.0157	45.3463	0.5472	2.1611	NaN
5	50.6831	44.0819	0.5550	2.1740	NaN
6	51.4311	43.6654	0.5612	2.1707	NaN
7	52.1678	43.3915	0.5566	2.1595	NaN
8	52.2883	44.1249	0.5390	2.1534	NaN
9	51.5991	46.1805	0.5245	2.1551	NaN
10	50.6185	48.6156	0.5152	2.1542	NaN
11	50.5171	49.6999	0.5152	2.1535	NaN
12	50.1209	49.4600	0.5180	2.1586	NaN
13	50.0558	48.3665	0.5156	2.1662	NaN
14	49.4216	46.8647	0.5070	2.1705	NaN
15	47.9804	45.7345	0.4913	2.1702	NaN
16	46.3324	45.5512	0.4696	2.1657	NaN
17	45.1378	45.9222	0.4570	2.1579	NaN
18	45.2291	46.4844	0.4654	2.1533	NaN
19	45.6106	49.6481	0.4952	2.1526	NaN

```
[28]: df.shape
```

```
[28]: (58499, 5)
```

```
[29]: df.info()
```



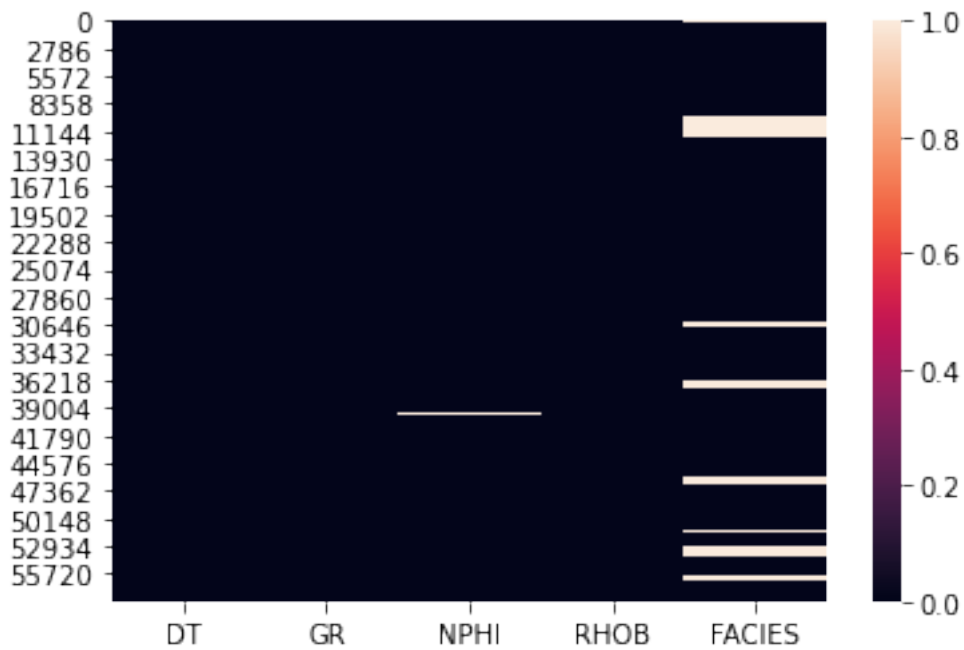
```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 58499 entries, 0 to 58498
Data columns (total 5 columns):
#   Column  Non-Null Count  Dtype  
---  -
0    DT      58499 non-null    float64
1    GR      58379 non-null    float64
2    NPHI     58172 non-null    float64
3    RHOB     58499 non-null    float64
4    FACIES   52641 non-null    float64
dtypes: float64(5)
memory usage: 2.2 MB

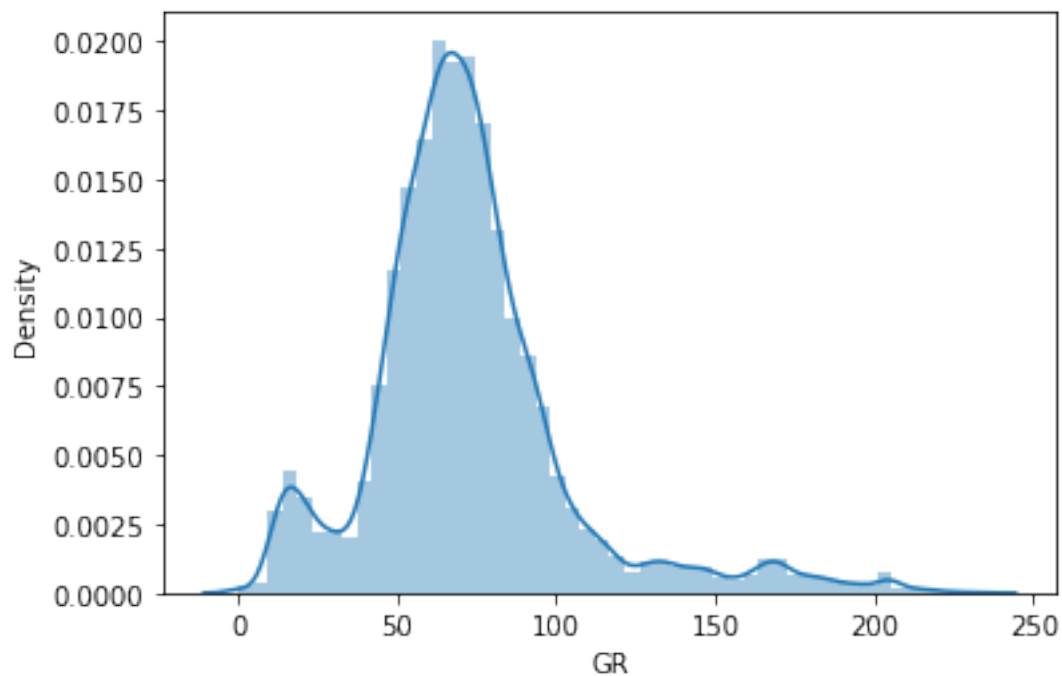
```

```
[30]: sns.heatmap(df.isnull())
```

```
[30]: <AxesSubplot:>
```



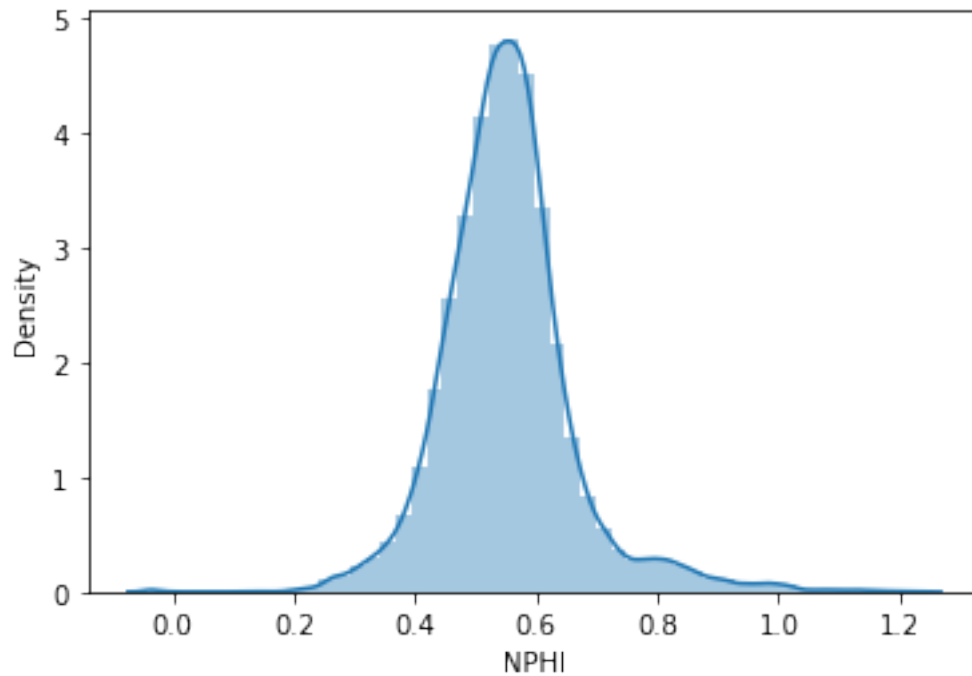
```
[31]: null_gr = sns.distplot(df.GR.dropna())
```



```
[32]: df.GR.describe()
```

```
[32]: count    58379.000000  
      mean      72.610942  
      std      32.140407  
      min       0.000000  
      25%      55.340300  
      50%      68.939700  
      75%      83.758300  
      max     233.707400  
      Name: GR, dtype: float64
```

```
[33]: null_nphi=sns.distplot(df.NPHI.dropna())
```

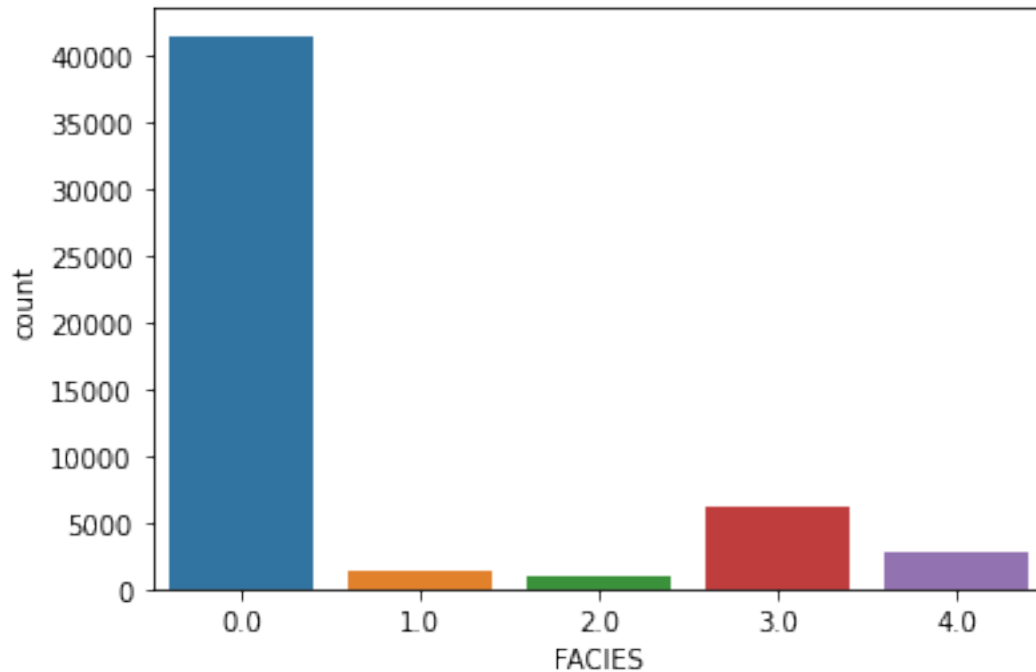


```
[34]: df.NPHI.describe()
```

```
[34]: count    58172.000000  
      mean      0.551710  
      std      0.109983  
      min     -0.038000  
      25%      0.489275  
      50%      0.546600  
      75%      0.600500  
      max      1.231200  
      Name: NPHI, dtype: float64
```

```
[35]: sns.countplot(x="FACIES",data=df)
```

```
[35]: <AxesSubplot:xlabel='FACIES', ylabel='count'>
```



```
[36]: df.FACIES.value_counts(dropna=False)
```

```
[36]: 0.0    41514
      3.0     6138
      NaN     5858
      4.0     2798
      1.0     1281
      2.0      910
      Name: FACIES, dtype: int64
```

```
[37]: def imputing(imputation_strategy,imputing_data):
      df=imputing_data
      if imputation_strategy == "Mean":
          df.GR.fillna(df.GR.mean(),inplace=True)
          print( df.GR.isnull().sum())
          print("Graph (GR) after filling null values with mean")
          sns.displot(df.GR.dropna())
          df.NPHI.fillna(df.NPHI.mean(),inplace=True)
          print("Graph (NPHI) after filling null values with mean")
          print(df.NPHI.isnull().sum())
          sns.displot(df.NPHI.dropna())
          #dropping FACIES rows with null
          df.dropna(axis=0,inplace=True)
          print(df.isnull().sum())
          df['FACIES'] = df.FACIES.astype(np.int64)
```

```

df.info()
df.FACIES.describe()
return df

elif imputation_strategy == "bfill":
    df = df.ffill(axis = 0)
    df = df.bfill(axis = 0)
    df['FACIES'] = df.FACIES.astype(np.int64)
    print(df.isnull().sum())
    return df

elif imputation_strategy == "KNNImputer":
    knn= KNNImputer(n_neighbors=3)
    X=df.drop('FACIES',1)
    t=knn.fit_transform(X)
    X=pd.DataFrame(t)
    Y=df['FACIES']
    Y=Y.ffill(axis=0)
    Y=Y.bfill(axis=0)
    X['FACIES']=Y
    df=X
    df['FACIES'] = df.FACIES.astype(np.int64)
    d=['DT', 'GR', 'NPHI', 'RHOB']
    for i in range(4):
        df.columns.values[i]=d[i]
    return df

elif imputation_strategy == "IterativeImputer":
    lr=LinearRegression()    #can use other regressions too. / default is
    ↪ bayesian
    imp=IterativeImputer(max_iter=3)
    X=df.drop('FACIES',1)
    t=imp.fit_transform(X)
    X=pd.DataFrame(t)
    Y=df['FACIES']
    Y=Y.ffill(axis=0)
    Y=Y.bfill(axis=0)
    X['FACIES']=Y
    df=X
    df['FACIES'] = df.FACIES.astype(np.int64)
    d=['DT', 'GR', 'NPHI', 'RHOB']
    for i in range(4):
        df.columns.values[i]=d[i]
    return df

elif imputation_strategy == "KNNImputer_floor" :
    X=df

```

```

knn= KNNImputer(n_neighbors=3)
t=knn.fit_transform(df)
df=pd.DataFrame(t)
d=['DT', 'GR', 'NPHI', 'RHOB', 'FACIES']
df['FACIES1'] = X.FACIES
for i in range(5):
    df.columns.values[i]=d[i]
df=df.drop('FACIES1',1)
df['FACIES'] = df.FACIES.astype(np.int64)
return df

elif imputation_strategy == "IterativeImputer_floor" :
X=df
lr=LinearRegression()
imp= IterativeImputer(max_iter=3)
t=imp.fit_transform(df)
df=pd.DataFrame(t)
d=['DT', 'GR', 'NPHI', 'RHOB', 'FACIES']
df['FACIES1'] = X.FACIES
for i in range(5):
    df.columns.values[i]=d[i]
df=df.drop('FACIES1',1)
df['FACIES'] = df.FACIES.astype(np.int64)
return df

elif imputation_strategy == "KNNBinning" :
X=df
knn= KNNImputer(n_neighbors=3)
t=knn.fit_transform(df)
df=pd.DataFrame(t)
d=['DT', 'GR', 'NPHI', 'RHOB', 'FACIES']
df['FACIES1'] = X.FACIES
for i in range(5):
    df.columns.values[i]=d[i]
df=df.drop('FACIES1',1)
#df['FACIES'] = pd.cut(x=df['FACIES'],bins=[0,0.5,1.5,2.5,3.5,4.0],
↪labels=['0','1','2','3','4'])
return df

elif imputation_strategy == "dropna":
df=df.dropna(axis=0)
return df

```

```

[38]: imputation_strategy = ["Mean" , "bfill" , "KNNImputer" , "IterativeImputer" ,
↪ "KNNImputer_floor" , "IterativeImputer_floor" , "KNNBinning","dropna"]
#select option from 0-7 (6 is experimental)
optionimputation=4

```

```
df=imputing(imputation_strategy[optionimputation],df)
```

```
[39]: #if option==6:  
#     df['FACIES'] = pd.cut(x=df['FACIES'],bins=[0.0,0.5,1.5,2.5,3.5,4.0],  
→ labels=['0','1','2','3','4'])
```

```
[40]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 58499 entries, 0 to 58498  
Data columns (total 5 columns):  
#   Column   Non-Null Count  Dtype  
---  ---  
0    DT       58499 non-null  float64  
1    GR       58499 non-null  float64  
2    NPFI     58499 non-null  float64  
3    RHOB     58499 non-null  float64  
4    FACIES   58499 non-null  int64  
dtypes: float64(4), int64(1)  
memory usage: 2.2 MB
```

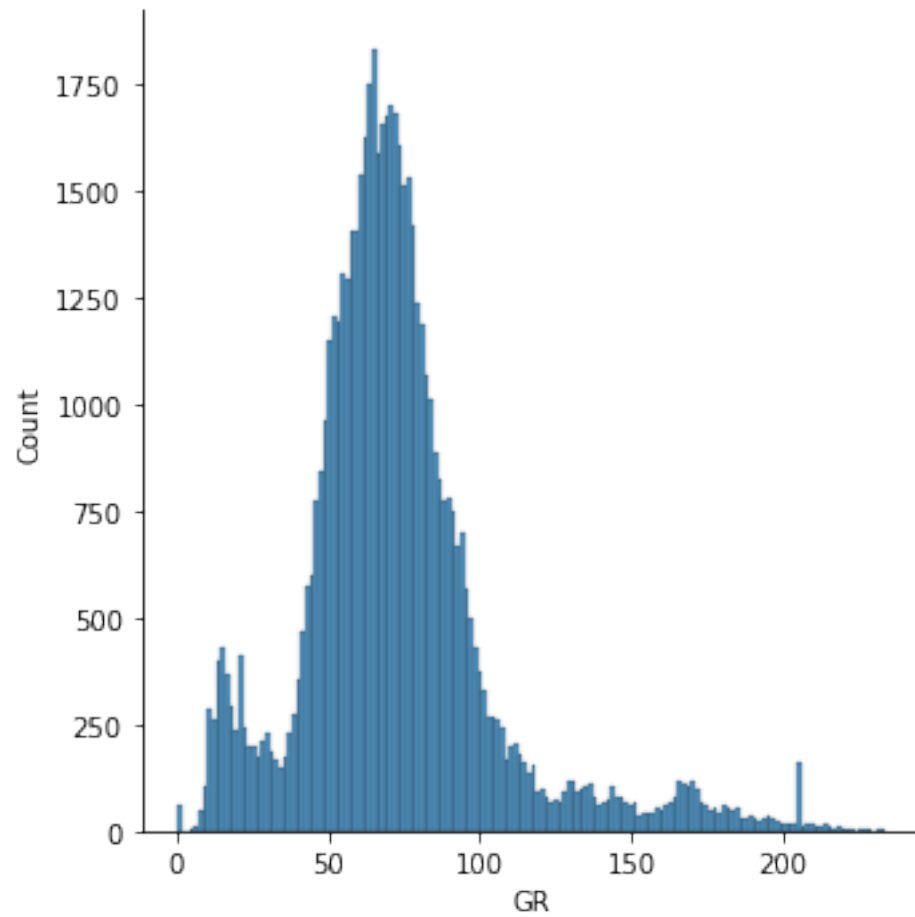
```
[41]: df.isnull().sum()
```

```
[41]: DT          0  
GR           0  
NPFI         0  
RHOB         0  
FACIES       0  
dtype: int64
```

```
[42]: df.to_csv("FACIES_imputed.csv",index=False)  
df=pd.read_csv("FACIES_imputed.csv")
```

```
[43]: sns.displot(df.GR.dropna())
```

```
[43]: <seaborn.axisgrid.FacetGrid at 0x7fa3042a8be0>
```

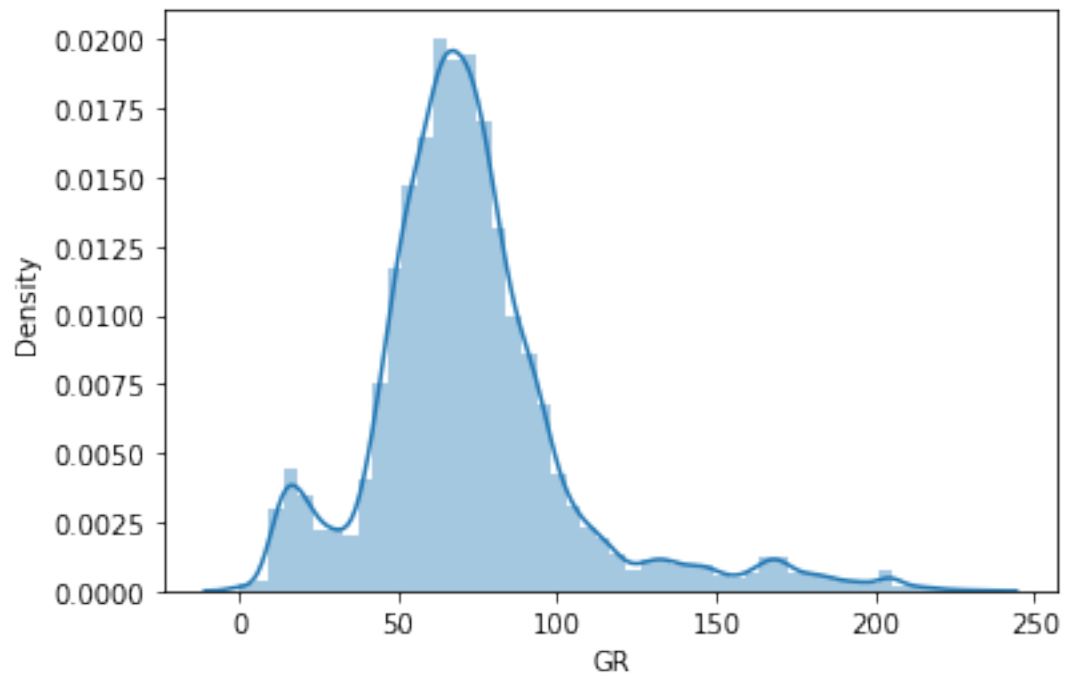


```
[44]: print("WHEN GR WAS NULL")  
      null_gr.figure
```

WHEN GR WAS NULL

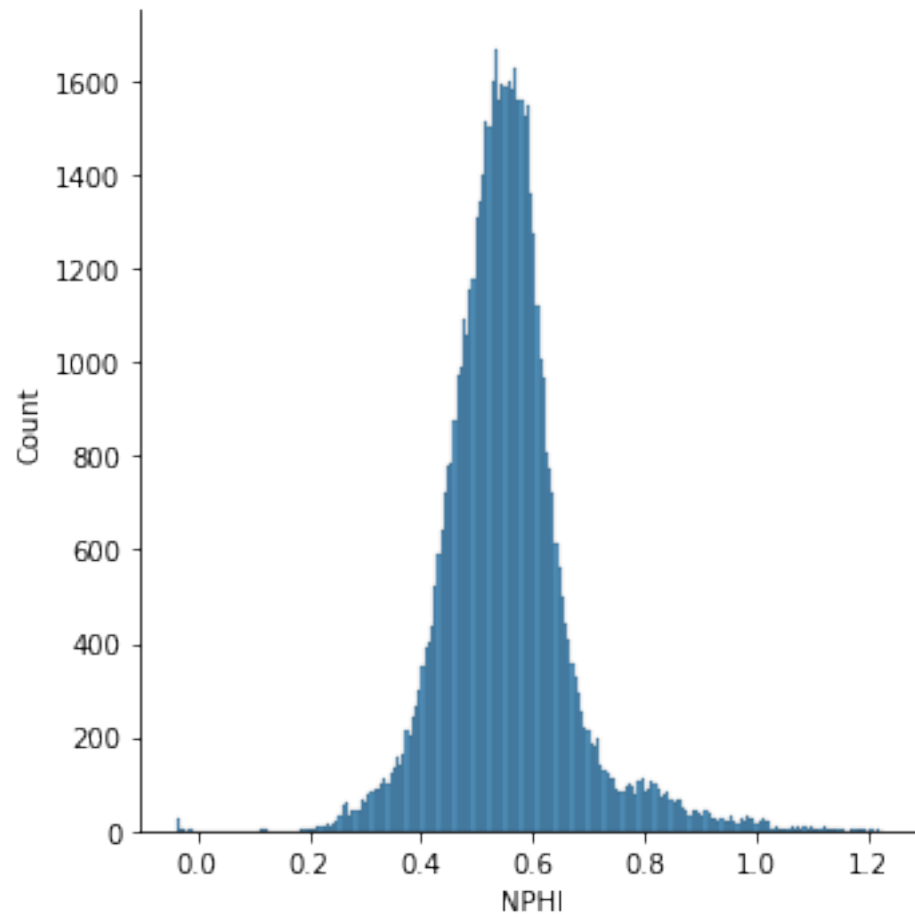
```
[44]:
```





```
[45]: sns.displot(df.NPHI.dropna())
```

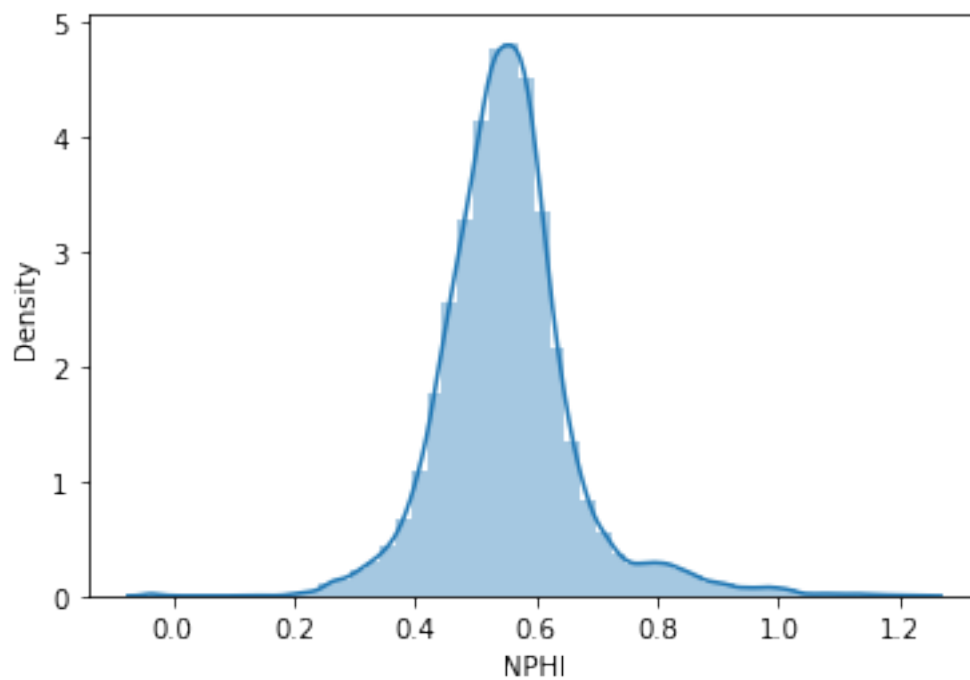
```
[45]: <seaborn.axisgrid.FacetGrid at 0x7fa30472c070>
```



```
[46]: print("WHEN NPHI WAS NULL")  
      null_nphi.figure
```

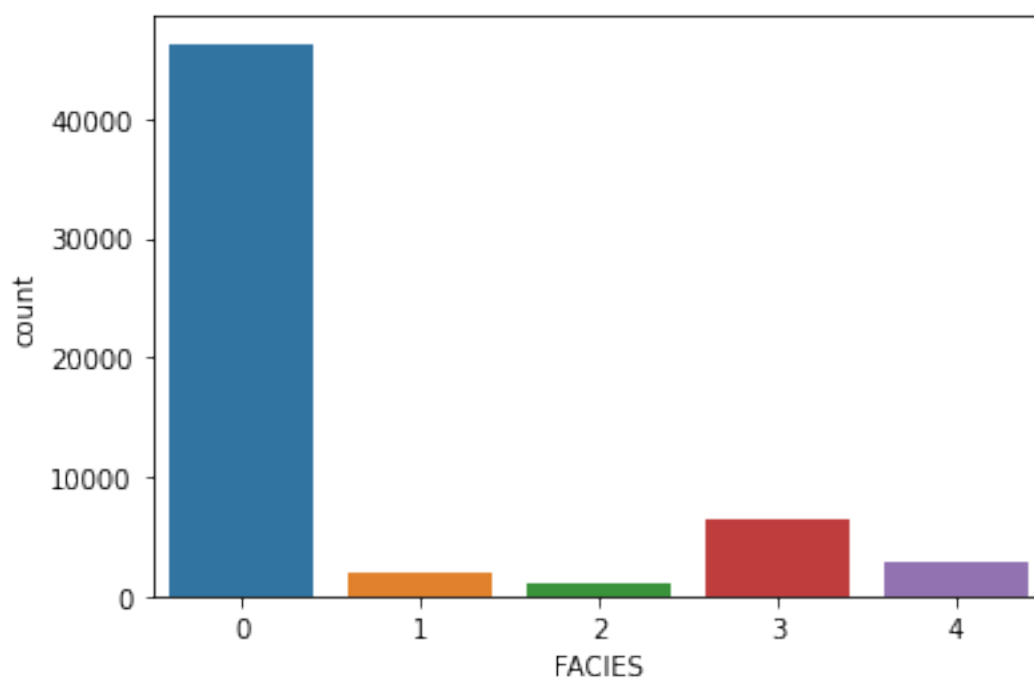
WHEN NPHI WAS NULL

```
[46]:
```



```
[47]: sns.countplot(x="FACIES",data=df)
```

```
[47]: <AxesSubplot:xlabel='FACIES', ylabel='count'>
```



## 4 DATA CONDITIONING / OUTLIER REMOVAL

```
[48]: df.head
```

```
[48]: <bound method NDFrame.head of
FACIES
0      50.2544   50.212800   0.5340   2.1228      2
1      50.3881   49.750900   0.5316   2.1250      3
2      49.8852   48.251300   0.5126   2.1316      3
3      49.9032   46.821200   0.5137   2.1437      3
4      50.0157   45.346300   0.5472   2.1611      3
...
58494  123.7404  130.872833   0.4993   2.4639      0
58495  123.8728   92.579667   0.5313   2.4660      0
58496  123.3722   81.624267   0.5448   2.4714      0
58497  122.6038  118.991767   0.5364   2.4750      0
58498  122.3045   70.033400   0.5331   2.4709      0

[58499 rows x 5 columns]>
```

### 4.1 WHOLE DATA OUTLIER VISUALIZATION

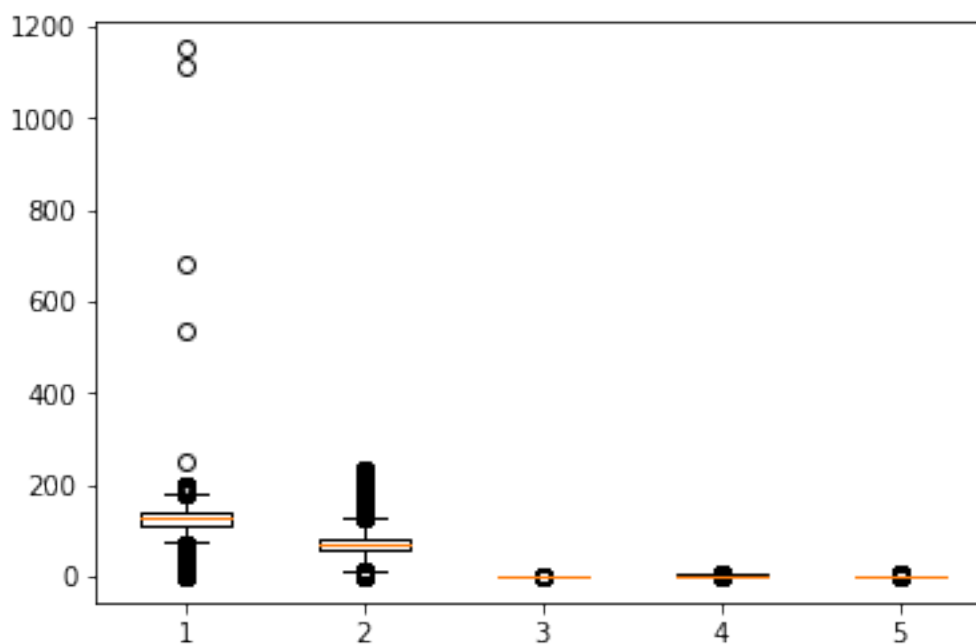
```
[49]: plt.boxplot(df)
```

```
[49]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa30411d220>,
<matplotlib.lines.Line2D at 0x7fa30411d5b0>,
<matplotlib.lines.Line2D at 0x7fa304128b80>,
<matplotlib.lines.Line2D at 0x7fa304128f10>,
<matplotlib.lines.Line2D at 0x7fa3041424f0>,
<matplotlib.lines.Line2D at 0x7fa304142880>,
<matplotlib.lines.Line2D at 0x7fa30414ae20>,
<matplotlib.lines.Line2D at 0x7fa3040951f0>,
<matplotlib.lines.Line2D at 0x7fa3040a0790>,
<matplotlib.lines.Line2D at 0x7fa3040a0b20>],
'caps': [<matplotlib.lines.Line2D at 0x7fa30411d940>,
<matplotlib.lines.Line2D at 0x7fa30411dcd0>,
<matplotlib.lines.Line2D at 0x7fa3041362e0>,
<matplotlib.lines.Line2D at 0x7fa304136670>,
<matplotlib.lines.Line2D at 0x7fa304142c10>,
<matplotlib.lines.Line2D at 0x7fa304142fa0>,
<matplotlib.lines.Line2D at 0x7fa304095580>,
<matplotlib.lines.Line2D at 0x7fa304095910>,
<matplotlib.lines.Line2D at 0x7fa3040a0eb0>,
<matplotlib.lines.Line2D at 0x7fa3040ac280>],
'boxes': [<matplotlib.lines.Line2D at 0x7fa30418ee50>,
```

```

<matplotlib.lines.Line2D at 0x7fa3041287f0>,
<matplotlib.lines.Line2D at 0x7fa304142160>,
<matplotlib.lines.Line2D at 0x7fa30414aa90>,
<matplotlib.lines.Line2D at 0x7fa3040a0400>],
'medians': [<matplotlib.lines.Line2D at 0x7fa3041280a0>,
<matplotlib.lines.Line2D at 0x7fa304136a00>,
<matplotlib.lines.Line2D at 0x7fa30414a370>,
<matplotlib.lines.Line2D at 0x7fa304095ca0>,
<matplotlib.lines.Line2D at 0x7fa3040ac610>],
'fliers': [<matplotlib.lines.Line2D at 0x7fa304128430>,
<matplotlib.lines.Line2D at 0x7fa304136d90>,
<matplotlib.lines.Line2D at 0x7fa30414a700>,
<matplotlib.lines.Line2D at 0x7fa3040a0070>,
<matplotlib.lines.Line2D at 0x7fa3040ac9d0>],
'means': []}

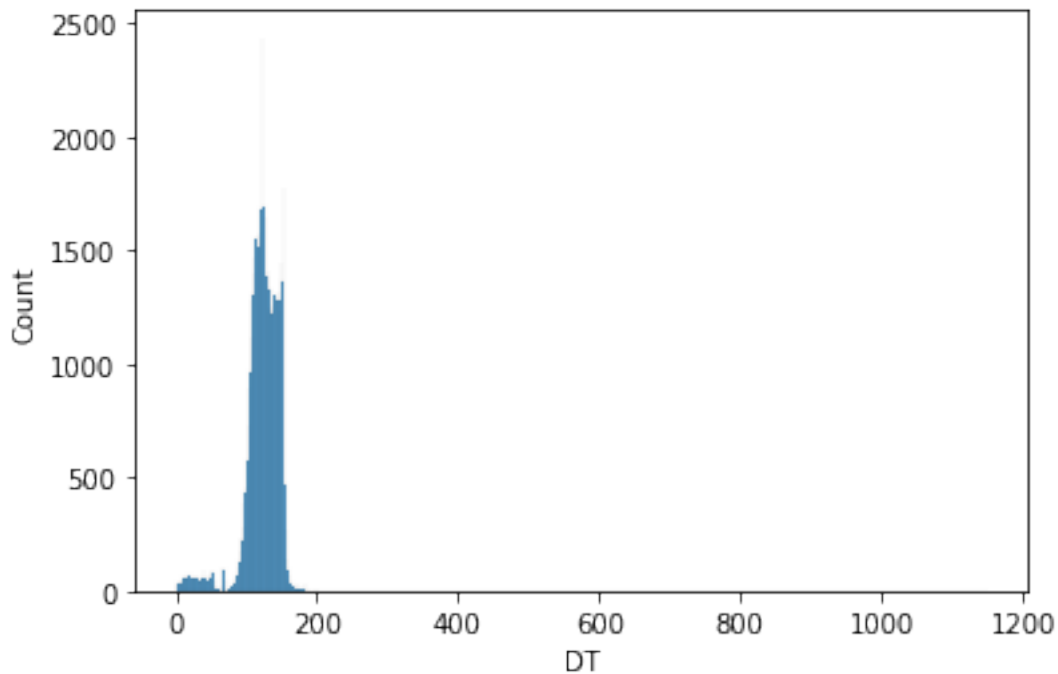
```



## 4.2 DT VISUALIZATION

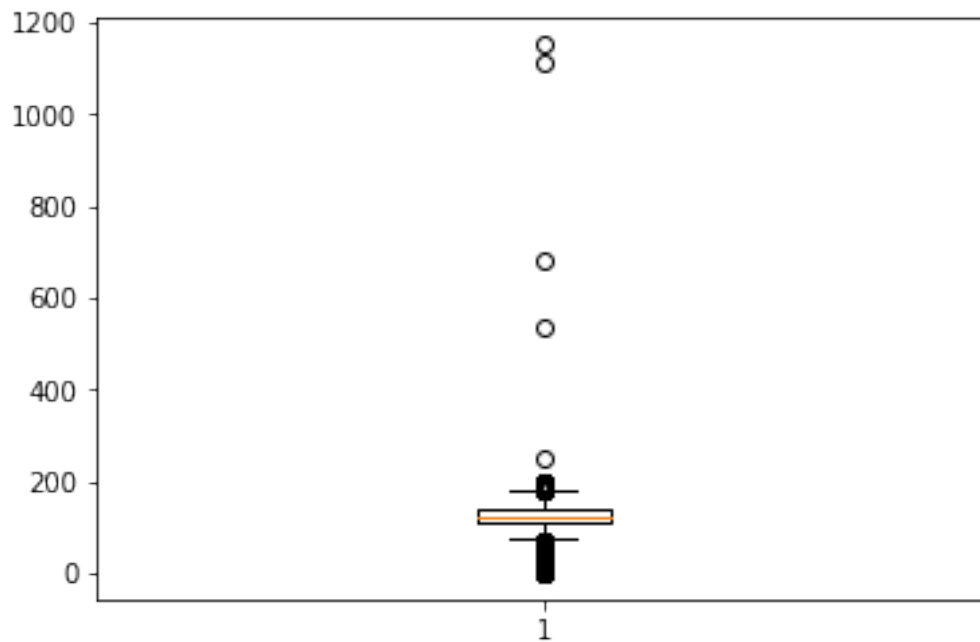
```
[50]: sns.histplot(df.DT)
```

```
[50]: <AxesSubplot:xlabel='DT', ylabel='Count'>
```



```
[51]: plt.boxplot(df["DT"])
```

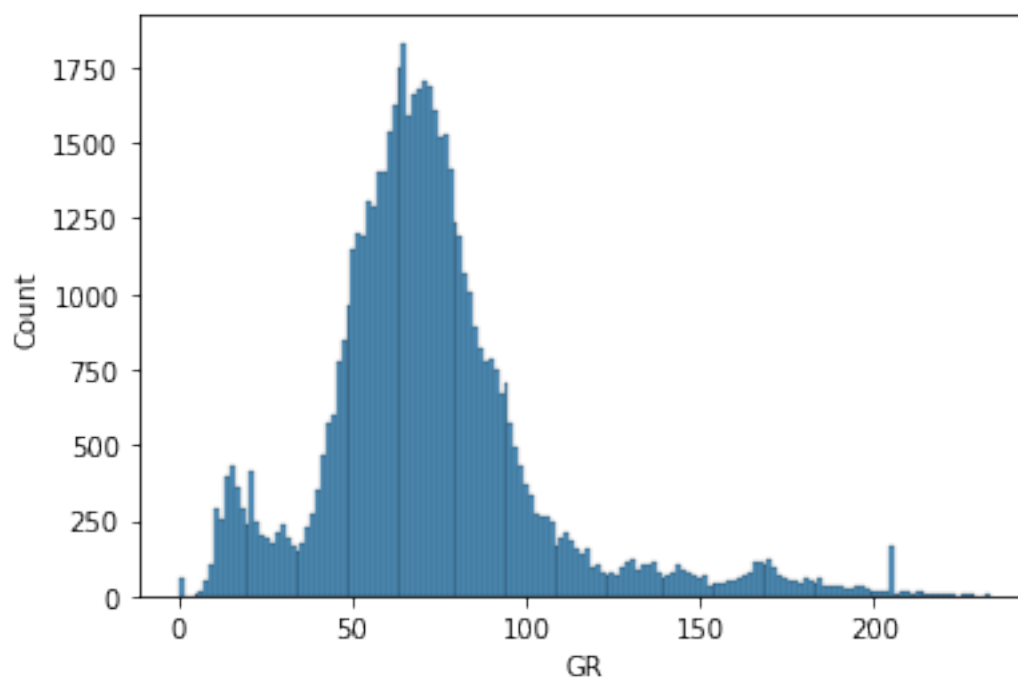
```
[51]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa3028bf1c0>,
<matplotlib.lines.Line2D at 0x7fa3028bf550>],
'caps': [<matplotlib.lines.Line2D at 0x7fa3028bf8e0>,
<matplotlib.lines.Line2D at 0x7fa3028bfc70>],
'boxes': [<matplotlib.lines.Line2D at 0x7fa3028b4df0>],
'medians': [<matplotlib.lines.Line2D at 0x7fa3028c8040>],
'fliers': [<matplotlib.lines.Line2D at 0x7fa3028c83d0>],
'means': []}
```



### 4.3 GR VISUALIZATION

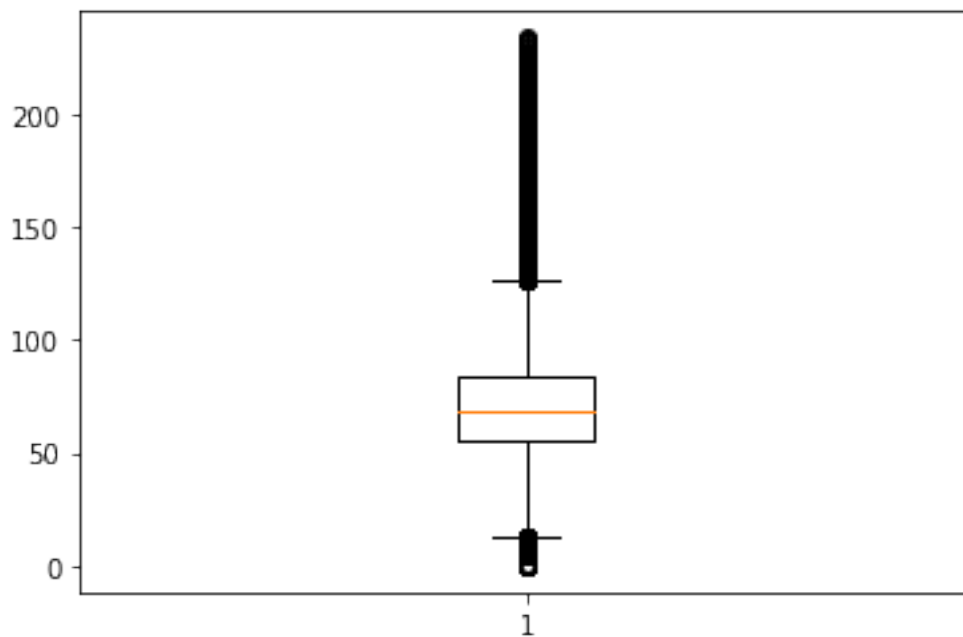
```
[52]: sns.histplot(df.GR)
```

```
[52]: <AxesSubplot:xlabel='GR', ylabel='Count'>
```



```
[53]: plt.boxplot(df.GR)
```

```
[53]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f62374f0>,  
                  <matplotlib.lines.Line2D at 0x7fa2f6237880>],  
       'caps': [<matplotlib.lines.Line2D at 0x7fa2f6237c10>,  
                <matplotlib.lines.Line2D at 0x7fa2f6237fa0>],  
       'boxes': [<matplotlib.lines.Line2D at 0x7fa2f6237160>],  
       'medians': [<matplotlib.lines.Line2D at 0x7fa2f6242370>],  
       'fliers': [<matplotlib.lines.Line2D at 0x7fa2f6242700>],  
       'means': []}
```

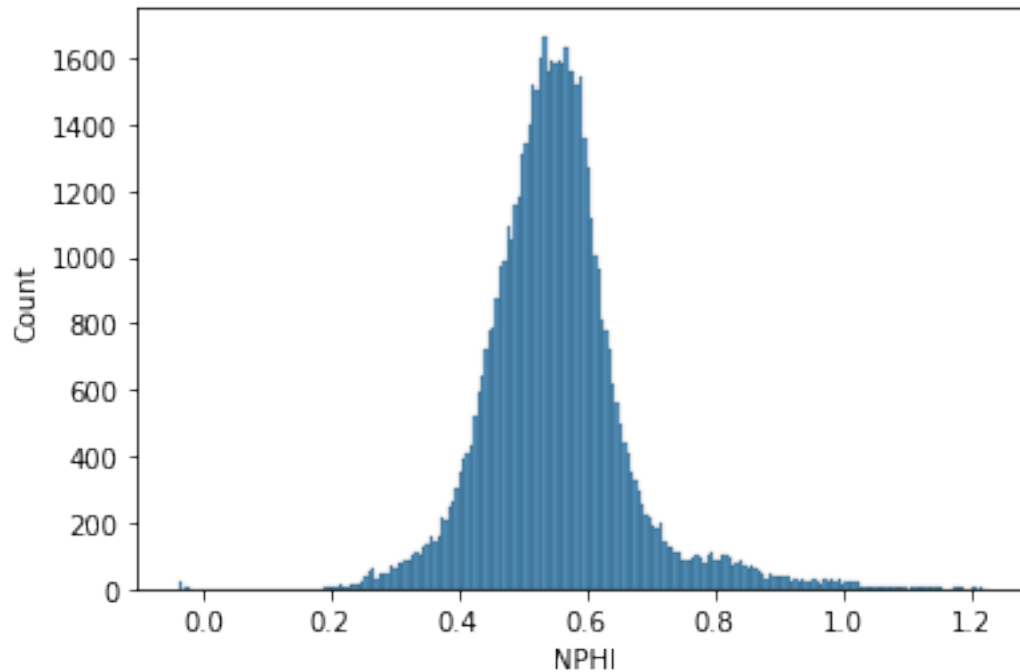


#### 4.4 NPHI VISUALIZATION

```
[54]: sns.histplot(df.NPHI)
```

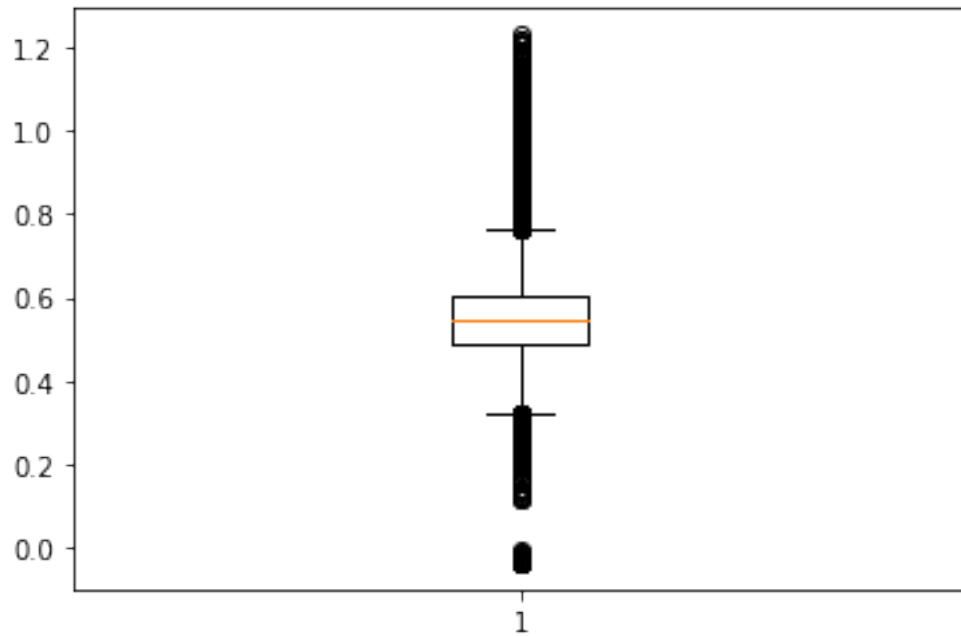
```
[54]: <AxesSubplot:xlabel='NPHI', ylabel='Count'>
```





```
[55]: plt.boxplot(df.NPHI)
```

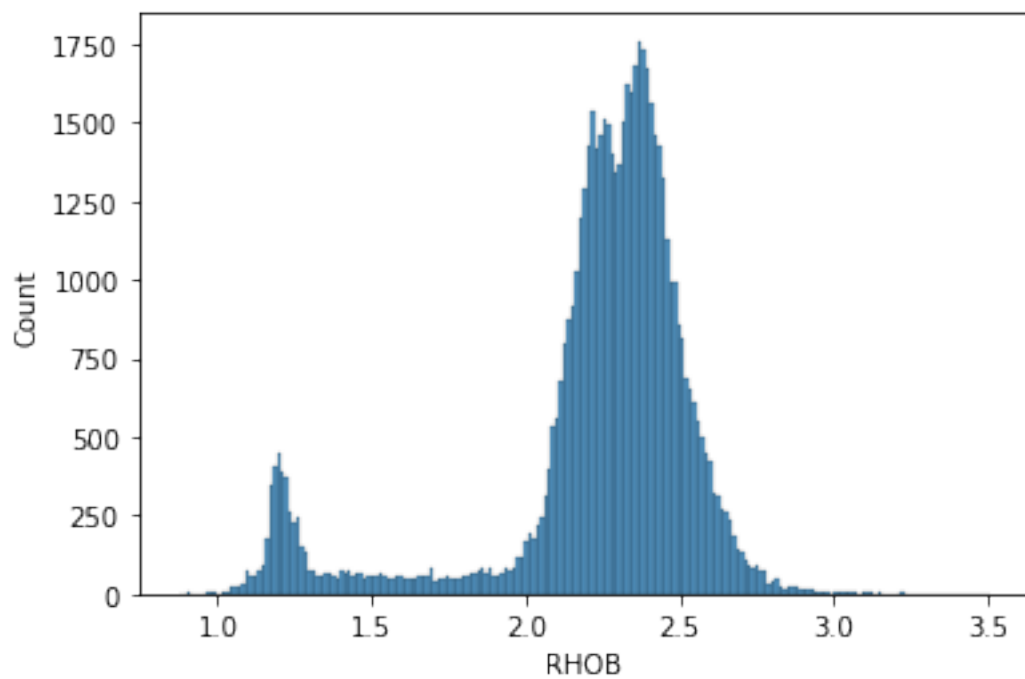
```
[55]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f5f1b430>,
<matplotlib.lines.Line2D at 0x7fa2f5f1b7c0>],
'caps': [<matplotlib.lines.Line2D at 0x7fa2f5f1bb50>,
<matplotlib.lines.Line2D at 0x7fa2f5f1bee0>],
'boxes': [<matplotlib.lines.Line2D at 0x7fa2f5f1b0a0>],
'medians': [<matplotlib.lines.Line2D at 0x7fa2f5f252b0>],
'fliers': [<matplotlib.lines.Line2D at 0x7fa2f5f25640>],
'means': []}
```



## 4.5 RHOB VISUALIZATION

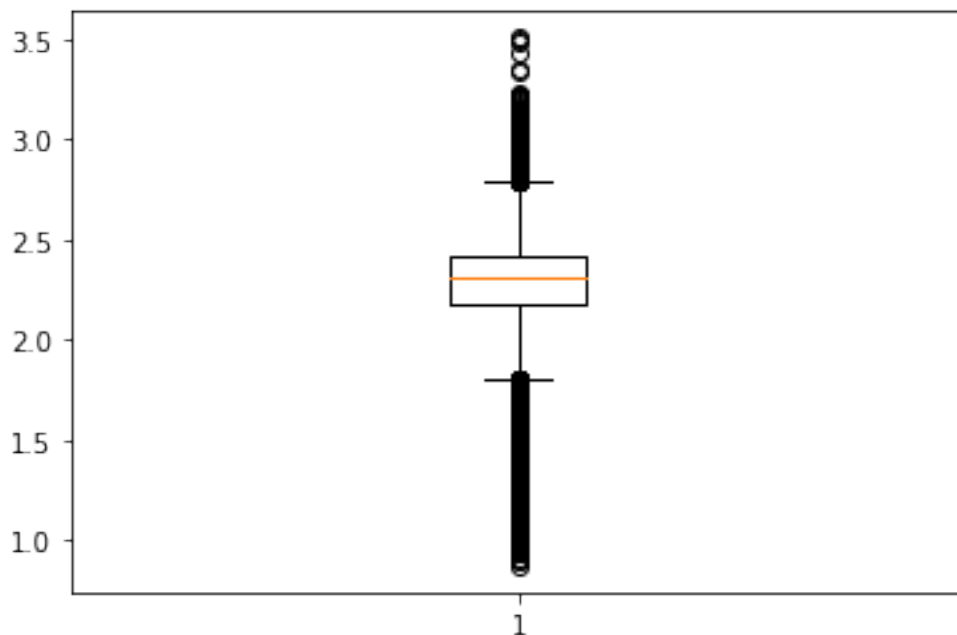
```
[56]: sns.histplot(df.RHOB)
```

```
[56]: <AxesSubplot:xlabel='RHOB', ylabel='Count'>
```



```
[57]: plt.boxplot(df.RHOB)
```

```
[57]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f5c59490>,  
                 <matplotlib.lines.Line2D at 0x7fa2f5c59820>],  
       'caps': [<matplotlib.lines.Line2D at 0x7fa2f5c59bb0>,  
               <matplotlib.lines.Line2D at 0x7fa2f5c59f40>],  
       'boxes': [<matplotlib.lines.Line2D at 0x7fa2f5c59100>],  
       'medians': [<matplotlib.lines.Line2D at 0x7fa2f5c63310>],  
       'fliers': [<matplotlib.lines.Line2D at 0x7fa2f5c636a0>],  
       'means': []}
```



```
[58]: def outliers(dataConditioningStrategy,dataframe, dataconditioningcolumns):  
    df=dataframe  
    if dataConditioningStrategy == "3_Standard_Deviation":  
        for column in dataconditioningcolumns:  
            print("column",column )  
            upperlimit = df[column].mean() + 3*df[column].std()  
            lowerlimit = df[column].mean() - 3*df[column].std()  
  
            print("3 standard deviation outliers -:")  
            print(df[(df[column] > upperlimit) | (df[column] < lowerlimit)])  
            print(df[(df[column] > upperlimit) | (df[column] < lowerlimit)].  
                  ↪shape)
```

```

        df= df[(df[column] < upperlimit) & (df[column] > lowerlimit) & (df.
↪FACIES >= 0) & (df.FACIES <= 4)]
        print(df)

    elif dataConditioningStrategy == "4_Standard_Deviation":
        for column in dataconditioningcolumns:
            print("column",column )
            upperlimit = df[column].mean() + 4*df[column].std()
            lowerlimit = df[column].mean() - 4*df[column].std()

            print("4 standard deviation outliers -:")
            print(df[(df[column] > upperlimit) | (df[column] < lowerlimit)])
            print(df[(df[column] > upperlimit) | (df[column] < lowerlimit)].
↪shape)

            df= df[(df[column] < upperlimit) & (df[column] > lowerlimit) & (df.
↪FACIES >= 0) & (df.FACIES <= 4)]
            print(df)

        elif dataConditioningStrategy == "InterquartileRange":
            for column in dataconditioningcolumns:
                print("column",column )
                q25, q75 = percentile(df[column], 25), percentile(df[column], 75)
                iqr = q75 - q25
                print('Percentiles: 25th=%.3f, 75th=%.3f, IQR=%.3f' % (q25, q75,
↪iqr))

                cut_off = iqr * 1.5
                lowerlimit, upperlimit = q25 - cut_off, q75 + cut_off

                print("InterQuartile Range Outliers-:")
                print(df[(df[column] > upperlimit) | (df[column] < lowerlimit)])
                print(df[(df[column] > upperlimit) | (df[column] < lowerlimit)].
↪shape)

                df= df[(df[column] < upperlimit) & (df[column] > lowerlimit) & (df.
↪FACIES >= 0) & (df.FACIES <= 4)]
                print(df)

    return df

```

```

[59]: DATAConditioningStrategy =
↪["3_Standard_Deviation","4_Standard_Deviation","InterquartileRange"]
DATAConditioningColumns = ["DT","GR","NPFI","RHOB"]
optionoutlier = 2
df = outliers(DATAConditioningStrategy[optionoutlier] , df,
↪DATAConditioningColumns)

```

column DT

Percentiles: 25th=112.649, 75th=139.678, IQR=27.029

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
0	50.2544	50.2128	0.5340	2.1228	2
1	50.3881	49.7509	0.5316	2.1250	3
2	49.8852	48.2513	0.5126	2.1316	3
3	49.9032	46.8212	0.5137	2.1437	3
4	50.0157	45.3463	0.5472	2.1611	3
...	...	...	...	...	...
44912	71.0756	39.1722	0.3397	3.1791	0
44913	71.3734	39.3511	0.3455	2.9875	0
44929	71.2182	55.9609	0.4199	2.7743	0
44930	70.1539	52.4927	0.3936	2.9376	0
44931	67.9970	48.9224	0.3727	3.0912	0

[2592 rows x 5 columns]

(2592, 5)

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.663200	0.4526	2.4314	0
219	76.1991	47.016400	0.4514	2.4413	0
2026	76.4115	48.396700	0.5571	1.0846	0
2027	78.0536	47.637300	0.5496	1.1340	0
2028	75.2216	48.504000	0.5402	1.1749	0
...	...	...	...	...	...
58494	123.7404	130.872833	0.4993	2.4639	0
58495	123.8728	92.579667	0.5313	2.4660	0
58496	123.3722	81.624267	0.5448	2.4714	0
58497	122.6038	118.991767	0.5364	2.4750	0
58498	122.3045	70.033400	0.5331	2.4709	0

[55907 rows x 5 columns]

column GR

Percentiles: 25th=55.447, 75th=84.382, IQR=28.935

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
4029	151.3950	11.621800	0.8730	1.1941	3
4030	151.2614	11.706100	0.8996	1.2056	3
4039	152.8249	11.756300	0.7718	1.1963	3
4040	152.8680	11.590300	0.7690	1.1947	3
9974	141.8031	153.032300	0.6077	2.1543	1
...	...	...	...	...	...
58180	110.9551	130.679400	0.5160	2.3705	0
58181	114.0812	131.847300	0.4959	2.3630	0
58182	115.8771	127.830000	0.4907	2.3684	0
58493	123.4216	159.013200	0.4626	2.4620	0
58494	123.7404	130.872833	0.4993	2.4639	0

[4130 rows x 5 columns]

(4130, 5)

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.663200	0.4526	2.4314	0
219	76.1991	47.016400	0.4514	2.4413	0
2026	76.4115	48.396700	0.5571	1.0846	0
2027	78.0536	47.637300	0.5496	1.1340	0
2028	75.2216	48.504000	0.5402	1.1749	0
...	...	...	...	...	...
58492	123.1318	89.947733	0.4492	2.4574	0
58495	123.8728	92.579667	0.5313	2.4660	0
58496	123.3722	81.624267	0.5448	2.4714	0
58497	122.6038	118.991767	0.5364	2.4750	0
58498	122.3045	70.033400	0.5331	2.4709	0

[51777 rows x 5 columns]

column NPHI

Percentiles: 25th=0.491, 75th=0.595, IQR=0.104

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
2361	151.4359	44.2168	0.7608	1.3596	3
2362	149.7643	36.0403	0.7885	1.2673	3
2363	149.5450	28.3286	0.7905	1.2324	3
2364	150.3661	22.7745	0.7713	1.2522	3
3039	143.1059	35.7501	0.7585	1.2089	3
...	...	...	...	...	...
54941	114.6628	114.2647	0.3314	2.2033	1
54953	109.6688	104.5008	0.3327	2.2693	1
57287	106.0689	97.8201	0.3325	2.2712	4
57981	150.8674	23.8442	0.7894	1.1197	3
58290	152.3449	17.4243	0.7641	1.1663	3

[2890 rows x 5 columns]

(2890, 5)

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.663200	0.4526	2.4314	0
219	76.1991	47.016400	0.4514	2.4413	0
2026	76.4115	48.396700	0.5571	1.0846	0
2027	78.0536	47.637300	0.5496	1.1340	0
2028	75.2216	48.504000	0.5402	1.1749	0
...	...	...	...	...	...
58492	123.1318	89.947733	0.4492	2.4574	0
58495	123.8728	92.579667	0.5313	2.4660	0
58496	123.3722	81.624267	0.5448	2.4714	0
58497	122.6038	118.991767	0.5364	2.4750	0
58498	122.3045	70.033400	0.5331	2.4709	0

[48887 rows x 5 columns]

column RHOB

Percentiles: 25th=2.205, 75th=2.429, IQR=0.224

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
2026	76.4115	48.396700	0.5571	1.0846	0
2027	78.0536	47.637300	0.5496	1.1340	0
2028	75.2216	48.504000	0.5402	1.1749	0
2359	153.0665	56.560300	0.6720	1.6982	3
2360	153.1740	51.458100	0.7148	1.5046	3
...	...	...	...	...	...
58400	94.9099	59.051400	0.4770	2.9270	0
58401	96.4064	57.049500	0.4644	2.7995	0
58477	100.5518	58.763333	0.4365	2.7816	0
58478	92.1297	55.115400	0.4509	2.8974	0
58479	92.0023	58.263400	0.4585	2.7846	0

[3505 rows x 5 columns]

(3505, 5)

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.663200	0.4526	2.4314	0
219	76.1991	47.016400	0.4514	2.4413	0
2250	137.8066	61.327800	0.5643	2.1857	0
2251	139.5873	61.995400	0.5611	2.1762	0
2252	140.0185	63.518800	0.5630	2.1946	0
...	...	...	...	...	...
58492	123.1318	89.947733	0.4492	2.4574	0
58495	123.8728	92.579667	0.5313	2.4660	0
58496	123.3722	81.624267	0.5448	2.4714	0
58497	122.6038	118.991767	0.5364	2.4750	0
58498	122.3045	70.033400	0.5331	2.4709	0

[45382 rows x 5 columns]

```
[60]: df.shape
```

```
[60]: (45382, 5)
```

## 4.6 WHOLE DATA AFTER REMOVING OUTLIERS

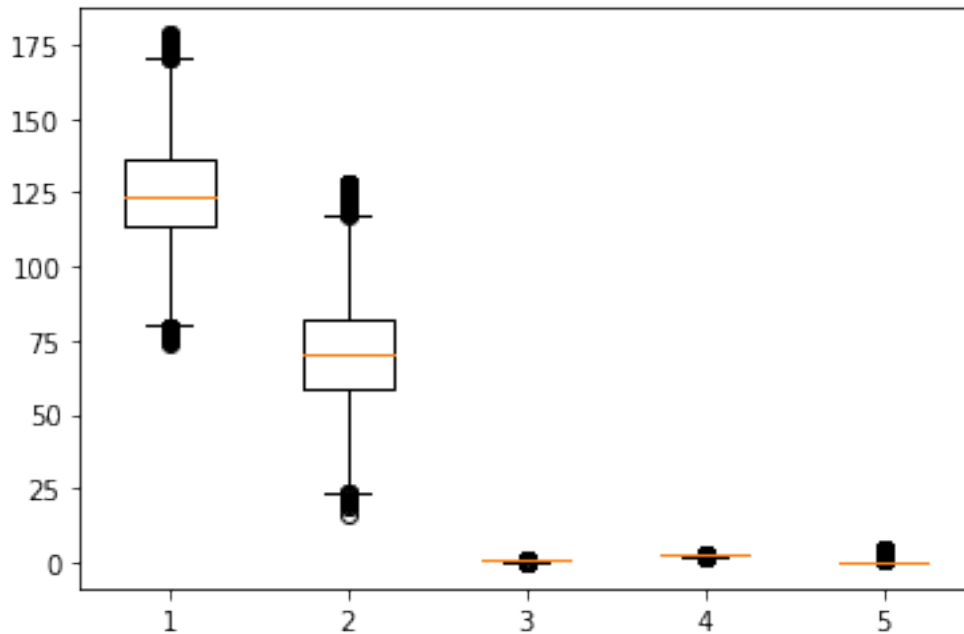
```
[61]: plt.boxplot(df)
```

```
[61]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f5c3fd00>,  
  <matplotlib.lines.Line2D at 0x7fa2f580c0d0>,  
  <matplotlib.lines.Line2D at 0x7fa2f58186a0>,  
  <matplotlib.lines.Line2D at 0x7fa2f5818a30>,  
  <matplotlib.lines.Line2D at 0x7fa2f5824fd0>,  
  <matplotlib.lines.Line2D at 0x7fa2f582f3a0>,  
  <matplotlib.lines.Line2D at 0x7fa2f5839940>,  
  <matplotlib.lines.Line2D at 0x7fa2f5839cd0>],
```

```

<matplotlib.lines.Line2D at 0x7fa2f5bcf2b0>,
<matplotlib.lines.Line2D at 0x7fa2f5bcf640>],
'caps': [<matplotlib.lines.Line2D at 0x7fa2f580c490>,
<matplotlib.lines.Line2D at 0x7fa2f580c820>,
<matplotlib.lines.Line2D at 0x7fa2f5818dc0>,
<matplotlib.lines.Line2D at 0x7fa2f5824190>,
<matplotlib.lines.Line2D at 0x7fa2f582f730>,
<matplotlib.lines.Line2D at 0x7fa2f582fac0>,
<matplotlib.lines.Line2D at 0x7fa2f58440a0>,
<matplotlib.lines.Line2D at 0x7fa2f5844430>,
<matplotlib.lines.Line2D at 0x7fa2f5bcf9d0>,
<matplotlib.lines.Line2D at 0x7fa2f5bcfd60>],
'boxes': [<matplotlib.lines.Line2D at 0x7fa2f5c3f970>,
<matplotlib.lines.Line2D at 0x7fa2f5818310>,
<matplotlib.lines.Line2D at 0x7fa2f5824c40>,
<matplotlib.lines.Line2D at 0x7fa2f58395b0>,
<matplotlib.lines.Line2D at 0x7fa2f5844ee0>],
'medians': [<matplotlib.lines.Line2D at 0x7fa2f580cbb0>,
<matplotlib.lines.Line2D at 0x7fa2f5824520>,
<matplotlib.lines.Line2D at 0x7fa2f582fe50>,
<matplotlib.lines.Line2D at 0x7fa2f58447c0>,
<matplotlib.lines.Line2D at 0x7fa2f5bd9130>],
'fliers': [<matplotlib.lines.Line2D at 0x7fa2f580cf40>,
<matplotlib.lines.Line2D at 0x7fa2f58248b0>,
<matplotlib.lines.Line2D at 0x7fa2f5839220>,
<matplotlib.lines.Line2D at 0x7fa2f5844b50>,
<matplotlib.lines.Line2D at 0x7fa2f5bd94c0>],
'means': []

```





```
[62]: df.head(5)
```

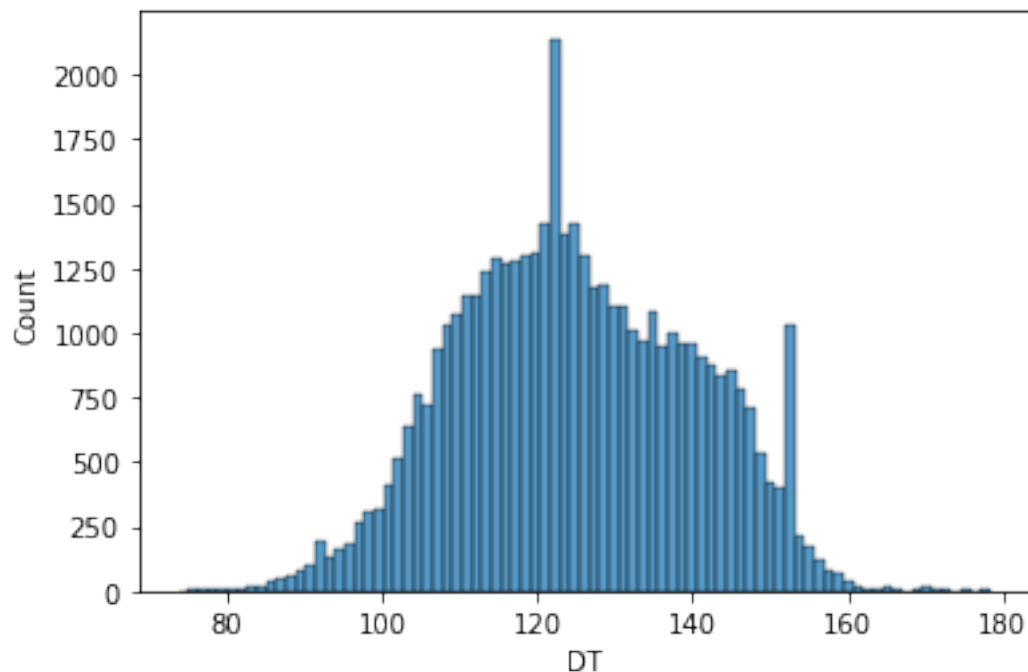
```
[62]:
```

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.6632	0.4526	2.4314	0
219	76.1991	47.0164	0.4514	2.4413	0
2250	137.8066	61.3278	0.5643	2.1857	0
2251	139.5873	61.9954	0.5611	2.1762	0
2252	140.0185	63.5188	0.5630	2.1946	0

## 4.7 DT AFTER REMOVING OUTLIER

```
[63]: sns.histplot(df.DT)
```

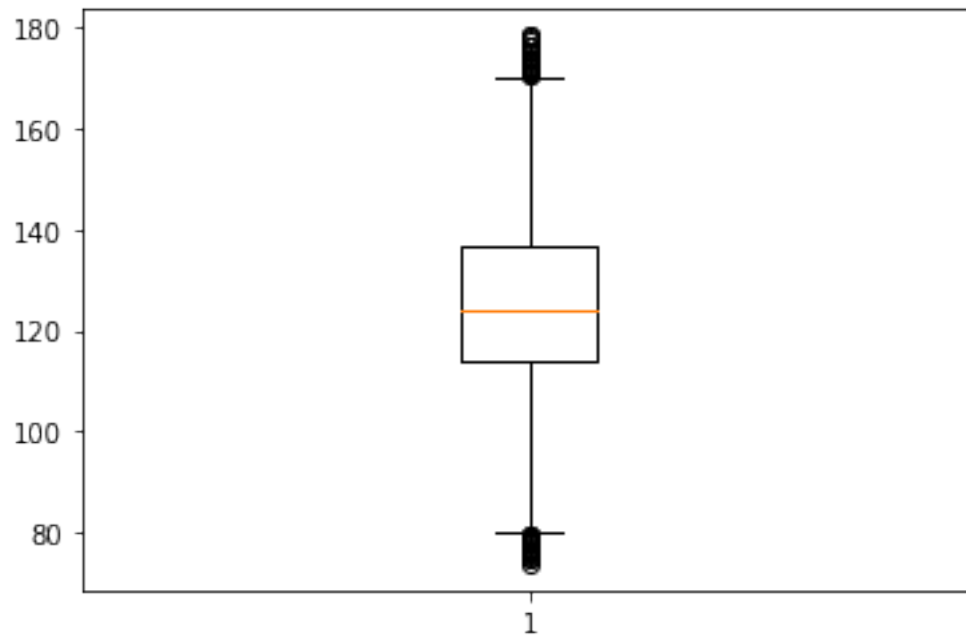
```
[63]: <AxesSubplot:xlabel='DT', ylabel='Count'>
```



```
[64]: plt.boxplot(df["DT"])
```

```
[64]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f5ad4ac0>,  
                 <matplotlib.lines.Line2D at 0x7fa2f5ad4e50>],  
       'caps': [<matplotlib.lines.Line2D at 0x7fa2f5ae1220>,  
                <matplotlib.lines.Line2D at 0x7fa2f5ae15b0>],  
       'boxes': [<matplotlib.lines.Line2D at 0x7fa2f5ad4730>],
```

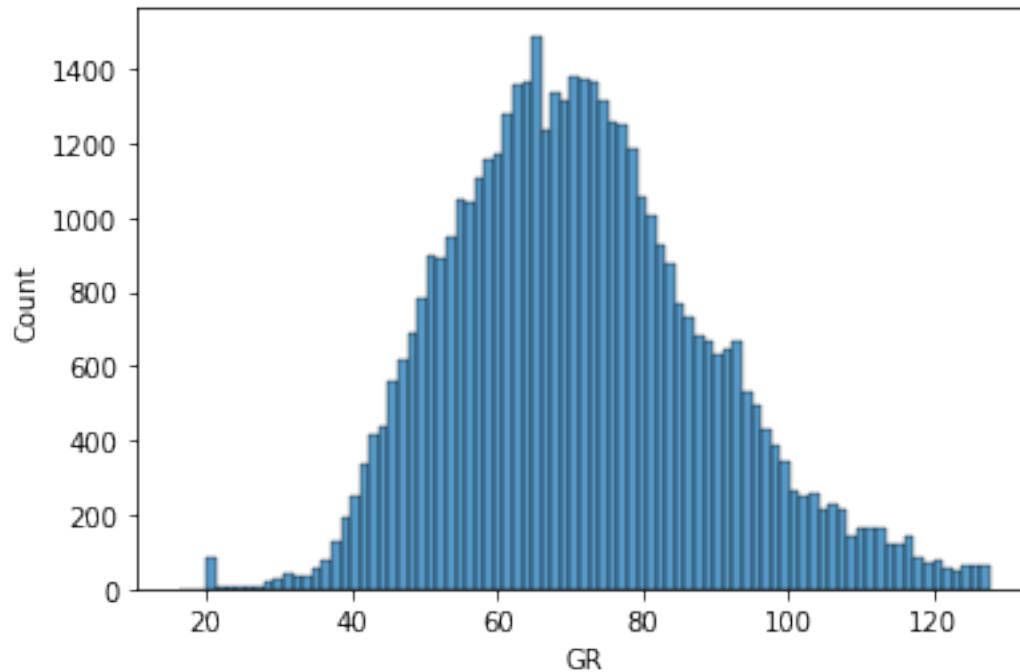
```
'medians': [<matplotlib.lines.Line2D at 0x7fa2f5ae1940>],  
'fliers': [<matplotlib.lines.Line2D at 0x7fa2f5ae1cd0>],  
'means': []}
```



#### 4.8 GR AFTER REMOVING OUTLIER

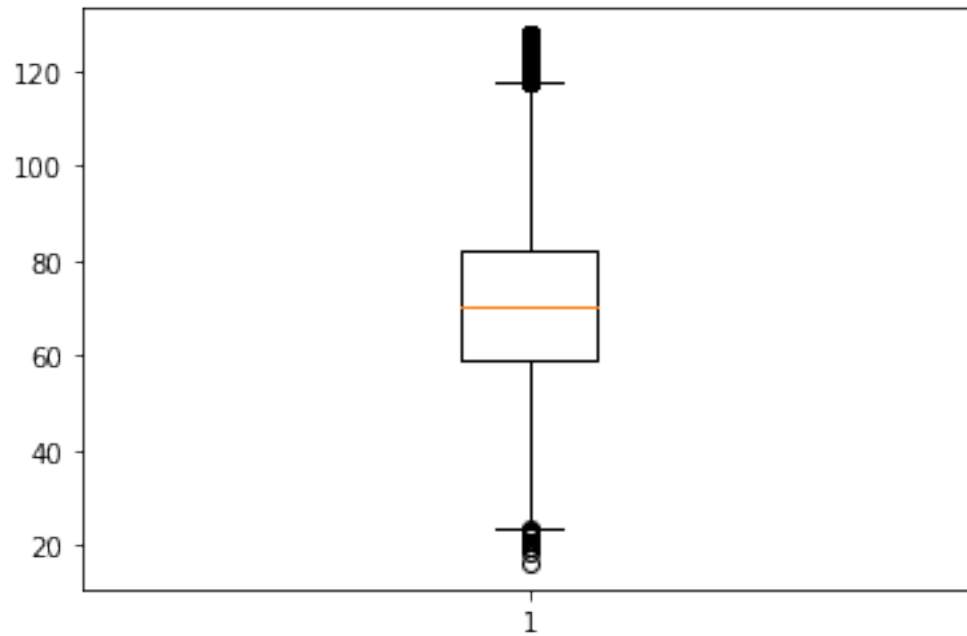
```
[65]: sns.histplot(df.GR)
```

```
[65]: <AxesSubplot:xlabel='GR', ylabel='Count'>
```



```
[66]: plt.boxplot(df.GR)
```

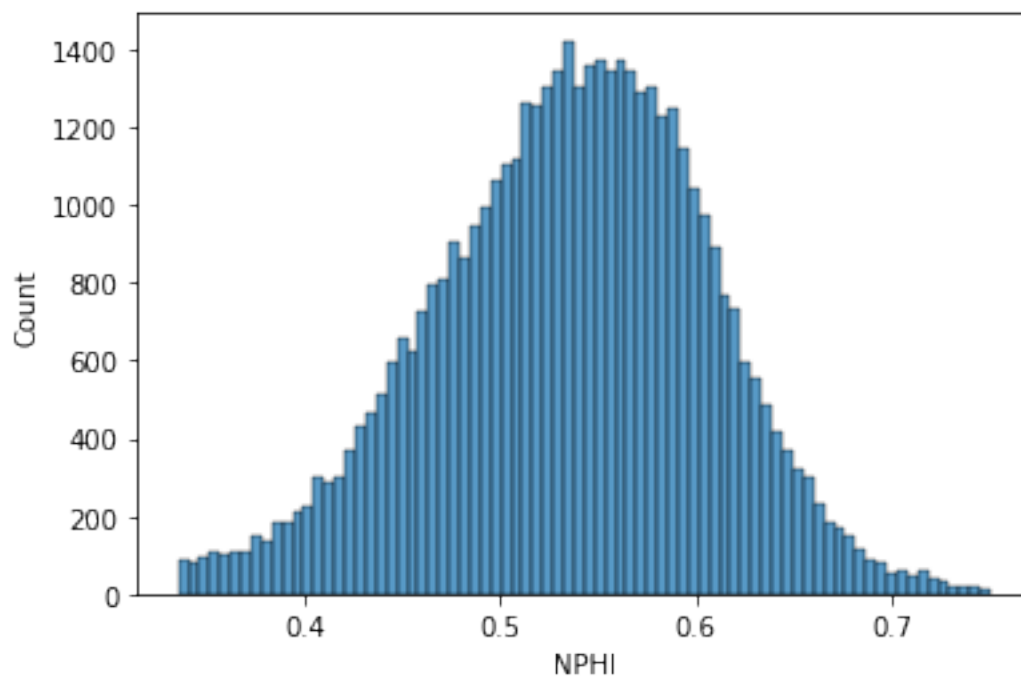
```
[66]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f595b490>,
<matplotlib.lines.Line2D at 0x7fa2f595b820>],
'caps': [<matplotlib.lines.Line2D at 0x7fa2f595bbb0>,
<matplotlib.lines.Line2D at 0x7fa2f595bf40>],
'boxes': [<matplotlib.lines.Line2D at 0x7fa2f595b100>],
'medians': [<matplotlib.lines.Line2D at 0x7fa2f5965310>],
'fliers': [<matplotlib.lines.Line2D at 0x7fa2f59656a0>],
'means': []}
```



#### 4.9 NPHI AFTER REMOVING OUTLIER

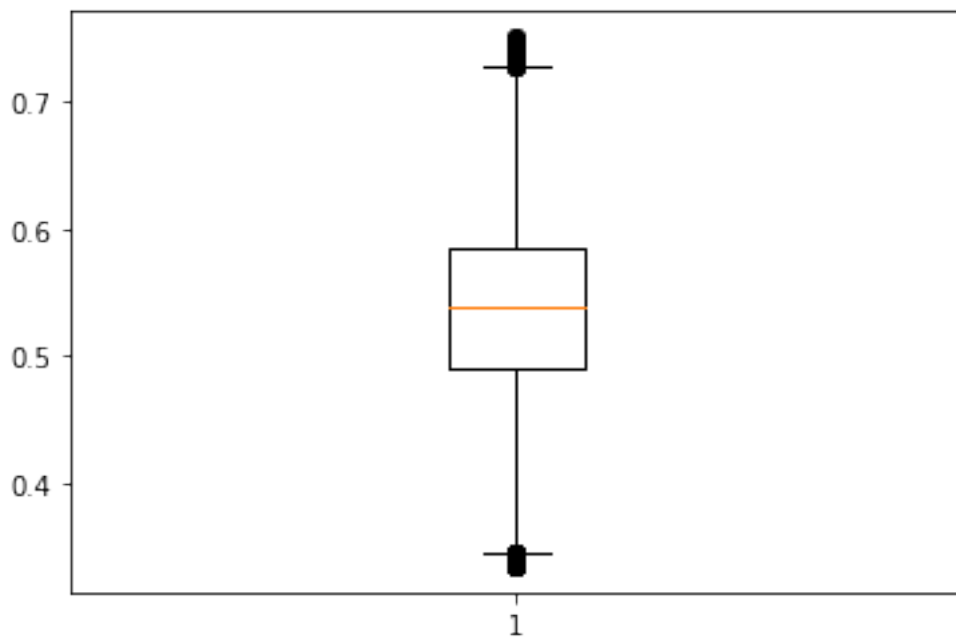
```
[67]: sns.histplot(df.NPHI)
```

```
[67]: <AxesSubplot:xlabel='NPHI', ylabel='Count'>
```



```
[68]: plt.boxplot(df.NPHI)
```

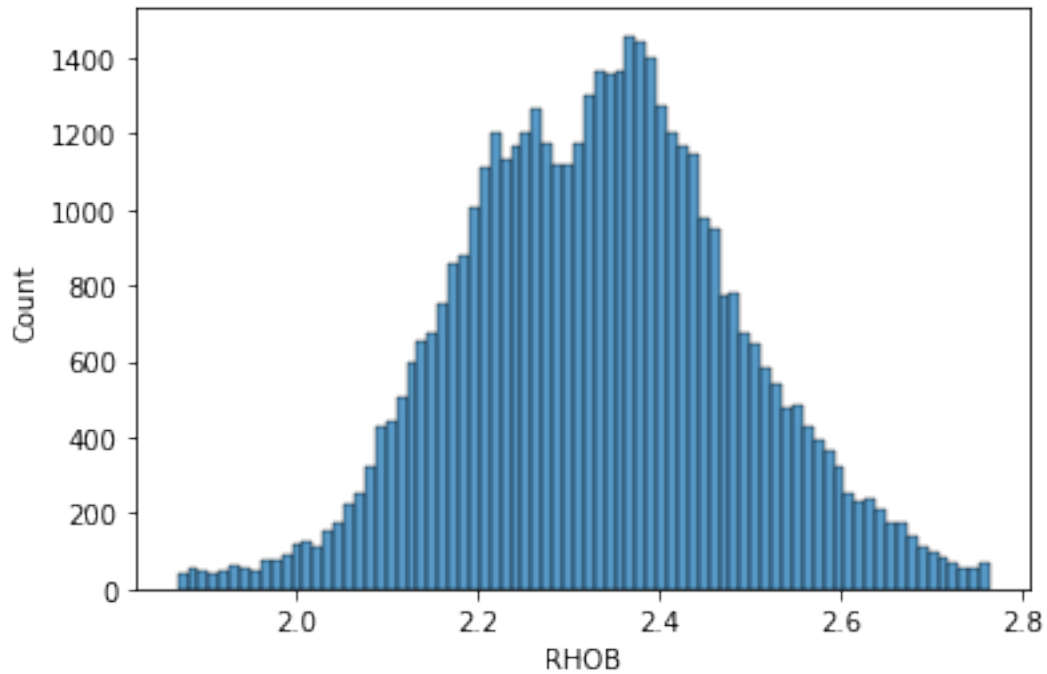
```
[68]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f57fb040>,  
<matplotlib.lines.Line2D at 0x7fa2f57fb3d0>],  
'caps': [<matplotlib.lines.Line2D at 0x7fa2f57fb760>,  
<matplotlib.lines.Line2D at 0x7fa2f57fbaf0>],  
'boxes': [<matplotlib.lines.Line2D at 0x7fa2f57edc70>],  
'medians': [<matplotlib.lines.Line2D at 0x7fa2f57fbe80>],  
'fliers': [<matplotlib.lines.Line2D at 0x7fa2f5805250>],  
'means': []}
```



#### 4.10 RHOB AFTER REMOVING OUTLIER

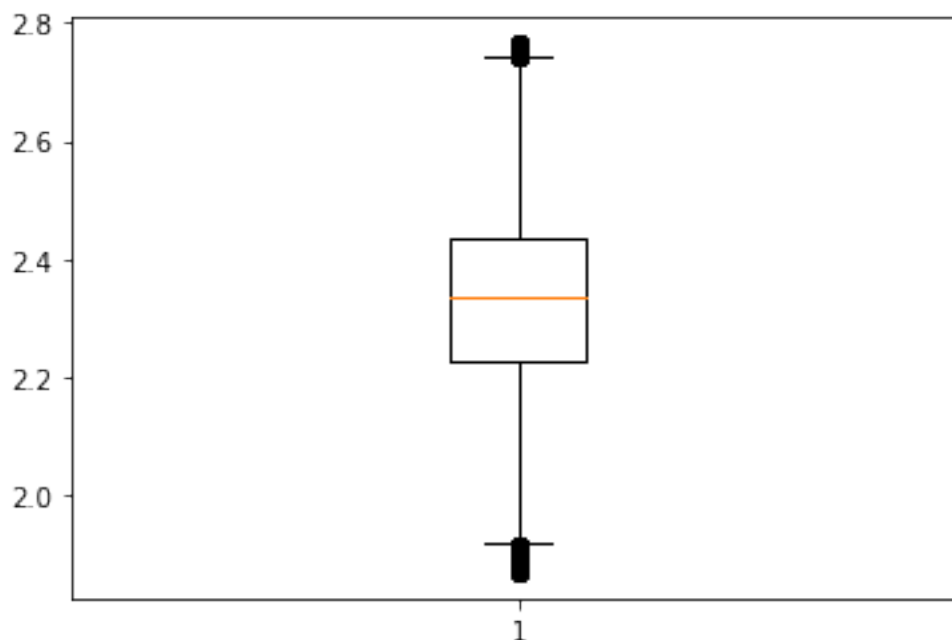
```
[69]: sns.histplot(df.RHOB)
```

```
[69]: <AxesSubplot:xlabel='RHOB', ylabel='Count'>
```



```
[70]: plt.boxplot(df.RHOB)
```

```
[70]: {'whiskers': [<matplotlib.lines.Line2D at 0x7fa2f5619a00>,  
                  <matplotlib.lines.Line2D at 0x7fa2f5619d90>],  
       'caps': [<matplotlib.lines.Line2D at 0x7fa2f5628160>,  
                <matplotlib.lines.Line2D at 0x7fa2f56284f0>],  
       'boxes': [<matplotlib.lines.Line2D at 0x7fa2f5619670>],  
       'medians': [<matplotlib.lines.Line2D at 0x7fa2f5628880>],  
       'fliers': [<matplotlib.lines.Line2D at 0x7fa2f5628c10>],  
       'means': []}
```



```
[71]: df
```

```
[71]:
```

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.663200	0.4526	2.4314	0
219	76.1991	47.016400	0.4514	2.4413	0
2250	137.8066	61.327800	0.5643	2.1857	0
2251	139.5873	61.995400	0.5611	2.1762	0
2252	140.0185	63.518800	0.5630	2.1946	0
...	...	...	...	...	...
58492	123.1318	89.947733	0.4492	2.4574	0
58495	123.8728	92.579667	0.5313	2.4660	0
58496	123.3722	81.624267	0.5448	2.4714	0
58497	122.6038	118.991767	0.5364	2.4750	0
58498	122.3045	70.033400	0.5331	2.4709	0

```
[45382 rows x 5 columns]
```

## 5 FEATURE SELECTION

```
[72]: df.head(10)
```

```
[72]:
```

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.6632	0.4526	2.4314	0
219	76.1991	47.0164	0.4514	2.4413	0
2250	137.8066	61.3278	0.5643	2.1857	0

2251	139.5873	61.9954	0.5611	2.1762	0
2252	140.0185	63.5188	0.5630	2.1946	0
2253	139.3474	64.9925	0.5677	2.1992	0
2254	138.8638	65.6985	0.5743	2.1992	0
2255	139.0847	65.1353	0.5844	2.2009	0
2256	139.2288	63.4583	0.5984	2.2021	0
2257	138.7143	61.7829	0.6146	2.2090	0

```
[73]: df.shape
```

```
[73]: (45382, 5)
```

```
[74]: features = df.shape[1]
features
```

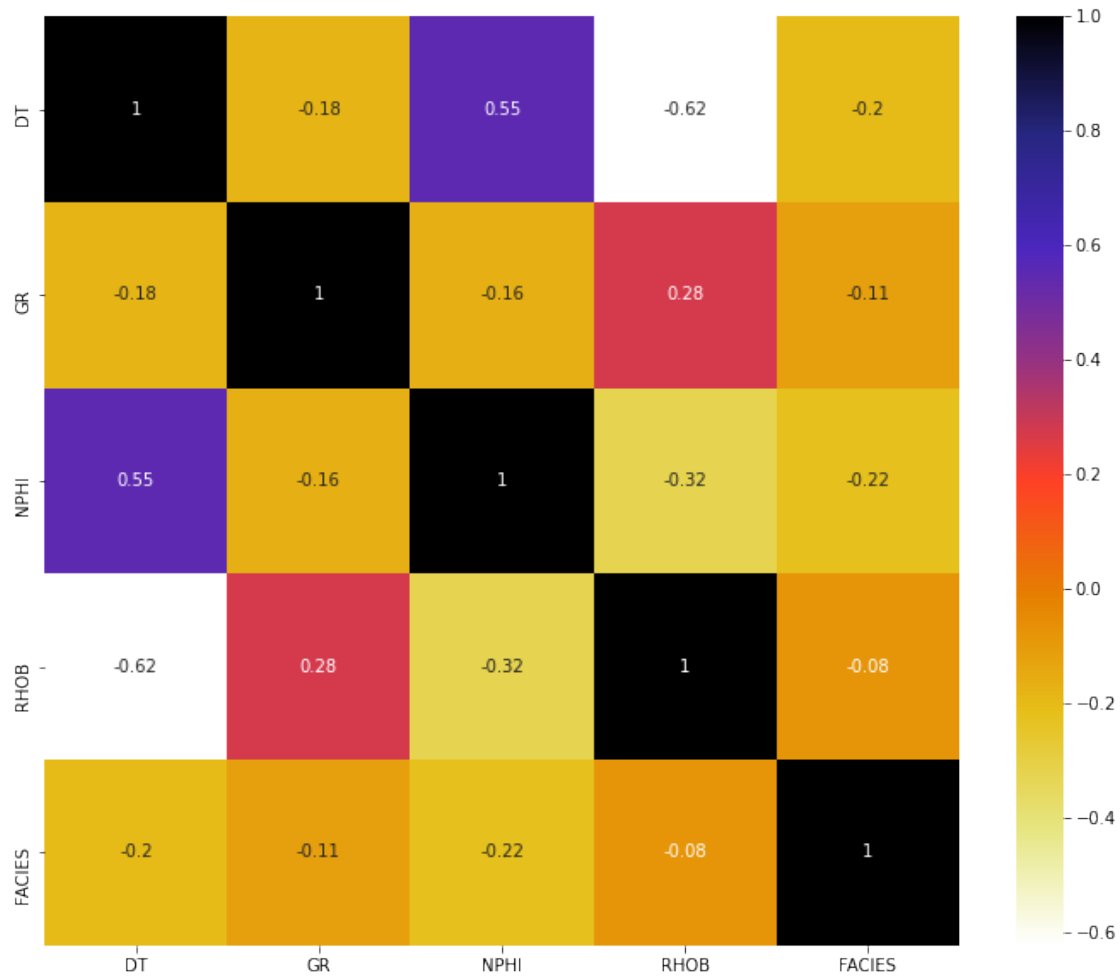
```
[74]: 5
```

```
[75]: df.var()
```

```
[75]: DT          230.983102
GR           313.280377
NPHI           0.004941
RHOB           0.023272
FACIES         1.045660
dtype: float64
```

```
[76]: plt.figure(figsize=(12,10))
cor = df.corr()
sns.heatmap(cor , annot=True , cmap=plt.cm.CMRmap_r)
plt.show()
```





```
[77]: def FeatureSelection(FeatureSelectionStrategy,dataframe):
    df=dataframe

    if(FeatureSelectionStrategy=="Variance_Threshold"):
        var_thres=VarianceThreshold(threshold=0.0)
        var_thres.fit(df)
        df.columns[var_thres.get_support()]
        cols = [column for column in df.columns
                  if column not in df.columns[var_thres.get_support()]]
        print(cols)
        df = df.drop(cols,axis=1)
        return df

    if(FeatureSelectionStrategy=="Absolute_Correlation"):
        threshold = 0.6
        col_corr = set()
```

```

corr_matrix = df.corr()
for i in range(len(corr_matrix.columns)):
    for j in range(i):
        if abs(corr_matrix.iloc[i,j]) > threshold :
            colname = corr_matrix.columns[i]
            print(colname)
            col_corr.add(colname)
df = df.drop(col_corr,axis=1)
return df

if (FeatureSelectionStrategy=="Correlation"):
    threshold = 0.6
    col_corr = set()
    corr_matrix = df.corr()
    for i in range(len(corr_matrix.columns)):
        for j in range(i):
            if (corr_matrix.iloc[i,j]) > threshold :
                colname = corr_matrix.columns[i]
                print(colname)
                col_corr.add(colname)
    df = df.drop(col_corr,axis=1)
    return df

if (FeatureSelectionStrategy == "SelectKBest"):
    x = df.drop("FACIES",1)
    y = df["FACIES"]
    mutual_info = mutual_info_classif(x,y)
    print(mutual_info)
    mutual_info=pd.Series(mutual_info)
    mutual_info.sort_values(ascending=False)
    mutual_info.sort_values(ascending=False).plot.bar(figsize=(20,8))
    select_col = SelectKBest(mutual_info_classif,k=1)
    select_col.fit(x,y)
    column1 = df.columns[select_col.get_support()]
    df = df.drop(column1,axis=1)
    return df

if (FeatureSelectionStrategy == "Mutual_Info_Class"):
    x = df.drop("FACIES",1)
    y = df["FACIES"]
    mutual_info = mutual_info_classif(x,y)
    print(mutual_info)
    mutual_info=pd.Series(mutual_info)
    mutual_info.sort_values(ascending=False)
    mutual_info.sort_values(ascending=False).plot.bar(figsize=(20,8))
    return df

```

```
[78]: FeatureSelectionStrategy=["Variance_Threshold","Absolute_Correlation","Correlation","SelectKBe
optionfeature = 0
df=FeatureSelection(FeatureSelectionStrategy[optionfeature],df)
```

```
[]
```

```
[79]: print("Deleted feature(s) = " + str(features-df.shape[1]))
```

```
Deleted feature(s) = 0
```

```
[80]: df
```

```
[80]:
```

	DT	GR	NPHI	RHOB	FACIES
218	75.8412	47.663200	0.4526	2.4314	0
219	76.1991	47.016400	0.4514	2.4413	0
2250	137.8066	61.327800	0.5643	2.1857	0
2251	139.5873	61.995400	0.5611	2.1762	0
2252	140.0185	63.518800	0.5630	2.1946	0
...	...	...	...	...	...
58492	123.1318	89.947733	0.4492	2.4574	0
58495	123.8728	92.579667	0.5313	2.4660	0
58496	123.3722	81.624267	0.5448	2.4714	0
58497	122.6038	118.991767	0.5364	2.4750	0
58498	122.3045	70.033400	0.5331	2.4709	0

```
[45382 rows x 5 columns]
```

## 6 SCALING DATA

```
[81]: def data_scaling( scaling_strategy , scaling_data , scaling_columns ):

    if scaling_strategy == "RobustScaler" :
        scaling_data[scaling_columns] = RobustScaler().
        ↪fit_transform(scaling_data[scaling_columns])

    elif scaling_strategy == "MinMaxScaler" :
        scaling_data[scaling_columns] = MinMaxScaler().
        ↪fit_transform(scaling_data[scaling_columns])

    else : # If any other scaling send by mistake still perform Robust Scalar
        scaling_data[scaling_columns] = RobustScaler().
        ↪fit_transform(scaling_data[scaling_columns])

    return scaling_data
```

```
[82]: scaling_strategy = ["RobustScaler","MinMaxScaler"]
optionscaling = 0
```

```
df = data_scaling( scaling_strategy[optionscaling] , df ,  
↳DATAConditioningColumns )
```

```
[83]: df
```

```
[83]:
```

	DT	GR	NPHI	RHOB	FACIES
218	-2.124306	-0.959885	-0.908473	0.465409	0
219	-2.108481	-0.987369	-0.921025	0.513304	0
2250	0.615641	-0.379246	0.259937	-0.723270	0
2251	0.694378	-0.350878	0.226464	-0.769231	0
2252	0.713445	-0.286145	0.246339	-0.680213	0
...	...	...	...	...	...
58492	-0.033240	0.836880	-0.944038	0.591195	0
58495	-0.000475	0.948717	-0.085251	0.632801	0
58496	-0.022611	0.483197	0.055962	0.658926	0
58497	-0.056587	2.071027	-0.031904	0.676343	0
58498	-0.069821	-0.009325	-0.066423	0.656507	0

```
[45382 rows x 5 columns]
```

```
[84]: df.to_csv("Preprocessed_data.csv",index=False)
```

## 7 SPLITTING DATA USING TRAIN\_TEST\_SPLIT

```
[85]: df=pd.read_csv('Preprocessed_data.csv')
```

```
[86]: df.head()
```

```
[86]:
```

	DT	GR	NPHI	RHOB	FACIES
0	-2.124306	-0.959885	-0.908473	0.465409	0
1	-2.108481	-0.987369	-0.921025	0.513304	0
2	0.615641	-0.379246	0.259937	-0.723270	0
3	0.694378	-0.350878	0.226464	-0.769231	0
4	0.713445	-0.286145	0.246339	-0.680213	0

```
[87]: df.isnull().sum()
```

```
[87]: DT      0  
      GR      0  
      NPHI    0  
      RHOB    0  
      FACIES  0  
      dtype: int64
```

```
[88]: x = df.drop("FACIES",1)  
      y = df["FACIES"]
```

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
↳random_state=8)
```

```
[89]: X_train.shape
```

```
[89]: (31767, 4)
```

```
[90]: X_test.shape
```

```
[90]: (13615, 4)
```

```
[91]: X_test
```

```
[91]:
```

	DT	GR	NPHI	RHOB
12426	-0.611997	-0.279801	-0.782950	0.364780
28889	-0.518721	1.098954	0.212866	0.943396
44033	0.491306	0.855792	0.134414	0.009192
8745	0.909116	-0.408790	0.718096	-1.192066
39849	0.403486	0.645060	-1.158473	-0.006773
...	...	...	...	...
23183	0.601655	-0.980226	0.178347	-0.124819
38398	0.257383	0.658645	-0.575837	-0.701500
1059	-0.679521	-0.525126	-1.781904	-0.685535
9662	-0.573669	0.135718	0.109310	0.110789
6669	-0.613257	1.159616	1.119770	0.870827

```
[13615 rows x 4 columns]
```

## 8 MODEL TRAINING

```
[92]: estimator=[]
```

```
[93]: gnb = GaussianNB()
```

```
[94]: model = LogisticRegression()
solvers = ['newton-cg', 'lbfgs', 'liblinear']
penalty = ['l2']
c_values = [100, 10, 1.0, 0.1, 0.01]

grid = {'solver':solvers, 'penalty':penalty, 'C':c_values}
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
grid_search = GridSearchCV(estimator=model, param_grid=grid, n_jobs=-1, cv=cv,
↳scoring='accuracy', error_score=0)
grid_result = grid_search.fit(X_train, y_train)

print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
```

```
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
    print("%f (%f) with: %r" % (mean, stdev, param))
```

```
Best: 0.889057 using {'C': 1.0, 'penalty': 'l2', 'solver': 'lbfgs'}
0.889025 (0.002086) with: {'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'}
0.889025 (0.002086) with: {'C': 100, 'penalty': 'l2', 'solver': 'lbfgs'}
0.887756 (0.001713) with: {'C': 100, 'penalty': 'l2', 'solver': 'liblinear'}
0.889025 (0.002086) with: {'C': 10, 'penalty': 'l2', 'solver': 'newton-cg'}
0.889025 (0.002086) with: {'C': 10, 'penalty': 'l2', 'solver': 'lbfgs'}
0.887777 (0.001705) with: {'C': 10, 'penalty': 'l2', 'solver': 'liblinear'}
0.889046 (0.002072) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'newton-cg'}
0.889057 (0.002075) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'lbfgs'}
0.887840 (0.001744) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}
0.888889 (0.002038) with: {'C': 0.1, 'penalty': 'l2', 'solver': 'newton-cg'}
0.888889 (0.002038) with: {'C': 0.1, 'penalty': 'l2', 'solver': 'lbfgs'}
0.887871 (0.001592) with: {'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
0.888490 (0.001355) with: {'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}
0.888490 (0.001355) with: {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
0.887252 (0.001042) with: {'C': 0.01, 'penalty': 'l2', 'solver': 'liblinear'}
```

```
[95]: dtclf = DecisionTreeClassifier(max_depth=5)
```

```
[96]: cat = CatBoostClassifier()
```

```
[97]: xgb= XGBClassifier(learning_rate =0.09,
n_estimators=494,
max_depth=5,
subsample = 0.70,
verbosity = 0,)
```

```
[98]: lgbm=LGBMClassifier(importance_type = "gain",
verbosity = -1,
max_bin = 60,
num_leaves=300,
boosting_type = 'dart',
learning_rate=0.1,
n_estimators=494,
max_depth=5, )
```

```
[99]: #neigh = KNeighborsClassifier(n_neighbors=3)
```

```
[100]: rdmclf = RandomForestClassifier(n_estimators=494,max_depth=5)
```

```
[101]: estimator.append(('gaussian',gnb))
estimator.append(('Gridlogistic',grid_search))
estimator.append(('catboost_classifier',cat))
```

```

estimator.append(('decision_tree',dtclf))
estimator.append(('xgbclassifier',xgb))
estimator.append(('LGBMclassifier',lgbm))
#estimator.append(('KNN',neigh))

```

```
[102]: vot_soft = VotingClassifier(estimators = estimator, voting = 'soft')
```

```
[103]: vot_soft.fit(X_train,y_train)
```

Learning rate set to 0.09439

0:	learn: 1.3367956	total: 56.4ms	remaining: 56.4s
1:	learn: 1.1602766	total: 62.4ms	remaining: 31.1s
2:	learn: 1.0298253	total: 68.6ms	remaining: 22.8s
3:	learn: 0.9292459	total: 74.8ms	remaining: 18.6s
4:	learn: 0.8483811	total: 80.8ms	remaining: 16.1s
5:	learn: 0.7831321	total: 86.9ms	remaining: 14.4s
6:	learn: 0.7299435	total: 93.3ms	remaining: 13.2s
7:	learn: 0.6829198	total: 99.3ms	remaining: 12.3s
8:	learn: 0.6439571	total: 105ms	remaining: 11.6s
9:	learn: 0.6095197	total: 111ms	remaining: 11s
10:	learn: 0.5801260	total: 117ms	remaining: 10.6s
11:	learn: 0.5552812	total: 123ms	remaining: 10.1s
12:	learn: 0.5327087	total: 129ms	remaining: 9.81s
13:	learn: 0.5132747	total: 135ms	remaining: 9.52s
14:	learn: 0.4964109	total: 141ms	remaining: 9.23s
15:	learn: 0.4805574	total: 147ms	remaining: 9.03s
16:	learn: 0.4666538	total: 152ms	remaining: 8.8s
17:	learn: 0.4543748	total: 158ms	remaining: 8.65s
18:	learn: 0.4441595	total: 165ms	remaining: 8.5s
19:	learn: 0.4347972	total: 170ms	remaining: 8.35s
20:	learn: 0.4261594	total: 177ms	remaining: 8.23s
21:	learn: 0.4186246	total: 182ms	remaining: 8.1s
22:	learn: 0.4120922	total: 189ms	remaining: 8.04s
23:	learn: 0.4052560	total: 197ms	remaining: 8s
24:	learn: 0.3997226	total: 204ms	remaining: 7.94s
25:	learn: 0.3944293	total: 211ms	remaining: 7.89s
26:	learn: 0.3894524	total: 218ms	remaining: 7.84s
27:	learn: 0.3849400	total: 225ms	remaining: 7.8s
28:	learn: 0.3806713	total: 232ms	remaining: 7.76s
29:	learn: 0.3770841	total: 238ms	remaining: 7.69s
30:	learn: 0.3745258	total: 244ms	remaining: 7.62s
31:	learn: 0.3719056	total: 249ms	remaining: 7.54s
32:	learn: 0.3690093	total: 256ms	remaining: 7.49s
33:	learn: 0.3663647	total: 262ms	remaining: 7.43s
34:	learn: 0.3638133	total: 268ms	remaining: 7.38s
35:	learn: 0.3615098	total: 274ms	remaining: 7.34s
36:	learn: 0.3595725	total: 280ms	remaining: 7.29s
37:	learn: 0.3583474	total: 286ms	remaining: 7.24s

38:	learn: 0.3568645	total: 292ms	remaining: 7.19s
39:	learn: 0.3553813	total: 298ms	remaining: 7.16s
40:	learn: 0.3538665	total: 304ms	remaining: 7.12s
41:	learn: 0.3522937	total: 310ms	remaining: 7.08s
42:	learn: 0.3510341	total: 316ms	remaining: 7.04s
43:	learn: 0.3497485	total: 322ms	remaining: 7s
44:	learn: 0.3488629	total: 328ms	remaining: 6.96s
45:	learn: 0.3478060	total: 334ms	remaining: 6.93s
46:	learn: 0.3469715	total: 340ms	remaining: 6.9s
47:	learn: 0.3457592	total: 347ms	remaining: 6.88s
48:	learn: 0.3443386	total: 353ms	remaining: 6.86s
49:	learn: 0.3434576	total: 360ms	remaining: 6.83s
50:	learn: 0.3425910	total: 366ms	remaining: 6.8s
51:	learn: 0.3414779	total: 372ms	remaining: 6.78s
52:	learn: 0.3405872	total: 378ms	remaining: 6.76s
53:	learn: 0.3399649	total: 385ms	remaining: 6.74s
54:	learn: 0.3390589	total: 391ms	remaining: 6.71s
55:	learn: 0.3386036	total: 397ms	remaining: 6.69s
56:	learn: 0.3374969	total: 403ms	remaining: 6.67s
57:	learn: 0.3370021	total: 409ms	remaining: 6.64s
58:	learn: 0.3362960	total: 415ms	remaining: 6.62s
59:	learn: 0.3358938	total: 421ms	remaining: 6.59s
60:	learn: 0.3350114	total: 427ms	remaining: 6.57s
61:	learn: 0.3342788	total: 433ms	remaining: 6.55s
62:	learn: 0.3336403	total: 439ms	remaining: 6.53s
63:	learn: 0.3330846	total: 445ms	remaining: 6.51s
64:	learn: 0.3325209	total: 452ms	remaining: 6.49s
65:	learn: 0.3320510	total: 458ms	remaining: 6.48s
66:	learn: 0.3316196	total: 464ms	remaining: 6.46s
67:	learn: 0.3311977	total: 470ms	remaining: 6.44s
68:	learn: 0.3304369	total: 476ms	remaining: 6.42s
69:	learn: 0.3299930	total: 483ms	remaining: 6.41s
70:	learn: 0.3294389	total: 489ms	remaining: 6.4s
71:	learn: 0.3290934	total: 495ms	remaining: 6.38s
72:	learn: 0.3284825	total: 501ms	remaining: 6.36s
73:	learn: 0.3280491	total: 507ms	remaining: 6.34s
74:	learn: 0.3277599	total: 513ms	remaining: 6.33s
75:	learn: 0.3271653	total: 519ms	remaining: 6.31s
76:	learn: 0.3266328	total: 526ms	remaining: 6.3s
77:	learn: 0.3261874	total: 532ms	remaining: 6.29s
78:	learn: 0.3258351	total: 538ms	remaining: 6.27s
79:	learn: 0.3252825	total: 544ms	remaining: 6.26s
80:	learn: 0.3250073	total: 550ms	remaining: 6.24s
81:	learn: 0.3246601	total: 556ms	remaining: 6.23s
82:	learn: 0.3242268	total: 563ms	remaining: 6.22s
83:	learn: 0.3237858	total: 570ms	remaining: 6.21s
84:	learn: 0.3234897	total: 577ms	remaining: 6.21s
85:	learn: 0.3231829	total: 583ms	remaining: 6.2s



86:	learn: 0.3229708	total: 589ms	remaining: 6.18s
87:	learn: 0.3226775	total: 595ms	remaining: 6.16s
88:	learn: 0.3222597	total: 601ms	remaining: 6.15s
89:	learn: 0.3219338	total: 607ms	remaining: 6.14s
90:	learn: 0.3215500	total: 613ms	remaining: 6.13s
91:	learn: 0.3211525	total: 619ms	remaining: 6.11s
92:	learn: 0.3204700	total: 626ms	remaining: 6.1s
93:	learn: 0.3203048	total: 632ms	remaining: 6.09s
94:	learn: 0.3200545	total: 638ms	remaining: 6.08s
95:	learn: 0.3197951	total: 644ms	remaining: 6.06s
96:	learn: 0.3194174	total: 650ms	remaining: 6.05s
97:	learn: 0.3191680	total: 656ms	remaining: 6.04s
98:	learn: 0.3188592	total: 662ms	remaining: 6.03s
99:	learn: 0.3183763	total: 668ms	remaining: 6.01s
100:	learn: 0.3180059	total: 675ms	remaining: 6s
101:	learn: 0.3178073	total: 681ms	remaining: 5.99s
102:	learn: 0.3175562	total: 686ms	remaining: 5.97s
103:	learn: 0.3172459	total: 692ms	remaining: 5.96s
104:	learn: 0.3169708	total: 698ms	remaining: 5.95s
105:	learn: 0.3165167	total: 704ms	remaining: 5.94s
106:	learn: 0.3162868	total: 710ms	remaining: 5.92s
107:	learn: 0.3159375	total: 716ms	remaining: 5.92s
108:	learn: 0.3157671	total: 723ms	remaining: 5.91s
109:	learn: 0.3155149	total: 730ms	remaining: 5.9s
110:	learn: 0.3152318	total: 736ms	remaining: 5.9s
111:	learn: 0.3150669	total: 743ms	remaining: 5.89s
112:	learn: 0.3147893	total: 749ms	remaining: 5.88s
113:	learn: 0.3145836	total: 755ms	remaining: 5.87s
114:	learn: 0.3143425	total: 763ms	remaining: 5.87s
115:	learn: 0.3141059	total: 769ms	remaining: 5.86s
116:	learn: 0.3138628	total: 776ms	remaining: 5.86s
117:	learn: 0.3135587	total: 783ms	remaining: 5.85s
118:	learn: 0.3133880	total: 789ms	remaining: 5.84s
119:	learn: 0.3130235	total: 795ms	remaining: 5.83s
120:	learn: 0.3127985	total: 801ms	remaining: 5.82s
121:	learn: 0.3124523	total: 808ms	remaining: 5.81s
122:	learn: 0.3122157	total: 814ms	remaining: 5.8s
123:	learn: 0.3119864	total: 820ms	remaining: 5.79s
124:	learn: 0.3116617	total: 827ms	remaining: 5.79s
125:	learn: 0.3113634	total: 834ms	remaining: 5.78s
126:	learn: 0.3111308	total: 840ms	remaining: 5.77s
127:	learn: 0.3109688	total: 846ms	remaining: 5.76s
128:	learn: 0.3105632	total: 853ms	remaining: 5.76s
129:	learn: 0.3102893	total: 859ms	remaining: 5.75s
130:	learn: 0.3101044	total: 865ms	remaining: 5.74s
131:	learn: 0.3097649	total: 872ms	remaining: 5.73s
132:	learn: 0.3093394	total: 878ms	remaining: 5.72s
133:	learn: 0.3090570	total: 884ms	remaining: 5.71s

134:	learn: 0.3088196	total: 890ms	remaining: 5.7s
135:	learn: 0.3086459	total: 896ms	remaining: 5.69s
136:	learn: 0.3083507	total: 902ms	remaining: 5.68s
137:	learn: 0.3081912	total: 908ms	remaining: 5.67s
138:	learn: 0.3080075	total: 914ms	remaining: 5.66s
139:	learn: 0.3077525	total: 920ms	remaining: 5.65s
140:	learn: 0.3074994	total: 926ms	remaining: 5.64s
141:	learn: 0.3072512	total: 933ms	remaining: 5.63s
142:	learn: 0.3070955	total: 938ms	remaining: 5.62s
143:	learn: 0.3069701	total: 944ms	remaining: 5.61s
144:	learn: 0.3067112	total: 951ms	remaining: 5.6s
145:	learn: 0.3064985	total: 957ms	remaining: 5.6s
146:	learn: 0.3062773	total: 964ms	remaining: 5.59s
147:	learn: 0.3061042	total: 970ms	remaining: 5.58s
148:	learn: 0.3058253	total: 976ms	remaining: 5.58s
149:	learn: 0.3056653	total: 982ms	remaining: 5.57s
150:	learn: 0.3054483	total: 989ms	remaining: 5.56s
151:	learn: 0.3053141	total: 995ms	remaining: 5.55s
152:	learn: 0.3050191	total: 1s	remaining: 5.54s
153:	learn: 0.3048245	total: 1.01s	remaining: 5.53s
154:	learn: 0.3046940	total: 1.01s	remaining: 5.52s
155:	learn: 0.3044304	total: 1.02s	remaining: 5.51s
156:	learn: 0.3041373	total: 1.02s	remaining: 5.5s
157:	learn: 0.3038548	total: 1.03s	remaining: 5.49s
158:	learn: 0.3037035	total: 1.04s	remaining: 5.48s
159:	learn: 0.3034388	total: 1.04s	remaining: 5.47s
160:	learn: 0.3032959	total: 1.05s	remaining: 5.46s
161:	learn: 0.3029703	total: 1.05s	remaining: 5.45s
162:	learn: 0.3028534	total: 1.06s	remaining: 5.44s
163:	learn: 0.3027141	total: 1.07s	remaining: 5.43s
164:	learn: 0.3025312	total: 1.07s	remaining: 5.43s
165:	learn: 0.3023283	total: 1.08s	remaining: 5.42s
166:	learn: 0.3021731	total: 1.08s	remaining: 5.41s
167:	learn: 0.3018942	total: 1.09s	remaining: 5.4s
168:	learn: 0.3015168	total: 1.1s	remaining: 5.39s
169:	learn: 0.3013187	total: 1.1s	remaining: 5.39s
170:	learn: 0.3011623	total: 1.11s	remaining: 5.38s
171:	learn: 0.3009444	total: 1.11s	remaining: 5.37s
172:	learn: 0.3007655	total: 1.12s	remaining: 5.36s
173:	learn: 0.3005106	total: 1.13s	remaining: 5.35s
174:	learn: 0.3001136	total: 1.13s	remaining: 5.35s
175:	learn: 0.2996040	total: 1.14s	remaining: 5.34s
176:	learn: 0.2994882	total: 1.15s	remaining: 5.34s
177:	learn: 0.2991650	total: 1.15s	remaining: 5.33s
178:	learn: 0.2988848	total: 1.16s	remaining: 5.32s
179:	learn: 0.2986654	total: 1.17s	remaining: 5.31s
180:	learn: 0.2984920	total: 1.17s	remaining: 5.3s
181:	learn: 0.2981041	total: 1.18s	remaining: 5.3s

182:	learn: 0.2979483	total: 1.19s	remaining: 5.29s
183:	learn: 0.2977568	total: 1.19s	remaining: 5.29s
184:	learn: 0.2976276	total: 1.2s	remaining: 5.28s
185:	learn: 0.2973269	total: 1.2s	remaining: 5.27s
186:	learn: 0.2972515	total: 1.21s	remaining: 5.26s
187:	learn: 0.2969509	total: 1.22s	remaining: 5.25s
188:	learn: 0.2967110	total: 1.22s	remaining: 5.24s
189:	learn: 0.2965108	total: 1.23s	remaining: 5.23s
190:	learn: 0.2961612	total: 1.23s	remaining: 5.22s
191:	learn: 0.2960441	total: 1.24s	remaining: 5.21s
192:	learn: 0.2957567	total: 1.25s	remaining: 5.21s
193:	learn: 0.2954925	total: 1.25s	remaining: 5.2s
194:	learn: 0.2952942	total: 1.26s	remaining: 5.19s
195:	learn: 0.2951111	total: 1.26s	remaining: 5.18s
196:	learn: 0.2949726	total: 1.27s	remaining: 5.17s
197:	learn: 0.2947348	total: 1.27s	remaining: 5.17s
198:	learn: 0.2945600	total: 1.28s	remaining: 5.16s
199:	learn: 0.2943188	total: 1.29s	remaining: 5.15s
200:	learn: 0.2942151	total: 1.29s	remaining: 5.14s
201:	learn: 0.2938776	total: 1.3s	remaining: 5.13s
202:	learn: 0.2936503	total: 1.31s	remaining: 5.13s
203:	learn: 0.2933281	total: 1.31s	remaining: 5.12s
204:	learn: 0.2932160	total: 1.32s	remaining: 5.11s
205:	learn: 0.2931260	total: 1.32s	remaining: 5.11s
206:	learn: 0.2929757	total: 1.33s	remaining: 5.1s
207:	learn: 0.2927186	total: 1.34s	remaining: 5.09s
208:	learn: 0.2925597	total: 1.34s	remaining: 5.09s
209:	learn: 0.2923648	total: 1.35s	remaining: 5.08s
210:	learn: 0.2920692	total: 1.36s	remaining: 5.07s
211:	learn: 0.2918583	total: 1.36s	remaining: 5.06s
212:	learn: 0.2917624	total: 1.37s	remaining: 5.06s
213:	learn: 0.2916380	total: 1.37s	remaining: 5.05s
214:	learn: 0.2914396	total: 1.38s	remaining: 5.04s
215:	learn: 0.2912095	total: 1.39s	remaining: 5.03s
216:	learn: 0.2909976	total: 1.39s	remaining: 5.02s
217:	learn: 0.2908663	total: 1.4s	remaining: 5.02s
218:	learn: 0.2907407	total: 1.4s	remaining: 5.01s
219:	learn: 0.2905774	total: 1.41s	remaining: 5s
220:	learn: 0.2903015	total: 1.42s	remaining: 4.99s
221:	learn: 0.2901868	total: 1.42s	remaining: 4.99s
222:	learn: 0.2899107	total: 1.43s	remaining: 4.98s
223:	learn: 0.2896550	total: 1.43s	remaining: 4.97s
224:	learn: 0.2894536	total: 1.44s	remaining: 4.96s
225:	learn: 0.2892752	total: 1.45s	remaining: 4.95s
226:	learn: 0.2892170	total: 1.45s	remaining: 4.95s
227:	learn: 0.2890366	total: 1.46s	remaining: 4.94s
228:	learn: 0.2889054	total: 1.47s	remaining: 4.93s
229:	learn: 0.2887696	total: 1.47s	remaining: 4.93s

230:	learn: 0.2886497	total: 1.48s	remaining: 4.92s
231:	learn: 0.2885708	total: 1.48s	remaining: 4.91s
232:	learn: 0.2884563	total: 1.49s	remaining: 4.91s
233:	learn: 0.2883899	total: 1.5s	remaining: 4.9s
234:	learn: 0.2881081	total: 1.5s	remaining: 4.89s
235:	learn: 0.2879521	total: 1.51s	remaining: 4.88s
236:	learn: 0.2877339	total: 1.51s	remaining: 4.88s
237:	learn: 0.2875652	total: 1.52s	remaining: 4.87s
238:	learn: 0.2874907	total: 1.53s	remaining: 4.86s
239:	learn: 0.2873773	total: 1.53s	remaining: 4.86s
240:	learn: 0.2872232	total: 1.54s	remaining: 4.86s
241:	learn: 0.2870978	total: 1.55s	remaining: 4.85s
242:	learn: 0.2869705	total: 1.55s	remaining: 4.84s
243:	learn: 0.2868598	total: 1.56s	remaining: 4.83s
244:	learn: 0.2866931	total: 1.56s	remaining: 4.82s
245:	learn: 0.2865765	total: 1.57s	remaining: 4.82s
246:	learn: 0.2864405	total: 1.58s	remaining: 4.81s
247:	learn: 0.2862633	total: 1.58s	remaining: 4.8s
248:	learn: 0.2860142	total: 1.59s	remaining: 4.79s
249:	learn: 0.2857744	total: 1.59s	remaining: 4.79s
250:	learn: 0.2856757	total: 1.6s	remaining: 4.78s
251:	learn: 0.2855731	total: 1.61s	remaining: 4.77s
252:	learn: 0.2854208	total: 1.61s	remaining: 4.77s
253:	learn: 0.2853023	total: 1.62s	remaining: 4.76s
254:	learn: 0.2851977	total: 1.63s	remaining: 4.75s
255:	learn: 0.2850590	total: 1.63s	remaining: 4.74s
256:	learn: 0.2849114	total: 1.64s	remaining: 4.74s
257:	learn: 0.2847444	total: 1.64s	remaining: 4.73s
258:	learn: 0.2846577	total: 1.65s	remaining: 4.72s
259:	learn: 0.2845010	total: 1.66s	remaining: 4.71s
260:	learn: 0.2842627	total: 1.66s	remaining: 4.71s
261:	learn: 0.2841076	total: 1.67s	remaining: 4.7s
262:	learn: 0.2839592	total: 1.68s	remaining: 4.7s
263:	learn: 0.2838281	total: 1.68s	remaining: 4.69s
264:	learn: 0.2837066	total: 1.69s	remaining: 4.69s
265:	learn: 0.2835626	total: 1.7s	remaining: 4.68s
266:	learn: 0.2833237	total: 1.7s	remaining: 4.68s
267:	learn: 0.2832175	total: 1.71s	remaining: 4.67s
268:	learn: 0.2831187	total: 1.72s	remaining: 4.67s
269:	learn: 0.2829665	total: 1.73s	remaining: 4.66s
270:	learn: 0.2827940	total: 1.73s	remaining: 4.66s
271:	learn: 0.2827198	total: 1.74s	remaining: 4.65s
272:	learn: 0.2825885	total: 1.75s	remaining: 4.65s
273:	learn: 0.2824501	total: 1.75s	remaining: 4.64s
274:	learn: 0.2823199	total: 1.76s	remaining: 4.64s
275:	learn: 0.2821785	total: 1.76s	remaining: 4.63s
276:	learn: 0.2820174	total: 1.77s	remaining: 4.62s
277:	learn: 0.2818062	total: 1.78s	remaining: 4.62s

278:	learn: 0.2815752	total: 1.78s	remaining: 4.61s
279:	learn: 0.2814820	total: 1.79s	remaining: 4.61s
280:	learn: 0.2812939	total: 1.8s	remaining: 4.6s
281:	learn: 0.2810843	total: 1.8s	remaining: 4.59s
282:	learn: 0.2809318	total: 1.81s	remaining: 4.59s
283:	learn: 0.2807730	total: 1.82s	remaining: 4.58s
284:	learn: 0.2804447	total: 1.82s	remaining: 4.57s
285:	learn: 0.2803105	total: 1.83s	remaining: 4.57s
286:	learn: 0.2802235	total: 1.83s	remaining: 4.56s
287:	learn: 0.2800613	total: 1.84s	remaining: 4.55s
288:	learn: 0.2799600	total: 1.85s	remaining: 4.55s
289:	learn: 0.2798184	total: 1.85s	remaining: 4.54s
290:	learn: 0.2796161	total: 1.86s	remaining: 4.53s
291:	learn: 0.2795185	total: 1.87s	remaining: 4.53s
292:	learn: 0.2793586	total: 1.87s	remaining: 4.52s
293:	learn: 0.2791426	total: 1.88s	remaining: 4.51s
294:	learn: 0.2789122	total: 1.89s	remaining: 4.51s
295:	learn: 0.2786988	total: 1.89s	remaining: 4.5s
296:	learn: 0.2786184	total: 1.9s	remaining: 4.49s
297:	learn: 0.2785593	total: 1.9s	remaining: 4.49s
298:	learn: 0.2784718	total: 1.91s	remaining: 4.48s
299:	learn: 0.2782639	total: 1.92s	remaining: 4.47s
300:	learn: 0.2782007	total: 1.92s	remaining: 4.47s
301:	learn: 0.2780756	total: 1.93s	remaining: 4.46s
302:	learn: 0.2779471	total: 1.94s	remaining: 4.45s
303:	learn: 0.2775524	total: 1.94s	remaining: 4.45s
304:	learn: 0.2774668	total: 1.95s	remaining: 4.44s
305:	learn: 0.2772705	total: 1.95s	remaining: 4.43s
306:	learn: 0.2771690	total: 1.96s	remaining: 4.42s
307:	learn: 0.2770358	total: 1.97s	remaining: 4.42s
308:	learn: 0.2769417	total: 1.97s	remaining: 4.41s
309:	learn: 0.2768011	total: 1.98s	remaining: 4.4s
310:	learn: 0.2766834	total: 1.98s	remaining: 4.39s
311:	learn: 0.2765089	total: 1.99s	remaining: 4.39s
312:	learn: 0.2763590	total: 2s	remaining: 4.38s
313:	learn: 0.2762599	total: 2s	remaining: 4.37s
314:	learn: 0.2761610	total: 2.01s	remaining: 4.37s
315:	learn: 0.2760224	total: 2.01s	remaining: 4.36s
316:	learn: 0.2758999	total: 2.02s	remaining: 4.35s
317:	learn: 0.2757953	total: 2.03s	remaining: 4.34s
318:	learn: 0.2756507	total: 2.03s	remaining: 4.34s
319:	learn: 0.2755430	total: 2.04s	remaining: 4.33s
320:	learn: 0.2754516	total: 2.04s	remaining: 4.33s
321:	learn: 0.2753148	total: 2.05s	remaining: 4.32s
322:	learn: 0.2752186	total: 2.06s	remaining: 4.31s
323:	learn: 0.2750016	total: 2.06s	remaining: 4.3s
324:	learn: 0.2749380	total: 2.07s	remaining: 4.3s
325:	learn: 0.2748031	total: 2.08s	remaining: 4.29s

326:	learn: 0.2746812	total: 2.08s	remaining: 4.28s
327:	learn: 0.2745749	total: 2.09s	remaining: 4.28s
328:	learn: 0.2743892	total: 2.1s	remaining: 4.27s
329:	learn: 0.2743152	total: 2.1s	remaining: 4.27s
330:	learn: 0.2741740	total: 2.11s	remaining: 4.26s
331:	learn: 0.2740444	total: 2.12s	remaining: 4.26s
332:	learn: 0.2738985	total: 2.12s	remaining: 4.25s
333:	learn: 0.2737223	total: 2.13s	remaining: 4.25s
334:	learn: 0.2735650	total: 2.13s	remaining: 4.24s
335:	learn: 0.2733959	total: 2.14s	remaining: 4.23s
336:	learn: 0.2731180	total: 2.15s	remaining: 4.23s
337:	learn: 0.2730099	total: 2.15s	remaining: 4.22s
338:	learn: 0.2729531	total: 2.16s	remaining: 4.21s
339:	learn: 0.2728817	total: 2.17s	remaining: 4.21s
340:	learn: 0.2727492	total: 2.17s	remaining: 4.2s
341:	learn: 0.2726345	total: 2.18s	remaining: 4.19s
342:	learn: 0.2724753	total: 2.18s	remaining: 4.18s
343:	learn: 0.2723593	total: 2.19s	remaining: 4.17s
344:	learn: 0.2722565	total: 2.19s	remaining: 4.17s
345:	learn: 0.2721841	total: 2.2s	remaining: 4.16s
346:	learn: 0.2720513	total: 2.21s	remaining: 4.15s
347:	learn: 0.2719258	total: 2.21s	remaining: 4.15s
348:	learn: 0.2718248	total: 2.22s	remaining: 4.14s
349:	learn: 0.2717345	total: 2.23s	remaining: 4.13s
350:	learn: 0.2716310	total: 2.23s	remaining: 4.13s
351:	learn: 0.2715329	total: 2.24s	remaining: 4.12s
352:	learn: 0.2713853	total: 2.24s	remaining: 4.11s
353:	learn: 0.2712704	total: 2.25s	remaining: 4.11s
354:	learn: 0.2712007	total: 2.25s	remaining: 4.1s
355:	learn: 0.2710623	total: 2.26s	remaining: 4.09s
356:	learn: 0.2709516	total: 2.27s	remaining: 4.08s
357:	learn: 0.2708063	total: 2.27s	remaining: 4.08s
358:	learn: 0.2706784	total: 2.28s	remaining: 4.07s
359:	learn: 0.2705619	total: 2.29s	remaining: 4.06s
360:	learn: 0.2704923	total: 2.29s	remaining: 4.06s
361:	learn: 0.2704005	total: 2.3s	remaining: 4.05s
362:	learn: 0.2702751	total: 2.31s	remaining: 4.05s
363:	learn: 0.2701480	total: 2.31s	remaining: 4.04s
364:	learn: 0.2700718	total: 2.32s	remaining: 4.03s
365:	learn: 0.2699038	total: 2.32s	remaining: 4.03s
366:	learn: 0.2697065	total: 2.33s	remaining: 4.02s
367:	learn: 0.2695748	total: 2.34s	remaining: 4.01s
368:	learn: 0.2694936	total: 2.34s	remaining: 4s
369:	learn: 0.2693984	total: 2.35s	remaining: 4s
370:	learn: 0.2692990	total: 2.35s	remaining: 3.99s
371:	learn: 0.2691059	total: 2.36s	remaining: 3.98s
372:	learn: 0.2690015	total: 2.37s	remaining: 3.98s
373:	learn: 0.2689173	total: 2.37s	remaining: 3.97s

374:	learn: 0.2688303	total: 2.38s	remaining: 3.96s
375:	learn: 0.2687481	total: 2.38s	remaining: 3.96s
376:	learn: 0.2686564	total: 2.39s	remaining: 3.95s
377:	learn: 0.2685327	total: 2.4s	remaining: 3.94s
378:	learn: 0.2684668	total: 2.4s	remaining: 3.94s
379:	learn: 0.2683571	total: 2.41s	remaining: 3.93s
380:	learn: 0.2682622	total: 2.42s	remaining: 3.92s
381:	learn: 0.2682010	total: 2.42s	remaining: 3.92s
382:	learn: 0.2680836	total: 2.43s	remaining: 3.91s
383:	learn: 0.2679303	total: 2.44s	remaining: 3.91s
384:	learn: 0.2678129	total: 2.44s	remaining: 3.9s
385:	learn: 0.2677309	total: 2.45s	remaining: 3.89s
386:	learn: 0.2676421	total: 2.45s	remaining: 3.88s
387:	learn: 0.2675883	total: 2.46s	remaining: 3.88s
388:	learn: 0.2673997	total: 2.47s	remaining: 3.87s
389:	learn: 0.2671956	total: 2.47s	remaining: 3.87s
390:	learn: 0.2670907	total: 2.48s	remaining: 3.86s
391:	learn: 0.2669769	total: 2.48s	remaining: 3.85s
392:	learn: 0.2668648	total: 2.49s	remaining: 3.85s
393:	learn: 0.2667405	total: 2.5s	remaining: 3.84s
394:	learn: 0.2666316	total: 2.5s	remaining: 3.84s
395:	learn: 0.2665452	total: 2.51s	remaining: 3.83s
396:	learn: 0.2664881	total: 2.52s	remaining: 3.82s
397:	learn: 0.2663782	total: 2.52s	remaining: 3.81s
398:	learn: 0.2663286	total: 2.53s	remaining: 3.81s
399:	learn: 0.2662265	total: 2.53s	remaining: 3.8s
400:	learn: 0.2661619	total: 2.54s	remaining: 3.79s
401:	learn: 0.2660834	total: 2.55s	remaining: 3.79s
402:	learn: 0.2659274	total: 2.55s	remaining: 3.78s
403:	learn: 0.2657917	total: 2.56s	remaining: 3.77s
404:	learn: 0.2656862	total: 2.56s	remaining: 3.77s
405:	learn: 0.2656121	total: 2.57s	remaining: 3.76s
406:	learn: 0.2655104	total: 2.58s	remaining: 3.75s
407:	learn: 0.2653870	total: 2.58s	remaining: 3.75s
408:	learn: 0.2653328	total: 2.59s	remaining: 3.74s
409:	learn: 0.2652230	total: 2.6s	remaining: 3.74s
410:	learn: 0.2651407	total: 2.6s	remaining: 3.73s
411:	learn: 0.2650402	total: 2.61s	remaining: 3.72s
412:	learn: 0.2648654	total: 2.62s	remaining: 3.72s
413:	learn: 0.2647231	total: 2.62s	remaining: 3.71s
414:	learn: 0.2646342	total: 2.63s	remaining: 3.71s
415:	learn: 0.2645008	total: 2.64s	remaining: 3.7s
416:	learn: 0.2643958	total: 2.64s	remaining: 3.69s
417:	learn: 0.2643196	total: 2.65s	remaining: 3.69s
418:	learn: 0.2642114	total: 2.65s	remaining: 3.68s
419:	learn: 0.2641582	total: 2.66s	remaining: 3.68s
420:	learn: 0.2640704	total: 2.67s	remaining: 3.67s
421:	learn: 0.2638255	total: 2.68s	remaining: 3.67s

422:	learn: 0.2637427	total: 2.68s	remaining: 3.66s
423:	learn: 0.2635994	total: 2.69s	remaining: 3.65s
424:	learn: 0.2634996	total: 2.7s	remaining: 3.65s
425:	learn: 0.2634363	total: 2.7s	remaining: 3.64s
426:	learn: 0.2633764	total: 2.71s	remaining: 3.64s
427:	learn: 0.2632237	total: 2.72s	remaining: 3.63s
428:	learn: 0.2631362	total: 2.72s	remaining: 3.62s
429:	learn: 0.2630197	total: 2.73s	remaining: 3.62s
430:	learn: 0.2629168	total: 2.74s	remaining: 3.61s
431:	learn: 0.2628323	total: 2.74s	remaining: 3.61s
432:	learn: 0.2627103	total: 2.75s	remaining: 3.6s
433:	learn: 0.2626552	total: 2.75s	remaining: 3.59s
434:	learn: 0.2625413	total: 2.76s	remaining: 3.58s
435:	learn: 0.2624752	total: 2.77s	remaining: 3.58s
436:	learn: 0.2623947	total: 2.77s	remaining: 3.57s
437:	learn: 0.2622653	total: 2.78s	remaining: 3.56s
438:	learn: 0.2622003	total: 2.78s	remaining: 3.56s
439:	learn: 0.2621552	total: 2.79s	remaining: 3.55s
440:	learn: 0.2621076	total: 2.8s	remaining: 3.54s
441:	learn: 0.2619954	total: 2.8s	remaining: 3.54s
442:	learn: 0.2619243	total: 2.81s	remaining: 3.53s
443:	learn: 0.2617870	total: 2.81s	remaining: 3.52s
444:	learn: 0.2617091	total: 2.82s	remaining: 3.52s
445:	learn: 0.2615964	total: 2.83s	remaining: 3.51s
446:	learn: 0.2615205	total: 2.83s	remaining: 3.5s
447:	learn: 0.2614178	total: 2.84s	remaining: 3.5s
448:	learn: 0.2612950	total: 2.85s	remaining: 3.49s
449:	learn: 0.2612002	total: 2.85s	remaining: 3.48s
450:	learn: 0.2611347	total: 2.86s	remaining: 3.48s
451:	learn: 0.2610463	total: 2.86s	remaining: 3.47s
452:	learn: 0.2608777	total: 2.87s	remaining: 3.47s
453:	learn: 0.2607636	total: 2.88s	remaining: 3.46s
454:	learn: 0.2606813	total: 2.88s	remaining: 3.45s
455:	learn: 0.2604076	total: 2.89s	remaining: 3.45s
456:	learn: 0.2602625	total: 2.9s	remaining: 3.44s
457:	learn: 0.2601186	total: 2.9s	remaining: 3.43s
458:	learn: 0.2600762	total: 2.91s	remaining: 3.43s
459:	learn: 0.2599852	total: 2.91s	remaining: 3.42s
460:	learn: 0.2598507	total: 2.92s	remaining: 3.41s
461:	learn: 0.2598201	total: 2.93s	remaining: 3.41s
462:	learn: 0.2597180	total: 2.93s	remaining: 3.4s
463:	learn: 0.2596171	total: 2.94s	remaining: 3.39s
464:	learn: 0.2594738	total: 2.94s	remaining: 3.39s
465:	learn: 0.2592503	total: 2.95s	remaining: 3.38s
466:	learn: 0.2591417	total: 2.96s	remaining: 3.38s
467:	learn: 0.2590521	total: 2.96s	remaining: 3.37s
468:	learn: 0.2589964	total: 2.97s	remaining: 3.36s
469:	learn: 0.2589497	total: 2.98s	remaining: 3.36s



470:	learn: 0.2588905	total: 2.98s	remaining: 3.35s
471:	learn: 0.2588037	total: 2.99s	remaining: 3.35s
472:	learn: 0.2587408	total: 3s	remaining: 3.34s
473:	learn: 0.2586714	total: 3s	remaining: 3.33s
474:	learn: 0.2586189	total: 3.01s	remaining: 3.33s
475:	learn: 0.2584936	total: 3.01s	remaining: 3.32s
476:	learn: 0.2584323	total: 3.02s	remaining: 3.31s
477:	learn: 0.2583432	total: 3.03s	remaining: 3.31s
478:	learn: 0.2582333	total: 3.03s	remaining: 3.3s
479:	learn: 0.2581367	total: 3.04s	remaining: 3.29s
480:	learn: 0.2580089	total: 3.04s	remaining: 3.29s
481:	learn: 0.2579155	total: 3.05s	remaining: 3.28s
482:	learn: 0.2578158	total: 3.06s	remaining: 3.27s
483:	learn: 0.2577281	total: 3.06s	remaining: 3.27s
484:	learn: 0.2576124	total: 3.07s	remaining: 3.26s
485:	learn: 0.2574869	total: 3.08s	remaining: 3.25s
486:	learn: 0.2573964	total: 3.08s	remaining: 3.25s
487:	learn: 0.2573201	total: 3.09s	remaining: 3.24s
488:	learn: 0.2571804	total: 3.1s	remaining: 3.23s
489:	learn: 0.2570901	total: 3.1s	remaining: 3.23s
490:	learn: 0.2569869	total: 3.11s	remaining: 3.22s
491:	learn: 0.2569291	total: 3.11s	remaining: 3.21s
492:	learn: 0.2567042	total: 3.12s	remaining: 3.21s
493:	learn: 0.2566493	total: 3.13s	remaining: 3.2s
494:	learn: 0.2565020	total: 3.13s	remaining: 3.19s
495:	learn: 0.2563995	total: 3.14s	remaining: 3.19s
496:	learn: 0.2563222	total: 3.14s	remaining: 3.18s
497:	learn: 0.2562671	total: 3.15s	remaining: 3.17s
498:	learn: 0.2561604	total: 3.15s	remaining: 3.17s
499:	learn: 0.2560843	total: 3.16s	remaining: 3.16s
500:	learn: 0.2560038	total: 3.17s	remaining: 3.15s
501:	learn: 0.2559036	total: 3.17s	remaining: 3.15s
502:	learn: 0.2558095	total: 3.18s	remaining: 3.14s
503:	learn: 0.2557170	total: 3.19s	remaining: 3.14s
504:	learn: 0.2556188	total: 3.19s	remaining: 3.13s
505:	learn: 0.2555331	total: 3.2s	remaining: 3.12s
506:	learn: 0.2555090	total: 3.2s	remaining: 3.12s
507:	learn: 0.2554205	total: 3.21s	remaining: 3.11s
508:	learn: 0.2553227	total: 3.22s	remaining: 3.1s
509:	learn: 0.2552442	total: 3.22s	remaining: 3.1s
510:	learn: 0.2551481	total: 3.23s	remaining: 3.09s
511:	learn: 0.2550837	total: 3.23s	remaining: 3.08s
512:	learn: 0.2550166	total: 3.24s	remaining: 3.08s
513:	learn: 0.2549379	total: 3.25s	remaining: 3.07s
514:	learn: 0.2548620	total: 3.25s	remaining: 3.06s
515:	learn: 0.2547962	total: 3.26s	remaining: 3.06s
516:	learn: 0.2546910	total: 3.27s	remaining: 3.05s
517:	learn: 0.2546081	total: 3.27s	remaining: 3.04s

518:	learn: 0.2544975	total: 3.28s	remaining: 3.04s
519:	learn: 0.2544199	total: 3.29s	remaining: 3.03s
520:	learn: 0.2543803	total: 3.29s	remaining: 3.02s
521:	learn: 0.2543131	total: 3.3s	remaining: 3.02s
522:	learn: 0.2542394	total: 3.3s	remaining: 3.01s
523:	learn: 0.2541603	total: 3.31s	remaining: 3.01s
524:	learn: 0.2540938	total: 3.32s	remaining: 3s
525:	learn: 0.2538871	total: 3.32s	remaining: 2.99s
526:	learn: 0.2537980	total: 3.33s	remaining: 2.99s
527:	learn: 0.2536619	total: 3.33s	remaining: 2.98s
528:	learn: 0.2535848	total: 3.34s	remaining: 2.97s
529:	learn: 0.2534846	total: 3.35s	remaining: 2.97s
530:	learn: 0.2534489	total: 3.35s	remaining: 2.96s
531:	learn: 0.2532777	total: 3.36s	remaining: 2.95s
532:	learn: 0.2532103	total: 3.37s	remaining: 2.95s
533:	learn: 0.2531271	total: 3.37s	remaining: 2.94s
534:	learn: 0.2529207	total: 3.38s	remaining: 2.94s
535:	learn: 0.2527685	total: 3.38s	remaining: 2.93s
536:	learn: 0.2526413	total: 3.39s	remaining: 2.92s
537:	learn: 0.2525089	total: 3.4s	remaining: 2.92s
538:	learn: 0.2523284	total: 3.4s	remaining: 2.91s
539:	learn: 0.2522780	total: 3.41s	remaining: 2.9s
540:	learn: 0.2522395	total: 3.41s	remaining: 2.9s
541:	learn: 0.2521524	total: 3.42s	remaining: 2.89s
542:	learn: 0.2520933	total: 3.43s	remaining: 2.88s
543:	learn: 0.2520166	total: 3.43s	remaining: 2.88s
544:	learn: 0.2519416	total: 3.44s	remaining: 2.87s
545:	learn: 0.2518710	total: 3.45s	remaining: 2.87s
546:	learn: 0.2517758	total: 3.45s	remaining: 2.86s
547:	learn: 0.2516452	total: 3.46s	remaining: 2.85s
548:	learn: 0.2515885	total: 3.46s	remaining: 2.85s
549:	learn: 0.2514847	total: 3.47s	remaining: 2.84s
550:	learn: 0.2514331	total: 3.48s	remaining: 2.83s
551:	learn: 0.2513382	total: 3.48s	remaining: 2.83s
552:	learn: 0.2511431	total: 3.49s	remaining: 2.82s
553:	learn: 0.2510455	total: 3.5s	remaining: 2.81s
554:	learn: 0.2509824	total: 3.5s	remaining: 2.81s
555:	learn: 0.2509060	total: 3.51s	remaining: 2.8s
556:	learn: 0.2507642	total: 3.52s	remaining: 2.8s
557:	learn: 0.2506752	total: 3.52s	remaining: 2.79s
558:	learn: 0.2505516	total: 3.53s	remaining: 2.78s
559:	learn: 0.2505055	total: 3.54s	remaining: 2.78s
560:	learn: 0.2504134	total: 3.54s	remaining: 2.77s
561:	learn: 0.2503059	total: 3.55s	remaining: 2.77s
562:	learn: 0.2501826	total: 3.55s	remaining: 2.76s
563:	learn: 0.2501198	total: 3.56s	remaining: 2.75s
564:	learn: 0.2500592	total: 3.57s	remaining: 2.75s
565:	learn: 0.2500024	total: 3.57s	remaining: 2.74s

566:	learn: 0.2499604	total: 3.58s	remaining: 2.73s
567:	learn: 0.2498728	total: 3.59s	remaining: 2.73s
568:	learn: 0.2498032	total: 3.59s	remaining: 2.72s
569:	learn: 0.2496540	total: 3.6s	remaining: 2.71s
570:	learn: 0.2495724	total: 3.61s	remaining: 2.71s
571:	learn: 0.2494451	total: 3.61s	remaining: 2.7s
572:	learn: 0.2493327	total: 3.62s	remaining: 2.7s
573:	learn: 0.2492647	total: 3.63s	remaining: 2.69s
574:	learn: 0.2491882	total: 3.63s	remaining: 2.69s
575:	learn: 0.2490807	total: 3.64s	remaining: 2.68s
576:	learn: 0.2489841	total: 3.65s	remaining: 2.67s
577:	learn: 0.2489482	total: 3.66s	remaining: 2.67s
578:	learn: 0.2488927	total: 3.66s	remaining: 2.66s
579:	learn: 0.2488102	total: 3.67s	remaining: 2.66s
580:	learn: 0.2487541	total: 3.67s	remaining: 2.65s
581:	learn: 0.2486643	total: 3.68s	remaining: 2.64s
582:	learn: 0.2484859	total: 3.69s	remaining: 2.64s
583:	learn: 0.2484295	total: 3.69s	remaining: 2.63s
584:	learn: 0.2483018	total: 3.7s	remaining: 2.62s
585:	learn: 0.2482065	total: 3.71s	remaining: 2.62s
586:	learn: 0.2481333	total: 3.71s	remaining: 2.61s
587:	learn: 0.2480836	total: 3.72s	remaining: 2.6s
588:	learn: 0.2479980	total: 3.72s	remaining: 2.6s
589:	learn: 0.2479136	total: 3.73s	remaining: 2.59s
590:	learn: 0.2477979	total: 3.74s	remaining: 2.59s
591:	learn: 0.2477116	total: 3.74s	remaining: 2.58s
592:	learn: 0.2476033	total: 3.75s	remaining: 2.57s
593:	learn: 0.2475288	total: 3.76s	remaining: 2.57s
594:	learn: 0.2474709	total: 3.76s	remaining: 2.56s
595:	learn: 0.2473696	total: 3.77s	remaining: 2.55s
596:	learn: 0.2472432	total: 3.77s	remaining: 2.55s
597:	learn: 0.2471214	total: 3.78s	remaining: 2.54s
598:	learn: 0.2470466	total: 3.79s	remaining: 2.54s
599:	learn: 0.2469823	total: 3.79s	remaining: 2.53s
600:	learn: 0.2469287	total: 3.8s	remaining: 2.52s
601:	learn: 0.2468413	total: 3.81s	remaining: 2.52s
602:	learn: 0.2467535	total: 3.81s	remaining: 2.51s
603:	learn: 0.2466709	total: 3.82s	remaining: 2.5s
604:	learn: 0.2465446	total: 3.83s	remaining: 2.5s
605:	learn: 0.2464751	total: 3.83s	remaining: 2.49s
606:	learn: 0.2463954	total: 3.84s	remaining: 2.49s
607:	learn: 0.2462520	total: 3.85s	remaining: 2.48s
608:	learn: 0.2462011	total: 3.85s	remaining: 2.47s
609:	learn: 0.2461367	total: 3.86s	remaining: 2.47s
610:	learn: 0.2460300	total: 3.86s	remaining: 2.46s
611:	learn: 0.2459615	total: 3.87s	remaining: 2.45s
612:	learn: 0.2458399	total: 3.88s	remaining: 2.45s
613:	learn: 0.2457985	total: 3.88s	remaining: 2.44s

614:	learn: 0.2457246	total: 3.89s	remaining: 2.43s
615:	learn: 0.2456538	total: 3.89s	remaining: 2.43s
616:	learn: 0.2455295	total: 3.9s	remaining: 2.42s
617:	learn: 0.2454038	total: 3.91s	remaining: 2.41s
618:	learn: 0.2453181	total: 3.91s	remaining: 2.41s
619:	learn: 0.2452490	total: 3.92s	remaining: 2.4s
620:	learn: 0.2451567	total: 3.92s	remaining: 2.4s
621:	learn: 0.2450922	total: 3.93s	remaining: 2.39s
622:	learn: 0.2450215	total: 3.94s	remaining: 2.38s
623:	learn: 0.2449133	total: 3.94s	remaining: 2.38s
624:	learn: 0.2448119	total: 3.95s	remaining: 2.37s
625:	learn: 0.2446947	total: 3.96s	remaining: 2.36s
626:	learn: 0.2446392	total: 3.96s	remaining: 2.36s
627:	learn: 0.2445249	total: 3.97s	remaining: 2.35s
628:	learn: 0.2444434	total: 3.98s	remaining: 2.35s
629:	learn: 0.2443596	total: 3.98s	remaining: 2.34s
630:	learn: 0.2443059	total: 3.99s	remaining: 2.33s
631:	learn: 0.2441549	total: 4s	remaining: 2.33s
632:	learn: 0.2441040	total: 4s	remaining: 2.32s
633:	learn: 0.2440266	total: 4.01s	remaining: 2.31s
634:	learn: 0.2439436	total: 4.01s	remaining: 2.31s
635:	learn: 0.2438812	total: 4.02s	remaining: 2.3s
636:	learn: 0.2437569	total: 4.03s	remaining: 2.29s
637:	learn: 0.2436947	total: 4.03s	remaining: 2.29s
638:	learn: 0.2436206	total: 4.04s	remaining: 2.28s
639:	learn: 0.2435426	total: 4.05s	remaining: 2.28s
640:	learn: 0.2433920	total: 4.05s	remaining: 2.27s
641:	learn: 0.2433449	total: 4.06s	remaining: 2.26s
642:	learn: 0.2432299	total: 4.07s	remaining: 2.26s
643:	learn: 0.2431250	total: 4.07s	remaining: 2.25s
644:	learn: 0.2430396	total: 4.08s	remaining: 2.24s
645:	learn: 0.2429292	total: 4.08s	remaining: 2.24s
646:	learn: 0.2428473	total: 4.09s	remaining: 2.23s
647:	learn: 0.2427629	total: 4.1s	remaining: 2.23s
648:	learn: 0.2427147	total: 4.1s	remaining: 2.22s
649:	learn: 0.2426303	total: 4.11s	remaining: 2.21s
650:	learn: 0.2425118	total: 4.11s	remaining: 2.21s
651:	learn: 0.2424726	total: 4.12s	remaining: 2.2s
652:	learn: 0.2424422	total: 4.13s	remaining: 2.19s
653:	learn: 0.2423798	total: 4.13s	remaining: 2.19s
654:	learn: 0.2423368	total: 4.14s	remaining: 2.18s
655:	learn: 0.2422963	total: 4.14s	remaining: 2.17s
656:	learn: 0.2422347	total: 4.15s	remaining: 2.17s
657:	learn: 0.2421959	total: 4.16s	remaining: 2.16s
658:	learn: 0.2421487	total: 4.16s	remaining: 2.15s
659:	learn: 0.2420713	total: 4.17s	remaining: 2.15s
660:	learn: 0.2419673	total: 4.17s	remaining: 2.14s
661:	learn: 0.2419033	total: 4.18s	remaining: 2.13s

662:	learn: 0.2418503	total: 4.19s	remaining: 2.13s
663:	learn: 0.2416912	total: 4.19s	remaining: 2.12s
664:	learn: 0.2416049	total: 4.2s	remaining: 2.12s
665:	learn: 0.2415561	total: 4.21s	remaining: 2.11s
666:	learn: 0.2414929	total: 4.21s	remaining: 2.1s
667:	learn: 0.2414186	total: 4.22s	remaining: 2.1s
668:	learn: 0.2413266	total: 4.23s	remaining: 2.09s
669:	learn: 0.2412428	total: 4.23s	remaining: 2.08s
670:	learn: 0.2411898	total: 4.24s	remaining: 2.08s
671:	learn: 0.2411216	total: 4.24s	remaining: 2.07s
672:	learn: 0.2410094	total: 4.25s	remaining: 2.06s
673:	learn: 0.2409216	total: 4.26s	remaining: 2.06s
674:	learn: 0.2408690	total: 4.26s	remaining: 2.05s
675:	learn: 0.2407428	total: 4.27s	remaining: 2.05s
676:	learn: 0.2406709	total: 4.28s	remaining: 2.04s
677:	learn: 0.2406149	total: 4.28s	remaining: 2.03s
678:	learn: 0.2405519	total: 4.29s	remaining: 2.03s
679:	learn: 0.2405014	total: 4.29s	remaining: 2.02s
680:	learn: 0.2403895	total: 4.3s	remaining: 2.01s
681:	learn: 0.2402308	total: 4.31s	remaining: 2.01s
682:	learn: 0.2401826	total: 4.31s	remaining: 2s
683:	learn: 0.2400005	total: 4.32s	remaining: 2s
684:	learn: 0.2399029	total: 4.33s	remaining: 1.99s
685:	learn: 0.2397791	total: 4.33s	remaining: 1.98s
686:	learn: 0.2397364	total: 4.34s	remaining: 1.98s
687:	learn: 0.2396854	total: 4.34s	remaining: 1.97s
688:	learn: 0.2396217	total: 4.35s	remaining: 1.96s
689:	learn: 0.2395792	total: 4.36s	remaining: 1.96s
690:	learn: 0.2395413	total: 4.36s	remaining: 1.95s
691:	learn: 0.2394504	total: 4.37s	remaining: 1.94s
692:	learn: 0.2393753	total: 4.37s	remaining: 1.94s
693:	learn: 0.2393297	total: 4.38s	remaining: 1.93s
694:	learn: 0.2392802	total: 4.38s	remaining: 1.92s
695:	learn: 0.2391343	total: 4.39s	remaining: 1.92s
696:	learn: 0.2390281	total: 4.4s	remaining: 1.91s
697:	learn: 0.2389856	total: 4.41s	remaining: 1.91s
698:	learn: 0.2389111	total: 4.41s	remaining: 1.9s
699:	learn: 0.2388349	total: 4.42s	remaining: 1.89s
700:	learn: 0.2387774	total: 4.42s	remaining: 1.89s
701:	learn: 0.2386921	total: 4.43s	remaining: 1.88s
702:	learn: 0.2385191	total: 4.44s	remaining: 1.87s
703:	learn: 0.2384420	total: 4.44s	remaining: 1.87s
704:	learn: 0.2383801	total: 4.45s	remaining: 1.86s
705:	learn: 0.2382425	total: 4.46s	remaining: 1.86s
706:	learn: 0.2381665	total: 4.46s	remaining: 1.85s
707:	learn: 0.2381411	total: 4.47s	remaining: 1.84s
708:	learn: 0.2380394	total: 4.48s	remaining: 1.84s
709:	learn: 0.2379921	total: 4.48s	remaining: 1.83s

710:	learn: 0.2379602	total: 4.49s	remaining: 1.82s
711:	learn: 0.2378902	total: 4.5s	remaining: 1.82s
712:	learn: 0.2378513	total: 4.5s	remaining: 1.81s
713:	learn: 0.2377523	total: 4.51s	remaining: 1.81s
714:	learn: 0.2376835	total: 4.51s	remaining: 1.8s
715:	learn: 0.2375807	total: 4.52s	remaining: 1.79s
716:	learn: 0.2375266	total: 4.53s	remaining: 1.79s
717:	learn: 0.2373979	total: 4.53s	remaining: 1.78s
718:	learn: 0.2373111	total: 4.54s	remaining: 1.77s
719:	learn: 0.2372269	total: 4.55s	remaining: 1.77s
720:	learn: 0.2371601	total: 4.55s	remaining: 1.76s
721:	learn: 0.2370757	total: 4.56s	remaining: 1.75s
722:	learn: 0.2370337	total: 4.57s	remaining: 1.75s
723:	learn: 0.2369889	total: 4.57s	remaining: 1.74s
724:	learn: 0.2369318	total: 4.58s	remaining: 1.74s
725:	learn: 0.2368698	total: 4.58s	remaining: 1.73s
726:	learn: 0.2368038	total: 4.59s	remaining: 1.72s
727:	learn: 0.2367750	total: 4.6s	remaining: 1.72s
728:	learn: 0.2367120	total: 4.61s	remaining: 1.71s
729:	learn: 0.2366281	total: 4.61s	remaining: 1.71s
730:	learn: 0.2365741	total: 4.62s	remaining: 1.7s
731:	learn: 0.2364302	total: 4.62s	remaining: 1.69s
732:	learn: 0.2363615	total: 4.63s	remaining: 1.69s
733:	learn: 0.2363141	total: 4.64s	remaining: 1.68s
734:	learn: 0.2362371	total: 4.64s	remaining: 1.67s
735:	learn: 0.2361416	total: 4.65s	remaining: 1.67s
736:	learn: 0.2360844	total: 4.66s	remaining: 1.66s
737:	learn: 0.2360398	total: 4.66s	remaining: 1.65s
738:	learn: 0.2359805	total: 4.67s	remaining: 1.65s
739:	learn: 0.2359010	total: 4.67s	remaining: 1.64s
740:	learn: 0.2358213	total: 4.68s	remaining: 1.64s
741:	learn: 0.2357256	total: 4.69s	remaining: 1.63s
742:	learn: 0.2356862	total: 4.69s	remaining: 1.62s
743:	learn: 0.2356294	total: 4.7s	remaining: 1.62s
744:	learn: 0.2355027	total: 4.71s	remaining: 1.61s
745:	learn: 0.2354547	total: 4.71s	remaining: 1.6s
746:	learn: 0.2354058	total: 4.72s	remaining: 1.6s
747:	learn: 0.2353392	total: 4.72s	remaining: 1.59s
748:	learn: 0.2352817	total: 4.73s	remaining: 1.58s
749:	learn: 0.2352325	total: 4.74s	remaining: 1.58s
750:	learn: 0.2351765	total: 4.74s	remaining: 1.57s
751:	learn: 0.2350930	total: 4.75s	remaining: 1.56s
752:	learn: 0.2350414	total: 4.75s	remaining: 1.56s
753:	learn: 0.2349765	total: 4.76s	remaining: 1.55s
754:	learn: 0.2349296	total: 4.77s	remaining: 1.55s
755:	learn: 0.2348586	total: 4.77s	remaining: 1.54s
756:	learn: 0.2347831	total: 4.78s	remaining: 1.53s
757:	learn: 0.2347161	total: 4.79s	remaining: 1.53s

758:	learn: 0.2345511	total: 4.79s	remaining: 1.52s
759:	learn: 0.2344571	total: 4.8s	remaining: 1.51s
760:	learn: 0.2344288	total: 4.8s	remaining: 1.51s
761:	learn: 0.2343687	total: 4.81s	remaining: 1.5s
762:	learn: 0.2342749	total: 4.82s	remaining: 1.5s
763:	learn: 0.2342094	total: 4.82s	remaining: 1.49s
764:	learn: 0.2341608	total: 4.83s	remaining: 1.48s
765:	learn: 0.2340881	total: 4.84s	remaining: 1.48s
766:	learn: 0.2340276	total: 4.84s	remaining: 1.47s
767:	learn: 0.2339892	total: 4.85s	remaining: 1.47s
768:	learn: 0.2339423	total: 4.86s	remaining: 1.46s
769:	learn: 0.2338938	total: 4.86s	remaining: 1.45s
770:	learn: 0.2336964	total: 4.87s	remaining: 1.45s
771:	learn: 0.2336250	total: 4.87s	remaining: 1.44s
772:	learn: 0.2335590	total: 4.88s	remaining: 1.43s
773:	learn: 0.2334825	total: 4.89s	remaining: 1.43s
774:	learn: 0.2334219	total: 4.89s	remaining: 1.42s
775:	learn: 0.2333646	total: 4.9s	remaining: 1.41s
776:	learn: 0.2332788	total: 4.9s	remaining: 1.41s
777:	learn: 0.2332277	total: 4.91s	remaining: 1.4s
778:	learn: 0.2331811	total: 4.92s	remaining: 1.39s
779:	learn: 0.2330965	total: 4.92s	remaining: 1.39s
780:	learn: 0.2330133	total: 4.93s	remaining: 1.38s
781:	learn: 0.2329399	total: 4.94s	remaining: 1.38s
782:	learn: 0.2328938	total: 4.94s	remaining: 1.37s
783:	learn: 0.2328374	total: 4.95s	remaining: 1.36s
784:	learn: 0.2327445	total: 4.96s	remaining: 1.36s
785:	learn: 0.2326466	total: 4.96s	remaining: 1.35s
786:	learn: 0.2326024	total: 4.97s	remaining: 1.34s
787:	learn: 0.2324894	total: 4.97s	remaining: 1.34s
788:	learn: 0.2324066	total: 4.98s	remaining: 1.33s
789:	learn: 0.2323237	total: 4.99s	remaining: 1.32s
790:	learn: 0.2322340	total: 4.99s	remaining: 1.32s
791:	learn: 0.2320852	total: 5s	remaining: 1.31s
792:	learn: 0.2320587	total: 5.01s	remaining: 1.31s
793:	learn: 0.2320297	total: 5.01s	remaining: 1.3s
794:	learn: 0.2319652	total: 5.02s	remaining: 1.29s
795:	learn: 0.2318766	total: 5.02s	remaining: 1.29s
796:	learn: 0.2318297	total: 5.03s	remaining: 1.28s
797:	learn: 0.2317367	total: 5.04s	remaining: 1.27s
798:	learn: 0.2316809	total: 5.04s	remaining: 1.27s
799:	learn: 0.2315868	total: 5.05s	remaining: 1.26s
800:	learn: 0.2315204	total: 5.05s	remaining: 1.25s
801:	learn: 0.2314833	total: 5.06s	remaining: 1.25s
802:	learn: 0.2314281	total: 5.07s	remaining: 1.24s
803:	learn: 0.2313624	total: 5.07s	remaining: 1.24s
804:	learn: 0.2312753	total: 5.08s	remaining: 1.23s
805:	learn: 0.2311553	total: 5.08s	remaining: 1.22s

806:	learn: 0.2310921	total: 5.09s	remaining: 1.22s
807:	learn: 0.2310169	total: 5.1s	remaining: 1.21s
808:	learn: 0.2309759	total: 5.1s	remaining: 1.2s
809:	learn: 0.2309000	total: 5.11s	remaining: 1.2s
810:	learn: 0.2308211	total: 5.12s	remaining: 1.19s
811:	learn: 0.2307796	total: 5.12s	remaining: 1.19s
812:	learn: 0.2307301	total: 5.13s	remaining: 1.18s
813:	learn: 0.2305604	total: 5.13s	remaining: 1.17s
814:	learn: 0.2305217	total: 5.14s	remaining: 1.17s
815:	learn: 0.2305039	total: 5.14s	remaining: 1.16s
816:	learn: 0.2304133	total: 5.15s	remaining: 1.15s
817:	learn: 0.2303347	total: 5.16s	remaining: 1.15s
818:	learn: 0.2302789	total: 5.17s	remaining: 1.14s
819:	learn: 0.2302340	total: 5.17s	remaining: 1.14s
820:	learn: 0.2301928	total: 5.18s	remaining: 1.13s
821:	learn: 0.2301002	total: 5.18s	remaining: 1.12s
822:	learn: 0.2300324	total: 5.19s	remaining: 1.12s
823:	learn: 0.2299554	total: 5.2s	remaining: 1.11s
824:	learn: 0.2298069	total: 5.2s	remaining: 1.1s
825:	learn: 0.2296801	total: 5.21s	remaining: 1.1s
826:	learn: 0.2295831	total: 5.22s	remaining: 1.09s
827:	learn: 0.2295326	total: 5.22s	remaining: 1.08s
828:	learn: 0.2294043	total: 5.23s	remaining: 1.08s
829:	learn: 0.2293202	total: 5.24s	remaining: 1.07s
830:	learn: 0.2292573	total: 5.24s	remaining: 1.06s
831:	learn: 0.2291717	total: 5.25s	remaining: 1.06s
832:	learn: 0.2291218	total: 5.25s	remaining: 1.05s
833:	learn: 0.2290752	total: 5.26s	remaining: 1.05s
834:	learn: 0.2290012	total: 5.26s	remaining: 1.04s
835:	learn: 0.2289129	total: 5.27s	remaining: 1.03s
836:	learn: 0.2287635	total: 5.28s	remaining: 1.03s
837:	learn: 0.2286848	total: 5.28s	remaining: 1.02s
838:	learn: 0.2286533	total: 5.29s	remaining: 1.01s
839:	learn: 0.2285960	total: 5.3s	remaining: 1.01s
840:	learn: 0.2285215	total: 5.3s	remaining: 1s
841:	learn: 0.2284880	total: 5.31s	remaining: 996ms
842:	learn: 0.2284395	total: 5.31s	remaining: 990ms
843:	learn: 0.2283810	total: 5.32s	remaining: 983ms
844:	learn: 0.2283137	total: 5.33s	remaining: 977ms
845:	learn: 0.2282648	total: 5.33s	remaining: 971ms
846:	learn: 0.2282006	total: 5.34s	remaining: 964ms
847:	learn: 0.2281620	total: 5.34s	remaining: 958ms
848:	learn: 0.2281181	total: 5.35s	remaining: 952ms
849:	learn: 0.2280233	total: 5.36s	remaining: 945ms
850:	learn: 0.2279703	total: 5.36s	remaining: 939ms
851:	learn: 0.2278874	total: 5.37s	remaining: 933ms
852:	learn: 0.2277998	total: 5.38s	remaining: 927ms
853:	learn: 0.2277181	total: 5.38s	remaining: 920ms



854:	learn: 0.2276485	total: 5.39s	remaining: 914ms
855:	learn: 0.2275799	total: 5.4s	remaining: 908ms
856:	learn: 0.2274946	total: 5.4s	remaining: 902ms
857:	learn: 0.2274407	total: 5.41s	remaining: 895ms
858:	learn: 0.2274062	total: 5.42s	remaining: 889ms
859:	learn: 0.2273402	total: 5.42s	remaining: 883ms
860:	learn: 0.2272629	total: 5.43s	remaining: 876ms
861:	learn: 0.2271729	total: 5.44s	remaining: 870ms
862:	learn: 0.2271416	total: 5.44s	remaining: 864ms
863:	learn: 0.2271014	total: 5.45s	remaining: 858ms
864:	learn: 0.2269825	total: 5.46s	remaining: 851ms
865:	learn: 0.2269415	total: 5.46s	remaining: 845ms
866:	learn: 0.2268464	total: 5.47s	remaining: 839ms
867:	learn: 0.2267915	total: 5.47s	remaining: 833ms
868:	learn: 0.2266848	total: 5.48s	remaining: 826ms
869:	learn: 0.2265812	total: 5.49s	remaining: 820ms
870:	learn: 0.2265016	total: 5.49s	remaining: 814ms
871:	learn: 0.2264310	total: 5.5s	remaining: 807ms
872:	learn: 0.2263866	total: 5.51s	remaining: 801ms
873:	learn: 0.2263287	total: 5.51s	remaining: 795ms
874:	learn: 0.2262659	total: 5.52s	remaining: 789ms
875:	learn: 0.2262100	total: 5.53s	remaining: 782ms
876:	learn: 0.2261557	total: 5.53s	remaining: 776ms
877:	learn: 0.2261018	total: 5.54s	remaining: 770ms
878:	learn: 0.2260357	total: 5.55s	remaining: 764ms
879:	learn: 0.2259866	total: 5.55s	remaining: 757ms
880:	learn: 0.2258871	total: 5.56s	remaining: 751ms
881:	learn: 0.2258322	total: 5.57s	remaining: 745ms
882:	learn: 0.2257858	total: 5.58s	remaining: 739ms
883:	learn: 0.2257631	total: 5.58s	remaining: 733ms
884:	learn: 0.2256987	total: 5.59s	remaining: 726ms
885:	learn: 0.2256482	total: 5.59s	remaining: 720ms
886:	learn: 0.2255352	total: 5.6s	remaining: 714ms
887:	learn: 0.2254570	total: 5.61s	remaining: 707ms
888:	learn: 0.2254047	total: 5.61s	remaining: 701ms
889:	learn: 0.2253315	total: 5.62s	remaining: 695ms
890:	learn: 0.2252782	total: 5.63s	remaining: 688ms
891:	learn: 0.2252309	total: 5.63s	remaining: 682ms
892:	learn: 0.2251848	total: 5.64s	remaining: 676ms
893:	learn: 0.2251094	total: 5.64s	remaining: 669ms
894:	learn: 0.2250472	total: 5.65s	remaining: 663ms
895:	learn: 0.2249581	total: 5.66s	remaining: 657ms
896:	learn: 0.2248783	total: 5.66s	remaining: 650ms
897:	learn: 0.2248290	total: 5.67s	remaining: 644ms
898:	learn: 0.2247592	total: 5.68s	remaining: 638ms
899:	learn: 0.2246687	total: 5.68s	remaining: 631ms
900:	learn: 0.2246037	total: 5.69s	remaining: 625ms
901:	learn: 0.2244631	total: 5.7s	remaining: 619ms

902:	learn: 0.2243602	total: 5.7s	remaining: 613ms
903:	learn: 0.2242553	total: 5.71s	remaining: 606ms
904:	learn: 0.2242066	total: 5.71s	remaining: 600ms
905:	learn: 0.2241182	total: 5.72s	remaining: 594ms
906:	learn: 0.2240614	total: 5.73s	remaining: 587ms
907:	learn: 0.2239784	total: 5.74s	remaining: 581ms
908:	learn: 0.2239348	total: 5.74s	remaining: 575ms
909:	learn: 0.2238844	total: 5.75s	remaining: 568ms
910:	learn: 0.2238298	total: 5.76s	remaining: 562ms
911:	learn: 0.2237131	total: 5.76s	remaining: 556ms
912:	learn: 0.2236696	total: 5.77s	remaining: 550ms
913:	learn: 0.2235361	total: 5.78s	remaining: 543ms
914:	learn: 0.2234311	total: 5.78s	remaining: 537ms
915:	learn: 0.2233472	total: 5.79s	remaining: 531ms
916:	learn: 0.2233044	total: 5.79s	remaining: 524ms
917:	learn: 0.2232525	total: 5.8s	remaining: 518ms
918:	learn: 0.2231670	total: 5.81s	remaining: 512ms
919:	learn: 0.2230982	total: 5.81s	remaining: 505ms
920:	learn: 0.2230076	total: 5.82s	remaining: 499ms
921:	learn: 0.2229783	total: 5.83s	remaining: 493ms
922:	learn: 0.2229332	total: 5.83s	remaining: 486ms
923:	learn: 0.2228877	total: 5.84s	remaining: 480ms
924:	learn: 0.2228038	total: 5.84s	remaining: 474ms
925:	learn: 0.2227426	total: 5.85s	remaining: 467ms
926:	learn: 0.2226908	total: 5.85s	remaining: 461ms
927:	learn: 0.2226416	total: 5.86s	remaining: 455ms
928:	learn: 0.2225894	total: 5.87s	remaining: 448ms
929:	learn: 0.2225558	total: 5.87s	remaining: 442ms
930:	learn: 0.2224838	total: 5.88s	remaining: 436ms
931:	learn: 0.2224475	total: 5.89s	remaining: 429ms
932:	learn: 0.2223125	total: 5.89s	remaining: 423ms
933:	learn: 0.2222342	total: 5.9s	remaining: 417ms
934:	learn: 0.2221635	total: 5.91s	remaining: 411ms
935:	learn: 0.2221228	total: 5.91s	remaining: 404ms
936:	learn: 0.2220886	total: 5.92s	remaining: 398ms
937:	learn: 0.2220575	total: 5.92s	remaining: 392ms
938:	learn: 0.2220034	total: 5.93s	remaining: 385ms
939:	learn: 0.2219595	total: 5.94s	remaining: 379ms
940:	learn: 0.2219188	total: 5.94s	remaining: 373ms
941:	learn: 0.2218422	total: 5.95s	remaining: 366ms
942:	learn: 0.2217656	total: 5.96s	remaining: 360ms
943:	learn: 0.2216008	total: 5.96s	remaining: 354ms
944:	learn: 0.2214924	total: 5.97s	remaining: 348ms
945:	learn: 0.2213970	total: 5.98s	remaining: 341ms
946:	learn: 0.2213445	total: 5.98s	remaining: 335ms
947:	learn: 0.2212966	total: 5.99s	remaining: 329ms
948:	learn: 0.2212422	total: 6s	remaining: 322ms
949:	learn: 0.2211354	total: 6s	remaining: 316ms

950:	learn: 0.2210466	total: 6.01s	remaining: 310ms
951:	learn: 0.2209600	total: 6.01s	remaining: 303ms
952:	learn: 0.2208328	total: 6.02s	remaining: 297ms
953:	learn: 0.2207294	total: 6.03s	remaining: 291ms
954:	learn: 0.2206322	total: 6.03s	remaining: 284ms
955:	learn: 0.2205823	total: 6.04s	remaining: 278ms
956:	learn: 0.2205090	total: 6.05s	remaining: 272ms
957:	learn: 0.2204443	total: 6.05s	remaining: 265ms
958:	learn: 0.2203998	total: 6.06s	remaining: 259ms
959:	learn: 0.2203143	total: 6.06s	remaining: 253ms
960:	learn: 0.2202174	total: 6.07s	remaining: 246ms
961:	learn: 0.2201329	total: 6.08s	remaining: 240ms
962:	learn: 0.2200856	total: 6.08s	remaining: 234ms
963:	learn: 0.2200297	total: 6.09s	remaining: 227ms
964:	learn: 0.2199715	total: 6.09s	remaining: 221ms
965:	learn: 0.2199200	total: 6.1s	remaining: 215ms
966:	learn: 0.2198741	total: 6.11s	remaining: 208ms
967:	learn: 0.2198016	total: 6.11s	remaining: 202ms
968:	learn: 0.2197750	total: 6.12s	remaining: 196ms
969:	learn: 0.2197160	total: 6.12s	remaining: 189ms
970:	learn: 0.2196797	total: 6.13s	remaining: 183ms
971:	learn: 0.2196137	total: 6.14s	remaining: 177ms
972:	learn: 0.2195655	total: 6.15s	remaining: 171ms
973:	learn: 0.2195318	total: 6.15s	remaining: 164ms
974:	learn: 0.2194915	total: 6.16s	remaining: 158ms
975:	learn: 0.2194470	total: 6.17s	remaining: 152ms
976:	learn: 0.2194007	total: 6.17s	remaining: 145ms
977:	learn: 0.2193533	total: 6.18s	remaining: 139ms
978:	learn: 0.2192902	total: 6.18s	remaining: 133ms
979:	learn: 0.2192551	total: 6.19s	remaining: 126ms
980:	learn: 0.2192169	total: 6.2s	remaining: 120ms
981:	learn: 0.2191326	total: 6.2s	remaining: 114ms
982:	learn: 0.2191119	total: 6.21s	remaining: 107ms
983:	learn: 0.2190611	total: 6.21s	remaining: 101ms
984:	learn: 0.2190061	total: 6.22s	remaining: 94.7ms
985:	learn: 0.2189738	total: 6.23s	remaining: 88.4ms
986:	learn: 0.2189136	total: 6.23s	remaining: 82.1ms
987:	learn: 0.2188856	total: 6.24s	remaining: 75.8ms
988:	learn: 0.2188403	total: 6.25s	remaining: 69.5ms
989:	learn: 0.2187798	total: 6.25s	remaining: 63.1ms
990:	learn: 0.2187109	total: 6.26s	remaining: 56.8ms
991:	learn: 0.2186928	total: 6.26s	remaining: 50.5ms
992:	learn: 0.2186679	total: 6.27s	remaining: 44.2ms
993:	learn: 0.2185689	total: 6.28s	remaining: 37.9ms
994:	learn: 0.2185099	total: 6.28s	remaining: 31.6ms
995:	learn: 0.2184551	total: 6.29s	remaining: 25.3ms
996:	learn: 0.2184051	total: 6.29s	remaining: 18.9ms
997:	learn: 0.2183162	total: 6.3s	remaining: 12.6ms

```

998:   learn: 0.2182232      total: 6.31s   remaining: 6.31ms
999:   learn: 0.2181500      total: 6.31s   remaining: 0us

```

```

[103]: VotingClassifier(estimators=[('gaussian', GaussianNB()),
                                   ('Gridlogistic',
                                   GridSearchCV(cv=RepeatedStratifiedKFold(n_repeats=3, n_splits=10,
                                   random_state=1),
                                   error_score=0,
                                   estimator=LogisticRegression(),
                                   n_jobs=-1,
                                   param_grid={'C': [100, 10, 1.0, 0.1,
                                   0.01],
                                   'penalty': ['l2'],
                                   'solver': ['newton-cg',
                                   'lbfgs',
                                   'liblinear']}},
                                   scoring='accuracy')),
      ('catboost_classifier',
      <...
      n_estimators=494, n_jobs=None,
      num_parallel_tree=None,
      random_state=None, reg_alpha=None,
      reg_lambda=None,
      scale_pos_weight=None,
      subsample=0.7, tree_method=None,
      validate_parameters=None,
      verbosity=0)),
      ('LGBMclassifier',
      LGBMClassifier(boosting_type='dart',
                      importance_type='gain', max_bin=60,
                      max_depth=5, n_estimators=494,
                      num_leaves=300, verbosity=-1))],
      voting='soft')

```

```
[104]: y_pred = vot_soft.predict(X_test)
```

```
[105]: metrics.accuracy_score(y_test, y_pred)*100
```

```
[105]: 90.02570694087404
```

```

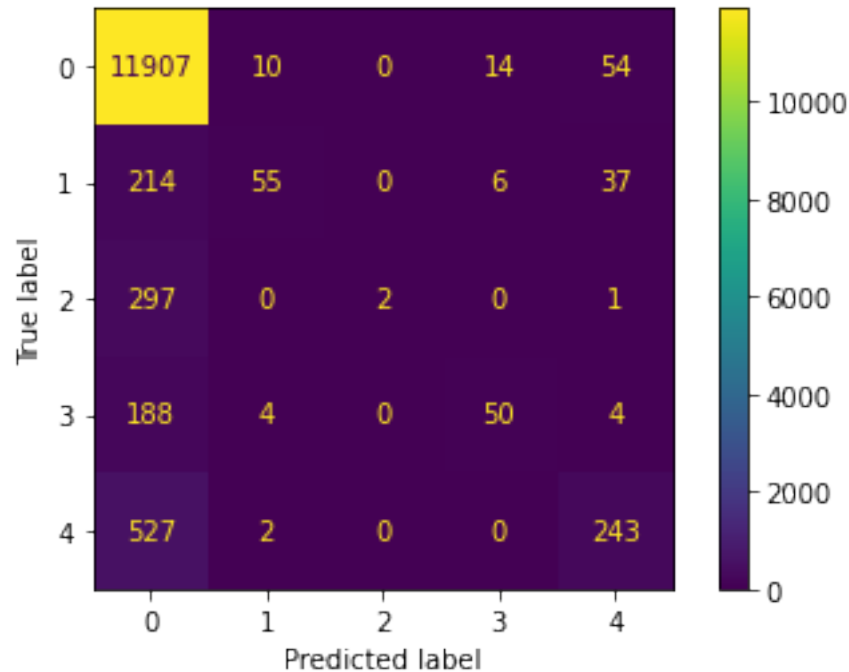
[106]: t = confusion_matrix(y_test, y_pred)
      disp = ConfusionMatrixDisplay(confusion_matrix= t, display_labels= vot_soft.
      ↪classes_)
      disp.plot()

```

```

[106]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
      0x7fa3c372ba60>

```



```
[107]: #metrics.accuracy_score(y_test, y_pred_gnb)*100
```

```
[108]: #confusion_matrix(y_test, y_pred_gnb)
```

```
[109]: #t = confusion_matrix(y_test, y_pred_gnb)
#disp = ConfusionMatrixDisplay(confusion_matrix= t, display_labels= gnb.
    ↳ classes_)
```

```
[110]: #disp.plot()
```

```
[111]: #metrics.accuracy_score(y_test, y_pred_log)*100
```

```
[112]: #t = confusion_matrix(y_test, y_pred_log)
#disp = ConfusionMatrixDisplay(confusion_matrix= t, display_labels= grid_search.
    ↳ classes_)
#disp.plot()
```

```
[113]: #metrics.accuracy_score(y_test, y_pred_cat)*100
```

```
[114]: #t = confusion_matrix(y_test, y_pred_cat)
#disp = ConfusionMatrixDisplay(confusion_matrix= t, display_labels= cat.
    ↳ classes_)
#disp.plot()
```

```
[115]: #metrics.accuracy_score(y_test, y_pred_dt)*100
```

```
[116]: #t = confusion_matrix(y_test, y_pred_dt)
#disp = ConfusionMatrixDisplay(confusion_matrix= t, display_labels= dtclf.
      ↳classes_)
#disp.plot()
```

## 9 TESTING DATA

```
[117]: path = '/media/mr-robot/Local Disk/summer_training/test'
os.chdir(path)
```

```
[118]: # Converting all las files in csv by iterating using lasio
for file in os.listdir():
    if file.endswith(".las"):
        file_path = f"{path}/{file}"
        las=lasio.read(file_path)
        size=len(file_path)
        filepath1=file_path[:size-3]
        las.to_csv(filepath1+'csv', units=False)
```

```
[119]: ## To avoid further merging data and redundancy
if(os.path.isfile('./merged_data.csv')):
    os.remove("merged_data.csv")

if(os.path.isfile('./FACIES_imputed.csv')):
    os.remove("FACIES_imputed.csv")

if(os.path.isfile('./FACIES_TRAIN.csv')):
    os.remove("FACIES_TRAIN.csv")
```

```
[120]: # Merging all Well Log using Glob
filenames = glob.glob(path + "/*.csv")
dfs = []
for filename in filenames:
    dfs.append(pd.read_csv(filename))
big_frame = pd.concat(dfs, ignore_index=True)
big_frame.to_csv('merged_data.csv', index=False)
```

```
[121]: df = pd.read_csv('merged_data.csv')
df
```

```
[121]:
```

	DEPTH	ACOUSTICIMPEDANCE1	AI	AVG_PIGN	BIT	CALI	\
0	1197.4072	5252.3882	5252388.0	NaN	0.2159	8.9012	
1	1197.5596	5289.7070	5289707.0	NaN	0.2159	8.9005	
2	1197.7120	5245.4429	5245443.0	NaN	0.2159	8.8957	
3	1197.8644	5181.9023	5181902.5	NaN	0.2159	8.8932	
4	1198.0168	5131.1343	5131134.5	NaN	0.2159	8.8980	
...	...	...	...	...	...	...	

29560	1689.5065		6013.4722	6013472.5	NaN	0.2159	NaN
29561	1689.6589		5953.0059	5953006.0	NaN	0.2159	NaN
29562	1689.8113		5954.4824	5954482.0	NaN	0.2159	NaN
29563	1689.9637		5911.3301	5911330.0	NaN	0.2159	NaN
29564	1690.1161		5930.9585	5930958.5	NaN	0.2159	NaN

	NPHI	DT	FACIES	FLD1	...	SPSD	ZCOR	BS	CALI[DERIVED]1	\
0	0.4682	133.4417	NaN	NaN	...	NaN	NaN	NaN		NaN
1	0.4585	132.4196	NaN	NaN	...	NaN	NaN	NaN		NaN
2	0.4543	133.3569	NaN	NaN	...	NaN	NaN	NaN		NaN
3	0.4827	134.7392	NaN	NaN	...	NaN	NaN	NaN		NaN
4	0.5361	135.7694	NaN	NaN	...	NaN	NaN	NaN		NaN
...	...	...	...	...	...	...	...	...		...
29560	NaN	126.6800	NaN	NaN	...	NaN	NaN	NaN		NaN
29561	NaN	127.9872	NaN	NaN	...	NaN	NaN	NaN		NaN
29562	NaN	127.9657	NaN	NaN	...	NaN	NaN	NaN		NaN
29563	NaN	128.9050	NaN	NaN	...	NaN	NaN	NaN		NaN
29564	NaN	128.4784	NaN	NaN	...	NaN	NaN	NaN		NaN

	DFL	GRCO	HDRS	HMRS	PHIT	TEMP1
0	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...
29560	NaN	NaN	NaN	NaN	NaN	NaN
29561	NaN	NaN	NaN	NaN	NaN	NaN
29562	NaN	NaN	NaN	NaN	NaN	NaN
29563	NaN	NaN	NaN	NaN	NaN	NaN
29564	NaN	NaN	NaN	NaN	NaN	NaN

[29565 rows x 55 columns]

```
[122]: #Selecting required feature
df=df[["DT","GR","NPHI","RHOB","FACIES"]]
```

```
[123]: df
```

```
[123]:
```

	DT	GR	NPHI	RHOB	FACIES
0	133.4417	87.3154	0.4682	2.2995	NaN
1	132.4196	88.5412	0.4585	2.2981	NaN
2	133.3569	87.5764	0.4543	2.2950	NaN
3	134.7392	86.0361	0.4827	2.2907	NaN
4	135.7694	85.0393	0.5361	2.2856	NaN
...	...	...	...	...	...
29560	126.6800	NaN	NaN	2.4993	NaN

29561	127.9872	NaN	NaN	2.4997	NaN
29562	127.9657	NaN	NaN	2.4999	NaN
29563	128.9050	NaN	NaN	2.5000	NaN
29564	128.4784	NaN	NaN	2.5000	NaN

[29565 rows x 5 columns]

```
[124]: df=imputing(imputation_strategy[optionimputation],df)
df
```

```
[124]:
```

	DT	GR	NPHI	RHOB	FACIES
0	133.4417	87.315400	0.468200	2.2995	0
1	132.4196	88.541200	0.458500	2.2981	0
2	133.3569	87.576400	0.454300	2.2950	0
3	134.7392	86.036100	0.482700	2.2907	0
4	135.7694	85.039300	0.536100	2.2856	0
...	...	...	...	...	...
29560	126.6800	78.920533	0.530600	2.4993	0
29561	127.9872	84.668033	0.500300	2.4997	1
29562	127.9657	96.542533	0.524533	2.4999	1
29563	128.9050	95.799167	0.501867	2.5000	0
29564	128.4784	122.594467	0.606433	2.5000	0

[29565 rows x 5 columns]

```
[125]: df = outliers(DATAConditioningStrategy[optionoutlier] , df,
↳DATAConditioningColumns)
```

column DT

Percentiles: 25th=114.139, 75th=137.342, IQR=23.202

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
2632	77.7408	55.287400	0.3062	2.6430	0
2633	77.3217	53.629600	0.3052	2.5920	0
3981	75.3027	73.368300	0.5153	2.5090	0
3982	73.6734	73.261800	0.5041	2.4475	0
6097	79.0923	87.085800	0.3700	2.8019	0
6110	76.3801	96.356900	0.3313	2.7004	0
6406	78.6538	59.692300	0.4038	2.6646	0
6448	79.3029	64.718200	0.3632	2.7212	0
13938	79.2984	108.679600	0.4490	2.8759	0
13939	70.9828	95.723000	0.4255	3.0317	0
13940	75.5917	94.711500	0.4245	2.9428	0
15679	175.1408	42.799200	0.5044	2.3501	0
15680	173.8879	42.799200	0.4875	2.3948	0
15706	172.7409	42.799200	0.5074	2.4185	0
15707	174.8540	42.799200	0.4967	2.4147	0
15708	172.7833	42.799200	0.4784	2.4165	0



16123	76.3119	88.145567	0.3927	3.0026	0
16907	173.0850	42.799200	0.6734	1.8918	0
23404	72.9019	86.674800	0.3879	2.6145	0
23405	73.6668	86.070200	0.3612	2.5231	0
25171	79.3205	78.216300	0.5893	2.2124	0
28926	78.1889	66.276900	0.4540	2.9479	0

(22, 5)

	DT	GR	NPHI	RHOB	FACIES
0	133.4417	87.315400	0.468200	2.2995	0
1	132.4196	88.541200	0.458500	2.2981	0
2	133.3569	87.576400	0.454300	2.2950	0
3	134.7392	86.036100	0.482700	2.2907	0
4	135.7694	85.039300	0.536100	2.2856	0
...	...	...	...	...	...
29560	126.6800	78.920533	0.530600	2.4993	0
29561	127.9872	84.668033	0.500300	2.4997	1
29562	127.9657	96.542533	0.524533	2.4999	1
29563	128.9050	95.799167	0.501867	2.5000	0
29564	128.4784	122.594467	0.606433	2.5000	0

[29543 rows x 5 columns]

column GR

Percentiles: 25th=76.837, 75th=103.425, IQR=26.588

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
1342	144.1047	35.5685	0.6130	1.2752	3
1625	149.5008	36.3442	0.6133	1.1143	3
1626	150.9417	29.3642	0.6122	1.0951	3
1627	149.6250	27.0870	0.6129	1.1136	3
1628	148.6148	28.5975	0.6274	1.2056	3
...	...	...	...	...	...
28969	151.0522	27.8672	0.7510	1.0626	3
28970	152.6379	27.9862	0.7093	1.0935	3
28971	154.5247	28.4657	0.6571	1.1246	3
28972	155.4262	29.3424	0.6296	1.1451	3
28973	154.5691	32.9489	0.6210	1.1474	3

[1842 rows x 5 columns]

(1842, 5)

	DT	GR	NPHI	RHOB	FACIES
0	133.4417	87.315400	0.468200	2.2995	0
1	132.4196	88.541200	0.458500	2.2981	0
2	133.3569	87.576400	0.454300	2.2950	0
3	134.7392	86.036100	0.482700	2.2907	0
4	135.7694	85.039300	0.536100	2.2856	0
...	...	...	...	...	...
29560	126.6800	78.920533	0.530600	2.4993	0
29561	127.9872	84.668033	0.500300	2.4997	1

29562	127.9657	96.542533	0.524533	2.4999	1
29563	128.9050	95.799167	0.501867	2.5000	0
29564	128.4784	122.594467	0.606433	2.5000	0

[27701 rows x 5 columns]

column NPHI

Percentiles: 25th=0.467, 75th=0.550, IQR=0.082

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
263	143.7784	72.7236	0.6766	2.1787	0
513	138.5944	75.0486	0.6775	2.2283	0
644	143.2483	78.0601	0.6805	1.9364	0
645	144.5881	78.3862	0.6749	1.7739	3
647	148.7089	60.5277	0.6966	1.3747	3
...	...	...	...	...	...
29028	105.8965	77.9666	0.2990	1.9829	1
29029	105.0871	73.8077	0.2886	1.9849	1
29030	105.3242	68.5815	0.2919	1.9918	1
29031	107.1987	64.1699	0.3269	2.0025	1
29038	113.2466	74.4795	0.3385	1.9897	1

[1510 rows x 5 columns]

(1510, 5)

	DT	GR	NPHI	RHOB	FACIES
0	133.4417	87.315400	0.468200	2.2995	0
1	132.4196	88.541200	0.458500	2.2981	0
2	133.3569	87.576400	0.454300	2.2950	0
3	134.7392	86.036100	0.482700	2.2907	0
4	135.7694	85.039300	0.536100	2.2856	0
...	...	...	...	...	...
29560	126.6800	78.920533	0.530600	2.4993	0
29561	127.9872	84.668033	0.500300	2.4997	1
29562	127.9657	96.542533	0.524533	2.4999	1
29563	128.9050	95.799167	0.501867	2.5000	0
29564	128.4784	122.594467	0.606433	2.5000	0

[26191 rows x 5 columns]

column RHOB

Percentiles: 25th=2.206, 75th=2.416, IQR=0.210

InterQuartile Range Outliers-:

	DT	GR	NPHI	RHOB	FACIES
646	146.9913	72.1231	0.6718	1.5568	3
1228	130.3615	77.5789	0.5451	1.6171	3
1229	133.5854	69.1480	0.5995	1.4461	3
1230	137.1125	58.9514	0.6035	1.4420	3
1231	139.1413	55.2131	0.5432	1.5727	3
...	...	...	...	...	...
29074	133.7901	78.6751	0.5387	1.8071	0

29125	96.3199	80.4237	0.4219	2.7614	0
29181	131.6097	94.2842	0.4822	1.8686	0
29182	130.0865	81.8287	0.4741	1.7645	0
29183	124.4891	75.3927	0.4875	1.7919	0

[1476 rows x 5 columns]  
(1476, 5)

	DT	GR	NPHI	RHOB	FACIES
0	133.4417	87.315400	0.468200	2.2995	0
1	132.4196	88.541200	0.458500	2.2981	0
2	133.3569	87.576400	0.454300	2.2950	0
3	134.7392	86.036100	0.482700	2.2907	0
4	135.7694	85.039300	0.536100	2.2856	0
...	...	...	...	...	...
29560	126.6800	78.920533	0.530600	2.4993	0
29561	127.9872	84.668033	0.500300	2.4997	1
29562	127.9657	96.542533	0.524533	2.4999	1
29563	128.9050	95.799167	0.501867	2.5000	0
29564	128.4784	122.594467	0.606433	2.5000	0

[24715 rows x 5 columns]

```
[126]: df = data_scaling( scaling_strategy[optionscaling] , df ,  
    ↪DATAConditioningColumns )
```

```
[127]: df.to_csv("testing_preprocessed.csv",index=False)
```

```
[128]: df=pd.read_csv('testing_preprocessed.csv')
```

```
[129]: df
```

```
[129]:
```

	DT	GR	NPHI	RHOB	FACIES
0	0.419198	-0.251613	-0.555556	-0.0715	0
1	0.370230	-0.197525	-0.686992	-0.0785	0
2	0.415135	-0.240096	-0.743902	-0.0940	0
3	0.481360	-0.308062	-0.359079	-0.1155	0
4	0.530716	-0.352045	0.364499	-0.1410	0
...	...	...	...	...	...
24710	0.095252	-0.622034	0.289973	0.9275	0
24711	0.157879	-0.368427	-0.120596	0.9295	1
24712	0.156849	0.155533	0.207769	0.9305	1
24713	0.201850	0.122732	-0.099368	0.9310	0
24714	0.181412	1.305068	1.317525	0.9310	0

[24715 rows x 5 columns]

```
[130]: X_testing=df[["DT","GR","NPHI","RHOB"]]  
    y_testing=df[["FACIES"]]
```

```
[131]: X_testing.isnull().sum()
```

```
[131]: DT      0
      GR      0
      NPHI    0
      RHOB    0
      dtype: int64
```

```
[132]: #X_testing=FeatureSelection(FeatureSelectionStrategy[optionfeature],X_testing,y_testing)
```

```
[ ]:
```

```
[133]: X_testing
```

```
[133]:
```

	DT	GR	NPHI	RHOB
0	0.419198	-0.251613	-0.555556	-0.0715
1	0.370230	-0.197525	-0.686992	-0.0785
2	0.415135	-0.240096	-0.743902	-0.0940
3	0.481360	-0.308062	-0.359079	-0.1155
4	0.530716	-0.352045	0.364499	-0.1410
...	...	...	...	...
24710	0.095252	-0.622034	0.289973	0.9275
24711	0.157879	-0.368427	-0.120596	0.9295
24712	0.156849	0.155533	0.207769	0.9305
24713	0.201850	0.122732	-0.099368	0.9310
24714	0.181412	1.305068	1.317525	0.9310

```
[24715 rows x 4 columns]
```

```
[134]: y_testing.describe()
```

```
[134]:
```

	FACIES
count	24715.000000
mean	0.334776
std	1.018206
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	4.000000

```
[135]: y_predicted = vot_soft.predict(X_testing)
```

```
[136]: y_predicted
```

```
[136]: array([0, 0, 0, ..., 0, 0, 0])
```

```
[137]: metrics.accuracy_score(y_testing, y_predicted)*100
```

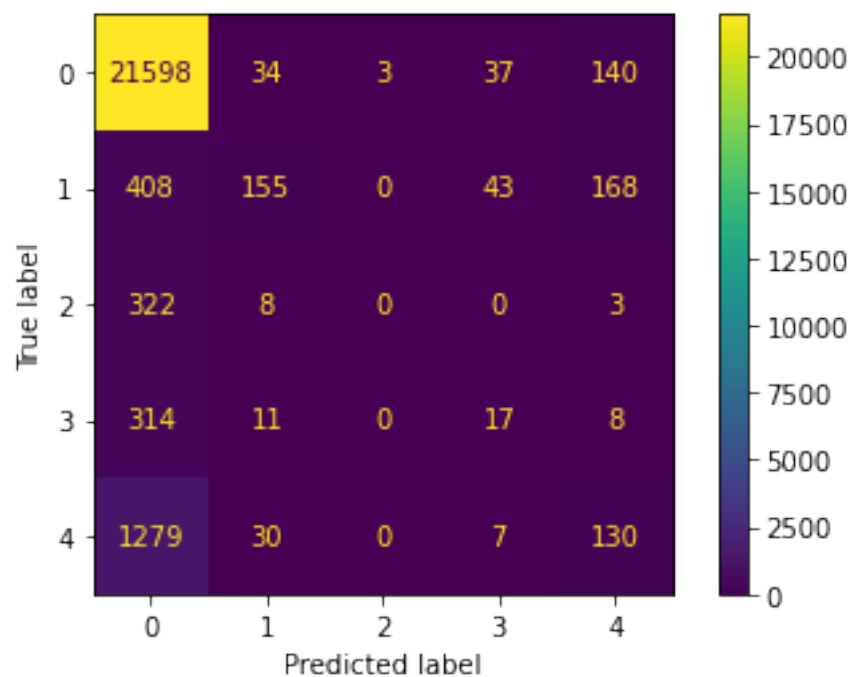
```
[137]: 88.61015577584462
```

```
[138]: confusion_matrix(y_testing, y_predicted)
```

```
[138]: array([[21598,   34,    3,   37,  140],
        [ 408,  155,   0,   43,  168],
        [ 322,    8,   0,    0,    3],
        [ 314,   11,   0,   17,    8],
        [1279,   30,   0,    7,  130]])
```

```
[139]: t = confusion_matrix(y_testing, y_predicted)
disp = ConfusionMatrixDisplay(confusion_matrix= t, display_labels= vot_soft.
↪classes_)
disp.plot()
```

```
[139]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7fa30482dac0>
```



```
[140]: t1=pd.DataFrame(y_testing)
```

```
[141]: t1.to_csv('y_given.csv',index=False)
```

```
[142]: t2=pd.DataFrame(y_predicted)
```

```
[143]: t2.to_csv('y_predicted.csv',index=False)
```

[ ]: