main_new_knn_5_2_0_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
NPHI        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	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
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
```

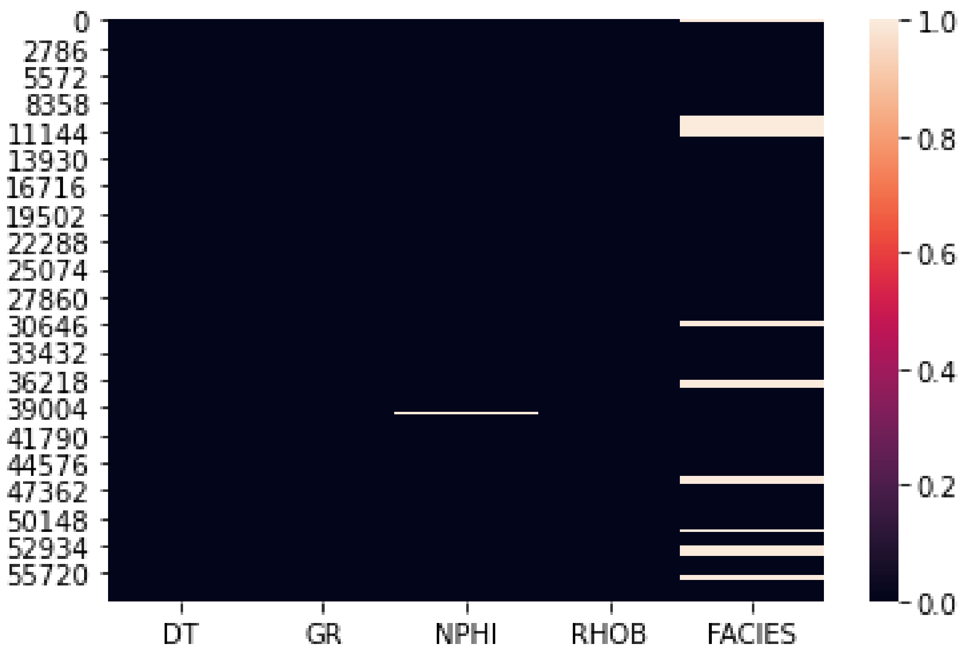
[28]:
```
(58499, 5)
```

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

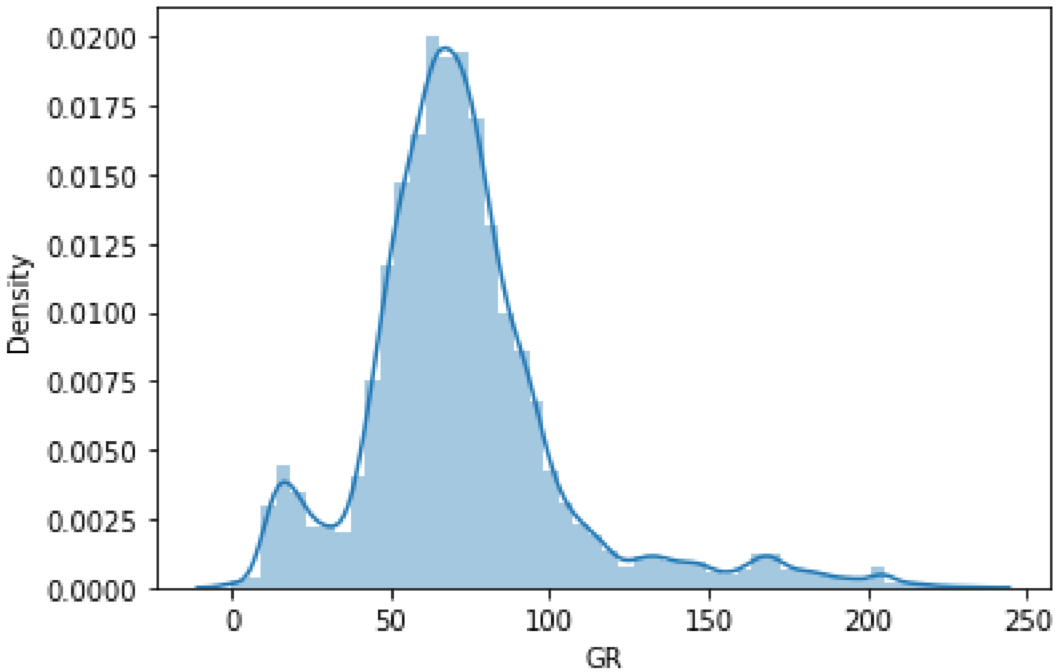
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 58499 entries, 0 to 58498
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   DT      58499 non-null  float64
 1   GR      58379 non-null  float64
 2   NPHI    58172 non-null  float64
 3 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



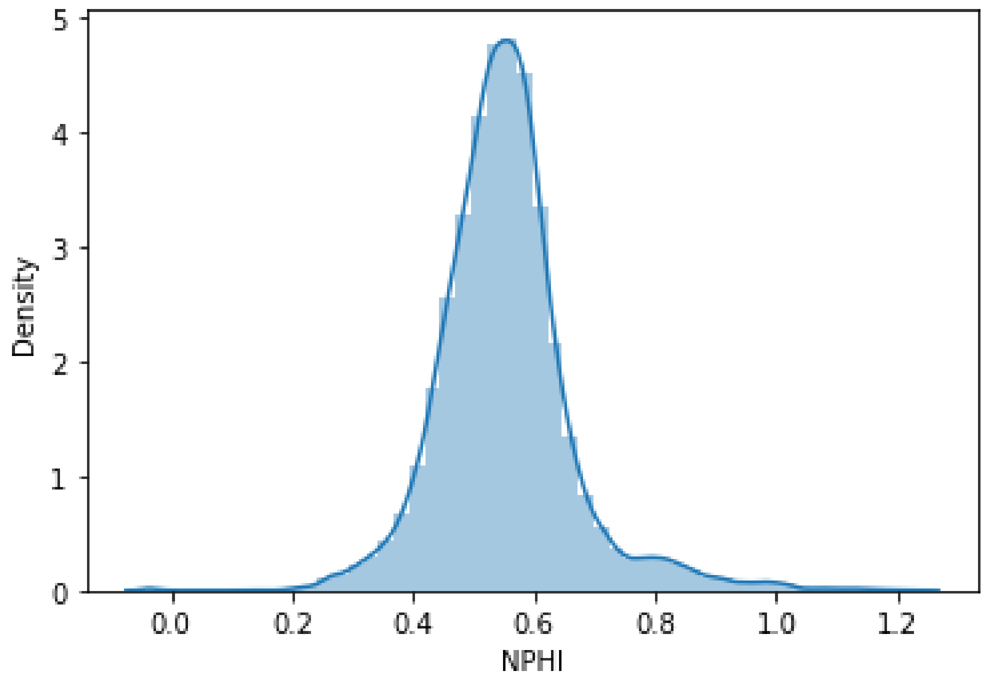
[31]:
```
null_gr = sns.distplot(df.GR.dropna())
```



[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())
```

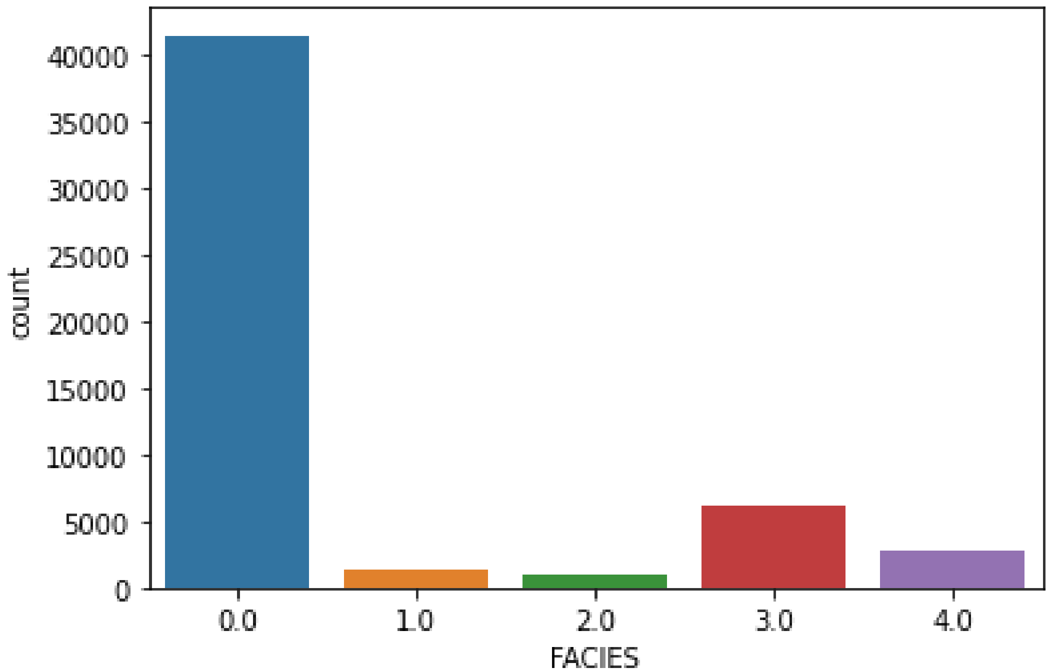


[34]:
```
df.NPHI.describe()
```

[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 == "bffill":
        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 beysian
        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" , "bffill" , "KNNImputer" , "IterativeImputer" ,␣
 ↪"KNNimputer_floor" , "IterativeImputer_floor" , "KNNBinning","dropna"]
#select option from 0-7 (6 is experimental)
optionimputation=5
```

```
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   NPHI    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
```

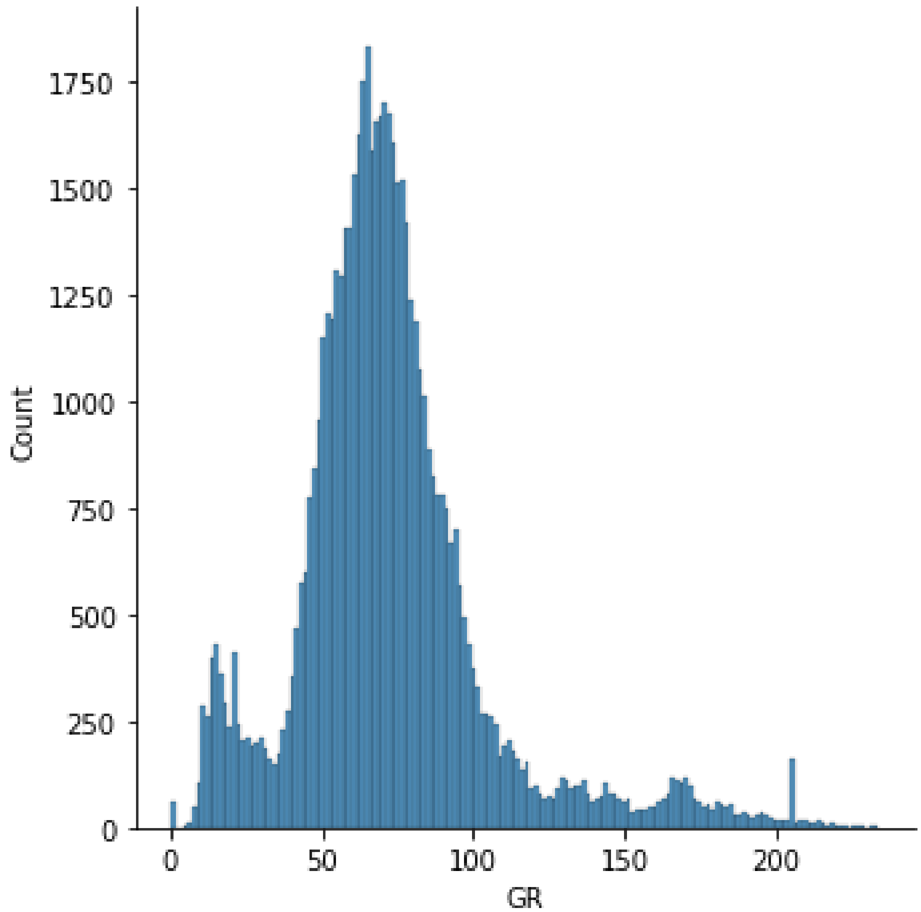
[41]:
```
df.isnull().sum()
```

[41]:
```
DT        0
GR        0
NPHI      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())
```

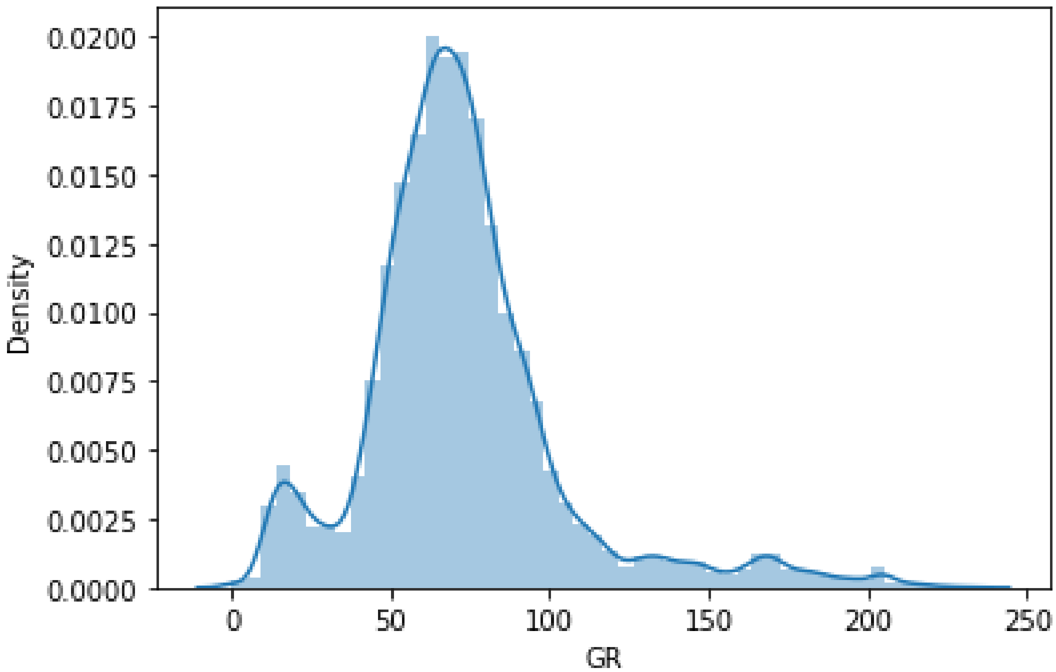
[43]:
```
<seaborn.axisgrid.FacetGrid at 0x7f43f16a3310>
```



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

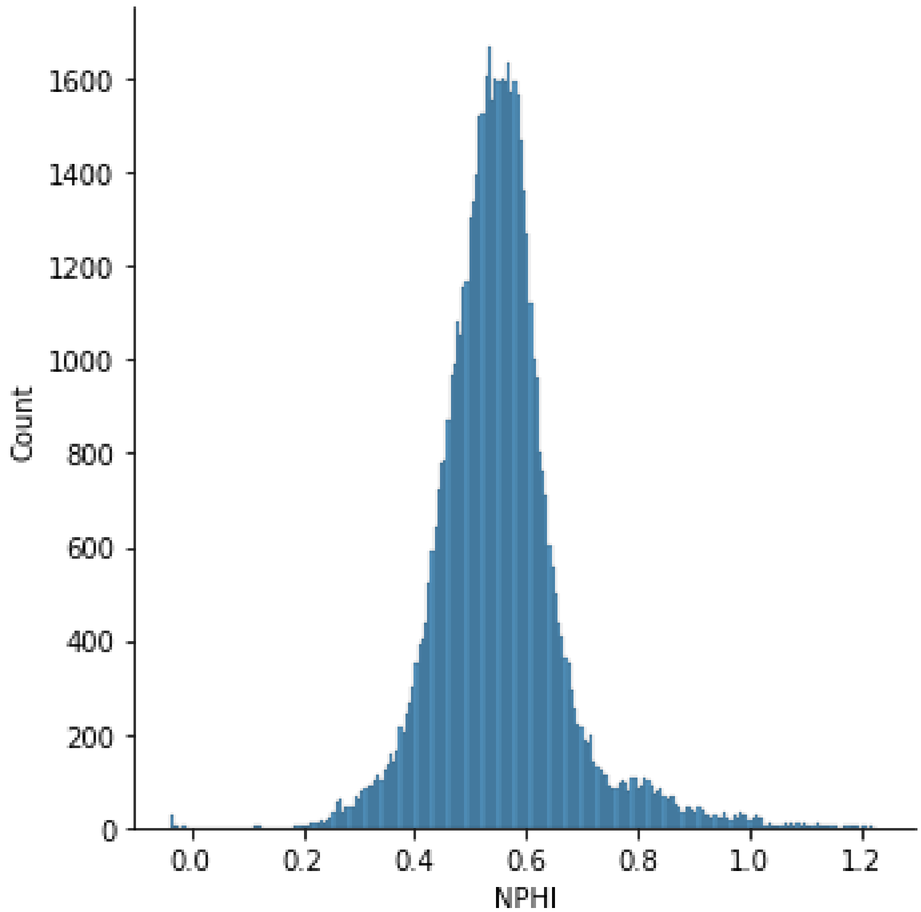
```
WHEN GR WAS NULL
```

[44]:



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

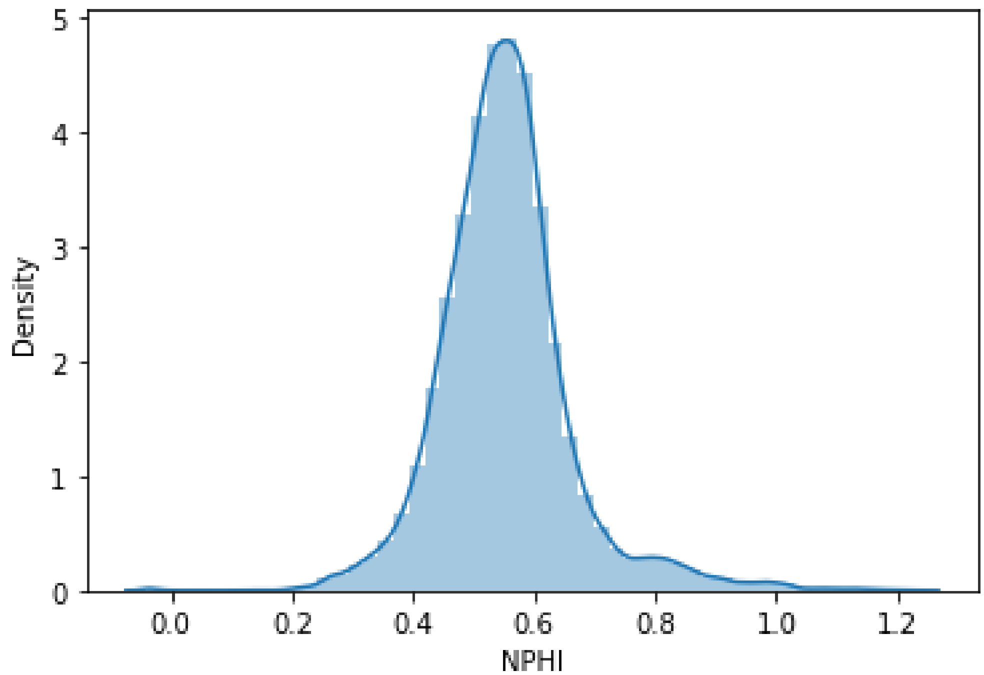
[45]: <seaborn.axisgrid.FacetGrid at 0x7f43f1b47760>



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

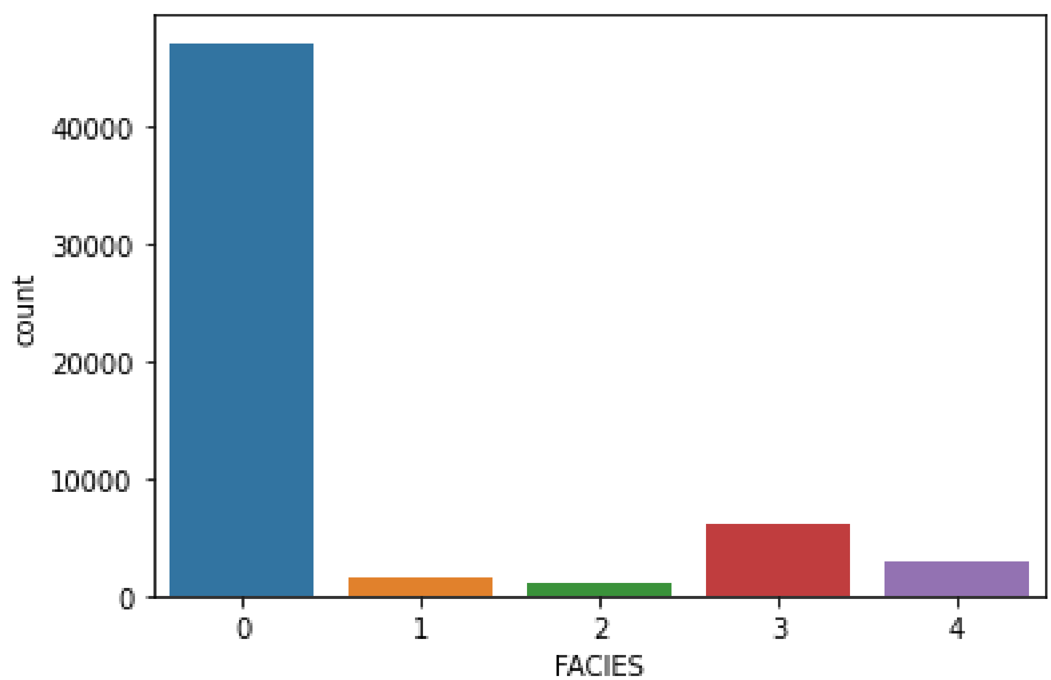
WHEN NPHI WAS NULL

[46]:



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

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



4 DATA CONDITIONING / OUTLIER REMOVAL

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

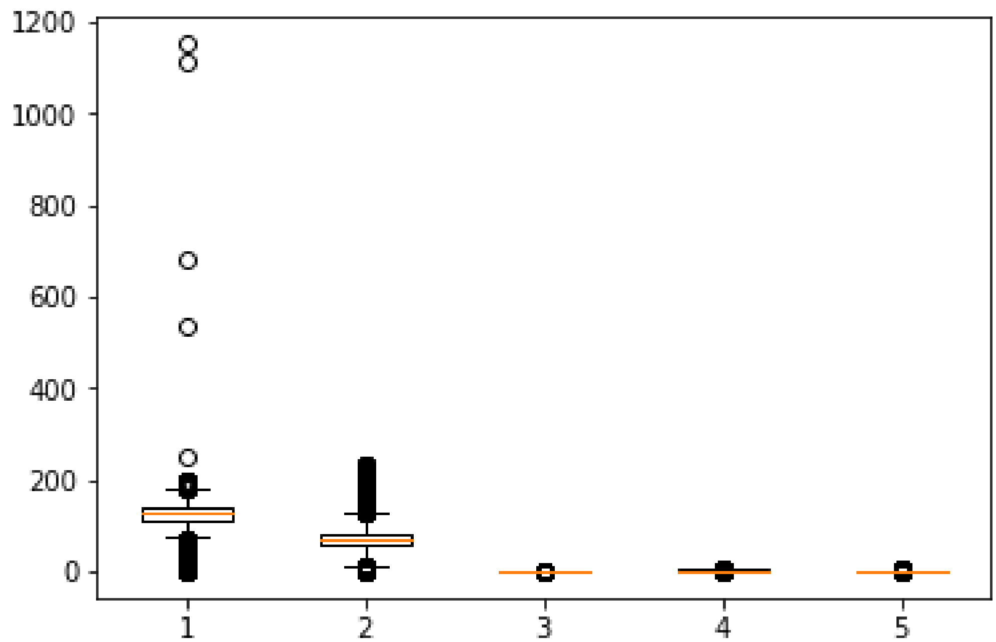
[48]:
```
<bound method NDFrame.head of                DT          GR    NPHI    RHOB  FACIES
0        50.2544  50.212800  0.5340  2.1228       1
1        50.3881  49.750900  0.5316  2.1250       1
2        49.8852  48.251300  0.5126  2.1316       1
3        49.9032  46.821200  0.5137  2.1437       1
4        50.0157  45.346300  0.5472  2.1611       1
…          …         …       …       …       …
58494  123.7404  80.913653  0.4993  2.4639       0
58495  123.8728  82.952576  0.5313  2.4660       0
58496  123.3722  84.044079  0.5448  2.4714       0
58497  122.6038  83.725389  0.5364  2.4750       0
58498  122.3045  83.329152  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 0x7f43f1570250>,
  <matplotlib.lines.Line2D at 0x7f43f15705e0>,
  <matplotlib.lines.Line2D at 0x7f43f157bbb0>,
  <matplotlib.lines.Line2D at 0x7f43f157bf40>,
  <matplotlib.lines.Line2D at 0x7f43f1594520>,
  <matplotlib.lines.Line2D at 0x7f43f15948b0>,
  <matplotlib.lines.Line2D at 0x7f43f159de50>,
  <matplotlib.lines.Line2D at 0x7f43f14e6220>,
  <matplotlib.lines.Line2D at 0x7f43f14f37c0>,
  <matplotlib.lines.Line2D at 0x7f43f14f3b50>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43f1570970>,
  <matplotlib.lines.Line2D at 0x7f43f1570d00>,
  <matplotlib.lines.Line2D at 0x7f43f1587310>,
  <matplotlib.lines.Line2D at 0x7f43f15876a0>,
  <matplotlib.lines.Line2D at 0x7f43f1594c40>,
  <matplotlib.lines.Line2D at 0x7f43f1594fd0>,
  <matplotlib.lines.Line2D at 0x7f43f14e65b0>,
  <matplotlib.lines.Line2D at 0x7f43f14e6940>,
  <matplotlib.lines.Line2D at 0x7f43f14f3ee0>,
  <matplotlib.lines.Line2D at 0x7f43f14fe2b0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43f1621e80>,
  <matplotlib.lines.Line2D at 0x7f43f157b820>,
```

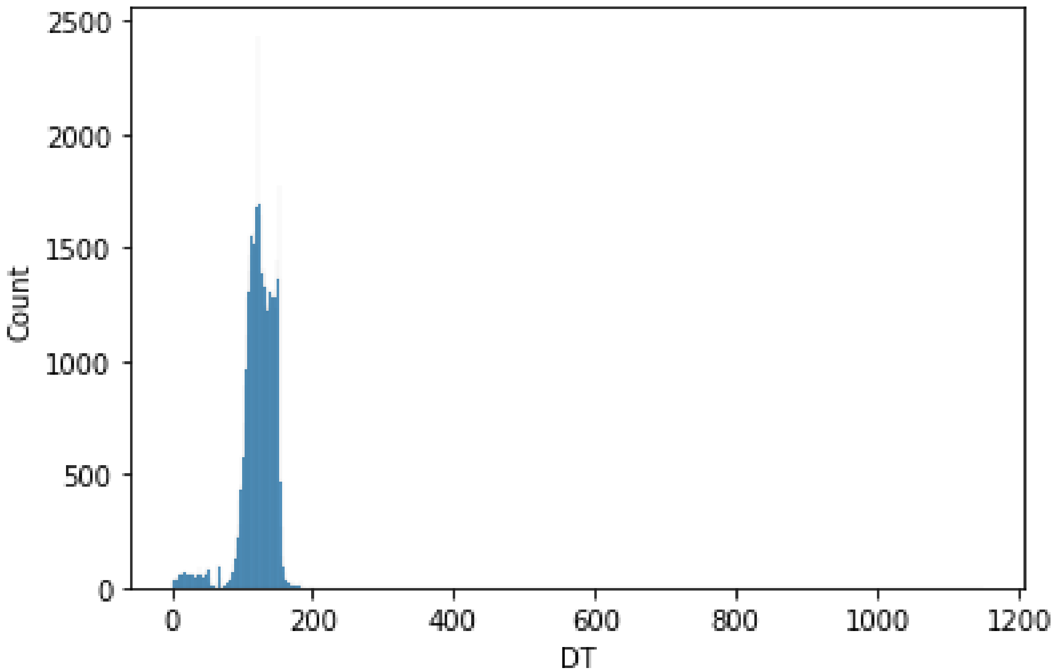
```
 <matplotlib.lines.Line2D at 0x7f43f1594190>,
 <matplotlib.lines.Line2D at 0x7f43f159dac0>,
 <matplotlib.lines.Line2D at 0x7f43f14f3430>],
'medians': [<matplotlib.lines.Line2D at 0x7f43f157b0d0>,
 <matplotlib.lines.Line2D at 0x7f43f1587a30>,
 <matplotlib.lines.Line2D at 0x7f43f159d3a0>,
 <matplotlib.lines.Line2D at 0x7f43f14e6cd0>,
 <matplotlib.lines.Line2D at 0x7f43f14fe640>],
'fliers': [<matplotlib.lines.Line2D at 0x7f43f157b460>,
 <matplotlib.lines.Line2D at 0x7f43f1587dc0>,
 <matplotlib.lines.Line2D at 0x7f43f159d730>,
 <matplotlib.lines.Line2D at 0x7f43f14f30a0>,
 <matplotlib.lines.Line2D at 0x7f43f14fea00>],
'means': []}
```



4.2 DT VISUALIZATION

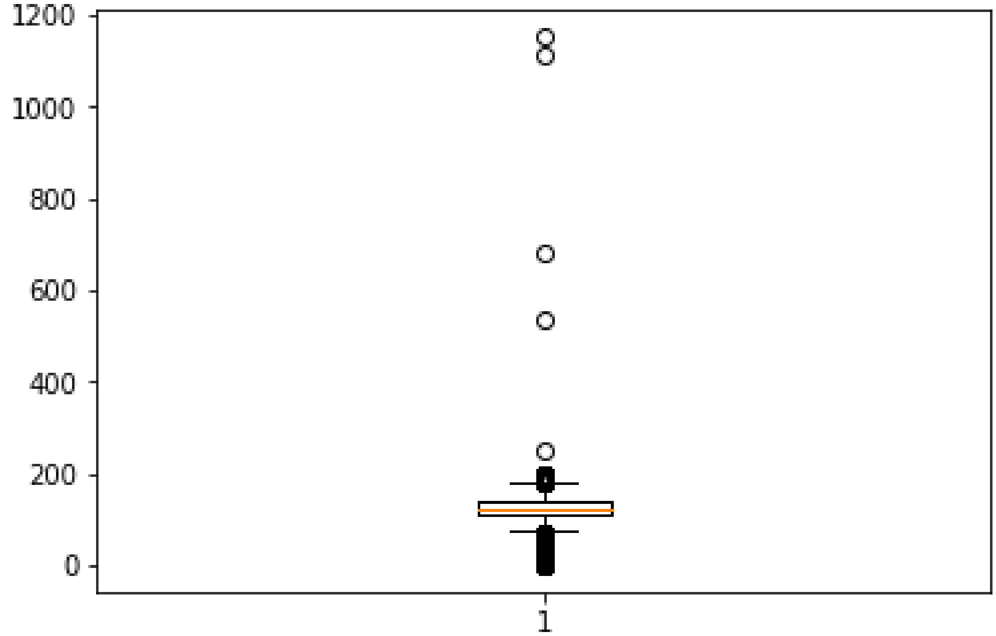
[50]:
```
sns.histplot(df.DT)
```

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



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

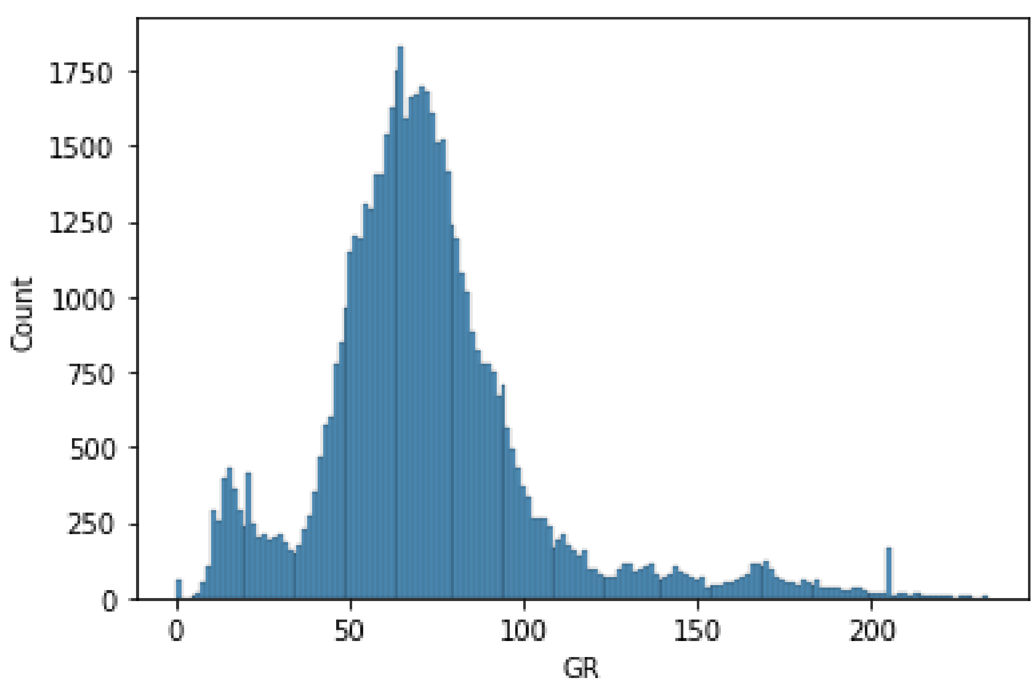
[51]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43efd0f1c0>,
  <matplotlib.lines.Line2D at 0x7f43efd0f550>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43efd0f8e0>,
  <matplotlib.lines.Line2D at 0x7f43efd0fc70>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43efd04df0>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43efd19040>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43efd193d0>],
 'means': []}
```



4.3 GR VISUALIZATION

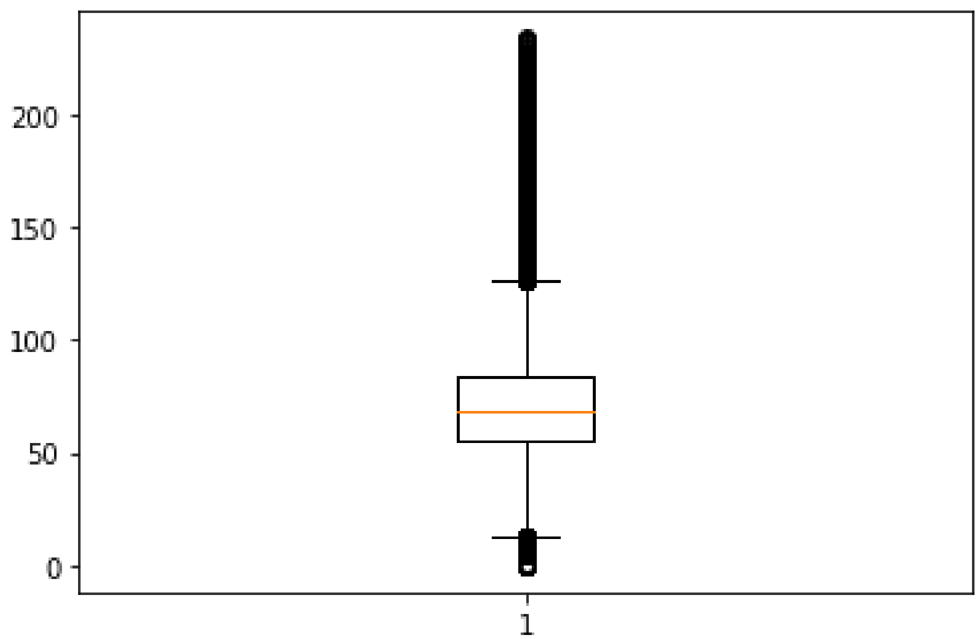
[52]:
```
sns.histplot(df.GR)
```

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



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

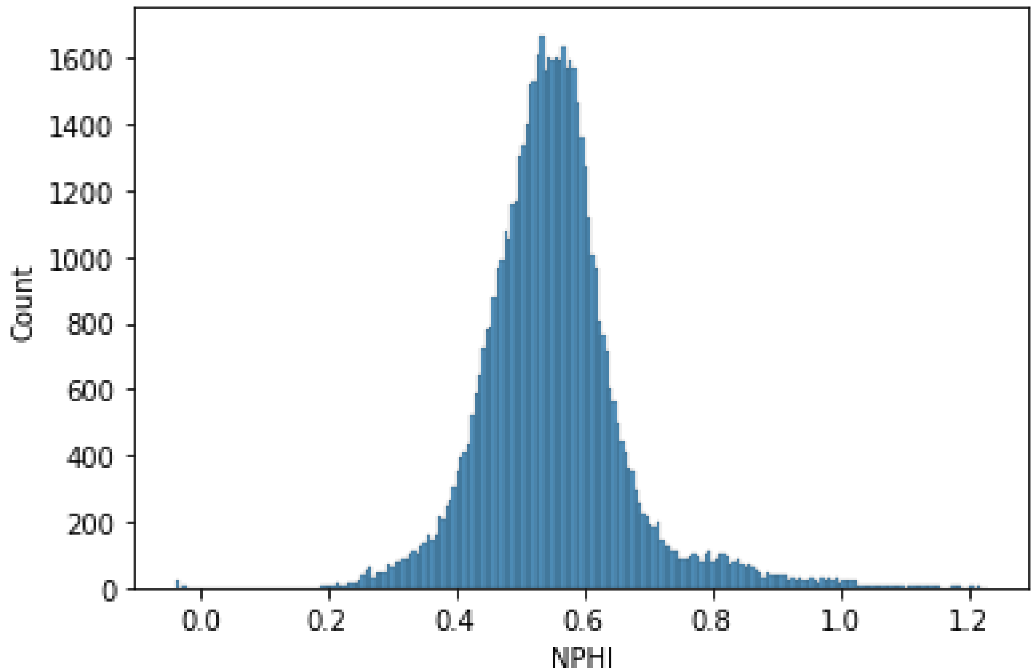
[53]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e3687550>,
  <matplotlib.lines.Line2D at 0x7f43e36878e0>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e3687c70>,
  <matplotlib.lines.Line2D at 0x7f43e3693040>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e36871c0>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43e36933d0>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e3693760>],
 'means': []}
```



4.4 NPHI VISUALIZATION

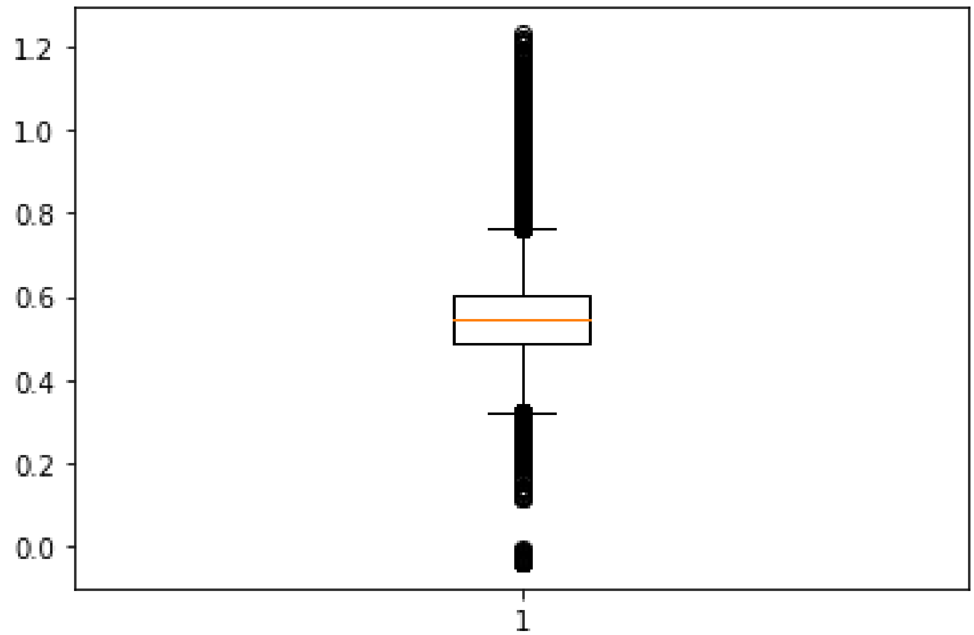
[54]:
```python
sns.histplot(df.NPHI)
```

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



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

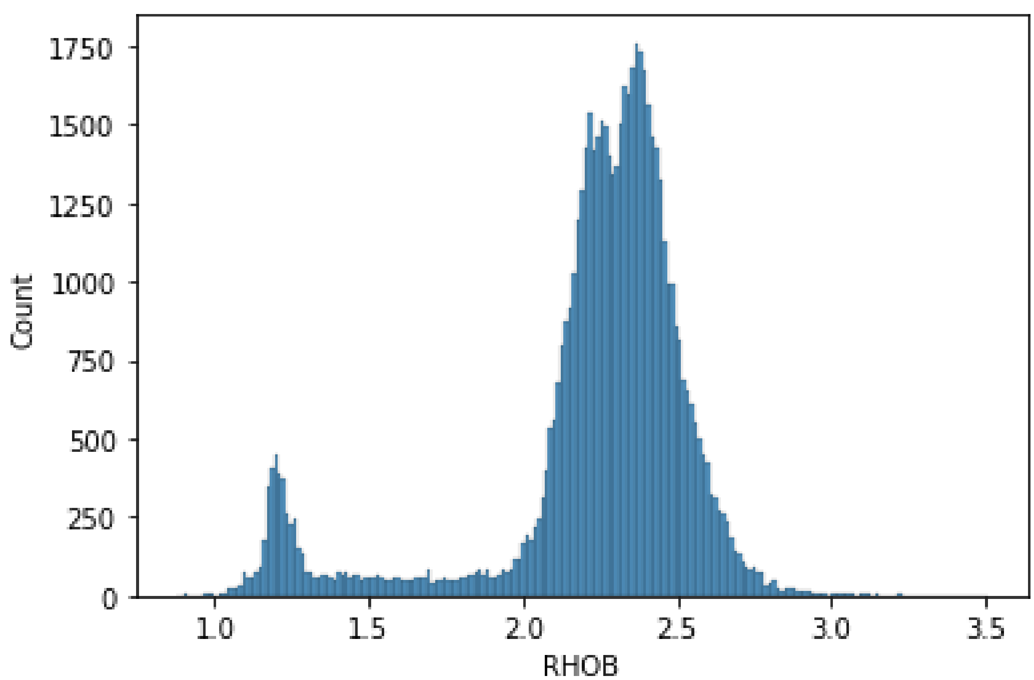
[55]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e336d310>,
  <matplotlib.lines.Line2D at 0x7f43e336d6a0>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e336da30>,
  <matplotlib.lines.Line2D at 0x7f43e336ddc0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e335ef40>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43e3376190>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e3376520>],
 'means': []}
```



4.5 RHOB VISUALIZATION

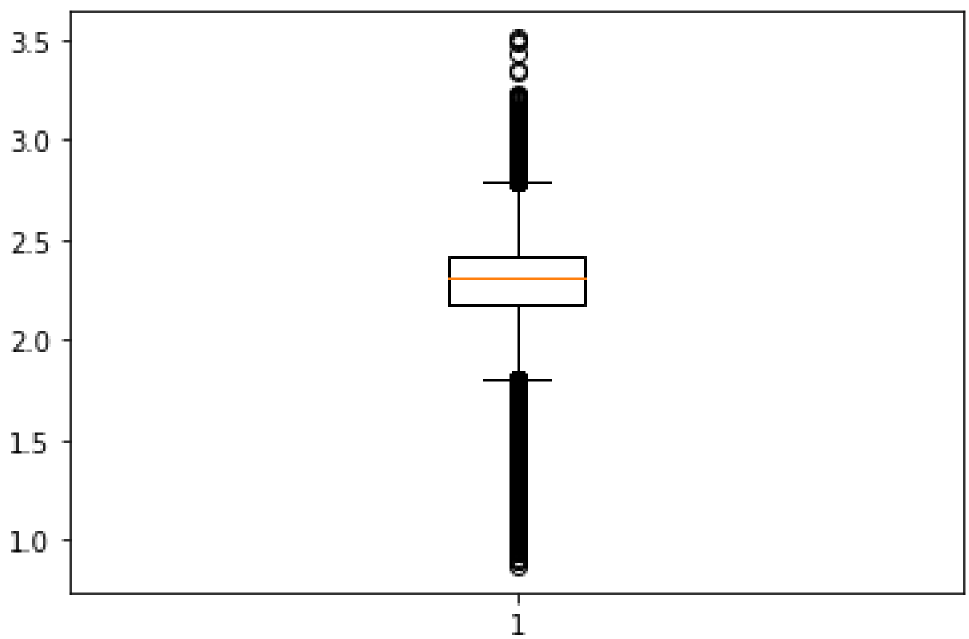
[56]:
```
sns.histplot(df.RHOB)
```

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



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

[57]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e30a83a0>,
  <matplotlib.lines.Line2D at 0x7f43e30a8730>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e30a8ac0>,
  <matplotlib.lines.Line2D at 0x7f43e30a8e50>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e309bfd0>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43e30b4220>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e30b45b0>],
 '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","NPHI","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       1
1      50.3881  49.7509  0.5316  2.1250       1
2      49.8852  48.2513  0.5126  2.1316       1
3      49.9032  46.8212  0.5137  2.1437       1
4      50.0157  45.3463  0.5472  2.1611       1
…          …        …       …       …       …
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  80.913653  0.4993  2.4639       0
58495  123.8728  82.952576  0.5313  2.4660       0
58496  123.3722  84.044079  0.5448  2.4714       0
58497  122.6038  83.725389  0.5364  2.4750       0
58498  122.3045  83.329152  0.5331  2.4709       0

[55907 rows x 5 columns]
column GR
Percentiles: 25th=55.447, 75th=84.350, IQR=28.902
InterQuartile Range Outliers-:
             DT        GR    NPHI    RHOB  FACIES
4029   151.3950   11.6218  0.8730  1.1941       3
4030   151.2614   11.7061  0.8996  1.2056       3
4039   152.8249   11.7563  0.7718  1.1963       3
4040   152.8680   11.5903  0.7690  1.1947       3
4041   152.9320   12.0709  0.7689  1.1923       3
…           …         …       …       …       …
57812  116.8102  136.5899  0.5287  2.4344       0
58179  110.8288  128.6649  0.5213  2.3846       0
58180  110.9551  130.6794  0.5160  2.3705       0
58181  114.0812  131.8473  0.4959  2.3630       0
58182  115.8771  127.8300  0.4907  2.3684       0

[4132 rows x 5 columns]
```

```
(4132, 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  80.913653  0.4993  2.4639       0
58495  123.8728  82.952576  0.5313  2.4660       0
58496  123.3722  84.044079  0.5448  2.4714       0
58497  122.6038  83.725389  0.5364  2.4750       0
58498  122.3045  83.329152  0.5331  2.4709       0

[51775 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

[2885 rows x 5 columns]
(2885, 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  80.913653  0.4993  2.4639       0
58495  123.8728  82.952576  0.5313  2.4660       0
58496  123.3722  84.044079  0.5448  2.4714       0
58497  122.6038  83.725389  0.5364  2.4750       0
58498  122.3045  83.329152  0.5331  2.4709       0

[48890 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  93.119065  0.4365  2.7816       0
58478   92.1297  99.825730  0.4509  2.8974       0
58479   92.0023  94.744734  0.4585  2.7846       0

[3498 rows x 5 columns]
(3498, 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
…           …          …       …       …     …
58494  123.7404  80.913653  0.4993  2.4639       0
58495  123.8728  82.952576  0.5313  2.4660       0
58496  123.3722  84.044079  0.5448  2.4714       0
58497  122.6038  83.725389  0.5364  2.4750       0
58498  122.3045  83.329152  0.5331  2.4709       0

[45392 rows x 5 columns]
```

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

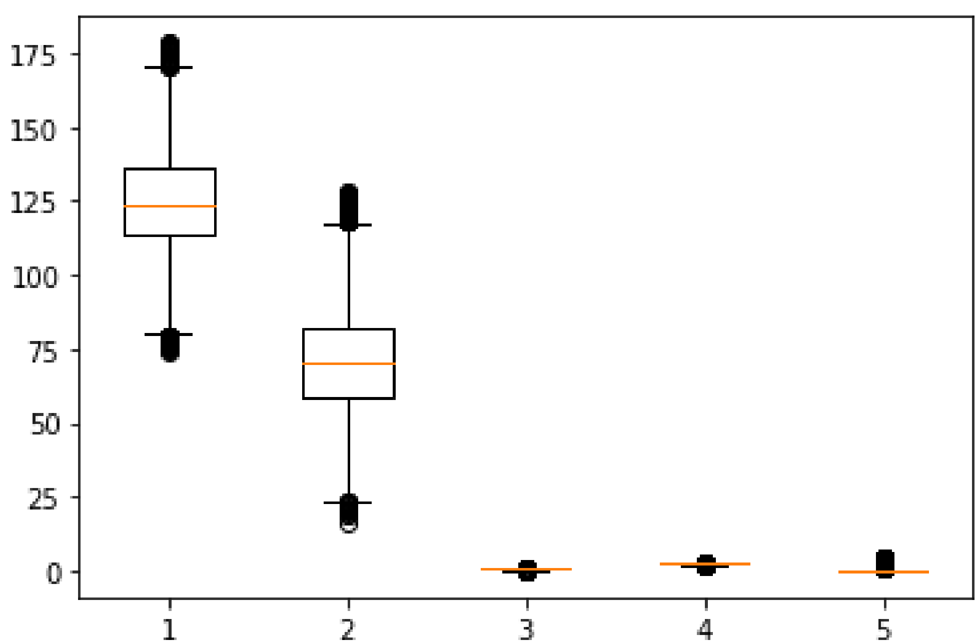
[60]: (45392, 5)

4.6 WHOLE DATA AFTER REMOVING OUTLIERS

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

[61]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e308fc70>,
  <matplotlib.lines.Line2D at 0x7f43e2c5e040>,
  <matplotlib.lines.Line2D at 0x7f43e2c69610>,
  <matplotlib.lines.Line2D at 0x7f43e2c699a0>,
  <matplotlib.lines.Line2D at 0x7f43e2c75f40>,
  <matplotlib.lines.Line2D at 0x7f43e2c80310>,
  <matplotlib.lines.Line2D at 0x7f43e2c8c8b0>,
  <matplotlib.lines.Line2D at 0x7f43e2c8cc40>,
```

```
 <matplotlib.lines.Line2D at 0x7f43e3022220>,
 <matplotlib.lines.Line2D at 0x7f43e30225b0>],
'caps': [<matplotlib.lines.Line2D at 0x7f43e2c5e400>,
 <matplotlib.lines.Line2D at 0x7f43e2c5e790>,
 <matplotlib.lines.Line2D at 0x7f43e2c69d30>,
 <matplotlib.lines.Line2D at 0x7f43e2c75100>,
 <matplotlib.lines.Line2D at 0x7f43e2c806a0>,
 <matplotlib.lines.Line2D at 0x7f43e2c80a30>,
 <matplotlib.lines.Line2D at 0x7f43e2c8cfd0>,
 <matplotlib.lines.Line2D at 0x7f43e2c953a0>,
 <matplotlib.lines.Line2D at 0x7f43e3022940>,
 <matplotlib.lines.Line2D at 0x7f43e3022cd0>],
'boxes': [<matplotlib.lines.Line2D at 0x7f43e308f8e0>,
 <matplotlib.lines.Line2D at 0x7f43e2c69280>,
 <matplotlib.lines.Line2D at 0x7f43e2c75bb0>,
 <matplotlib.lines.Line2D at 0x7f43e2c8c520>,
 <matplotlib.lines.Line2D at 0x7f43e2c95e50>],
'medians': [<matplotlib.lines.Line2D at 0x7f43e2c5eb20>,
 <matplotlib.lines.Line2D at 0x7f43e2c75490>,
 <matplotlib.lines.Line2D at 0x7f43e2c80dc0>,
 <matplotlib.lines.Line2D at 0x7f43e2c95730>,
 <matplotlib.lines.Line2D at 0x7f43e302c0a0>],
'fliers': [<matplotlib.lines.Line2D at 0x7f43e2c5eeb0>,
 <matplotlib.lines.Line2D at 0x7f43e2c75820>,
 <matplotlib.lines.Line2D at 0x7f43e2c8c190>,
 <matplotlib.lines.Line2D at 0x7f43e2c95ac0>,
 <matplotlib.lines.Line2D at 0x7f43e302c430>],
'means': []}
```



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

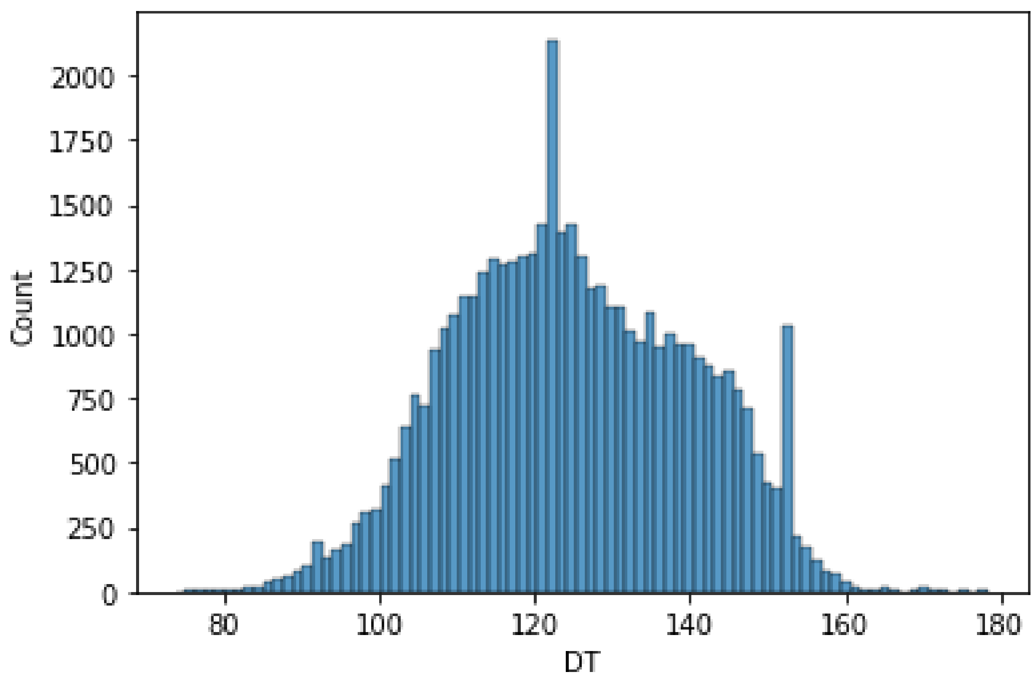
[62]:

	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)
```

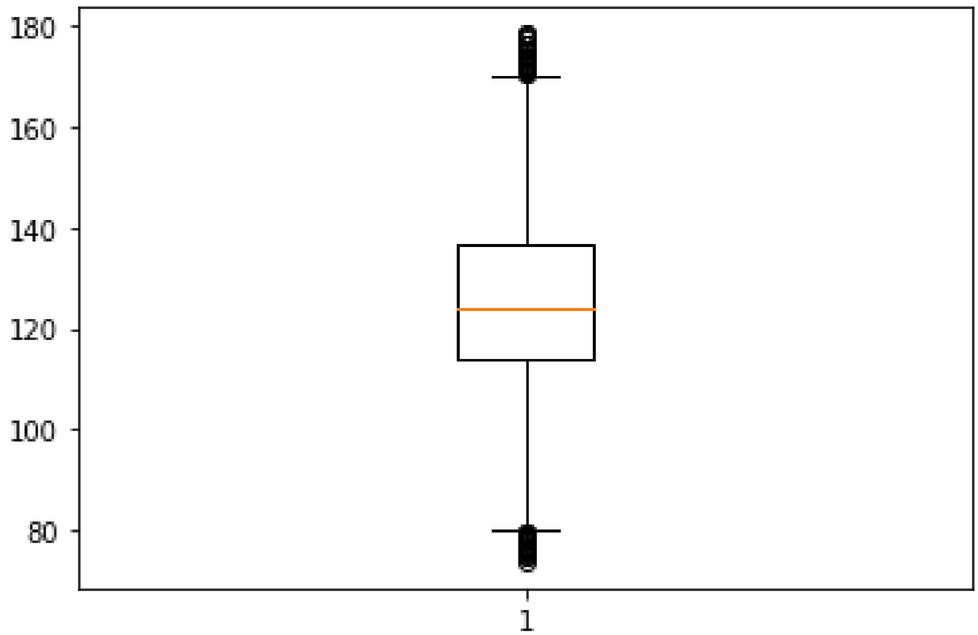
[63]: <AxesSubplot:xlabel='DT', ylabel='Count'>



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

[64]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e2f269d0>,
  <matplotlib.lines.Line2D at 0x7f43e2f26d60>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e2f33130>,
  <matplotlib.lines.Line2D at 0x7f43e2f334c0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e2f26640>],
```

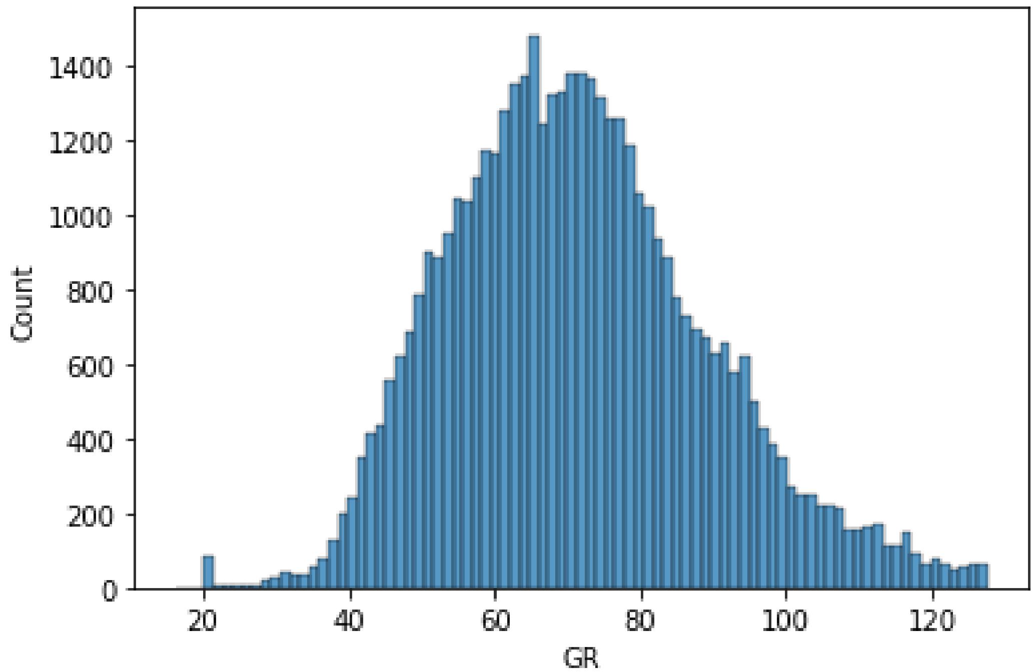
```
 'medians': [<matplotlib.lines.Line2D at 0x7f43e2f33850>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e2f33be0>],
 'means': []}
```



4.8 GR AFTER REMOVING OUTLIER

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

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

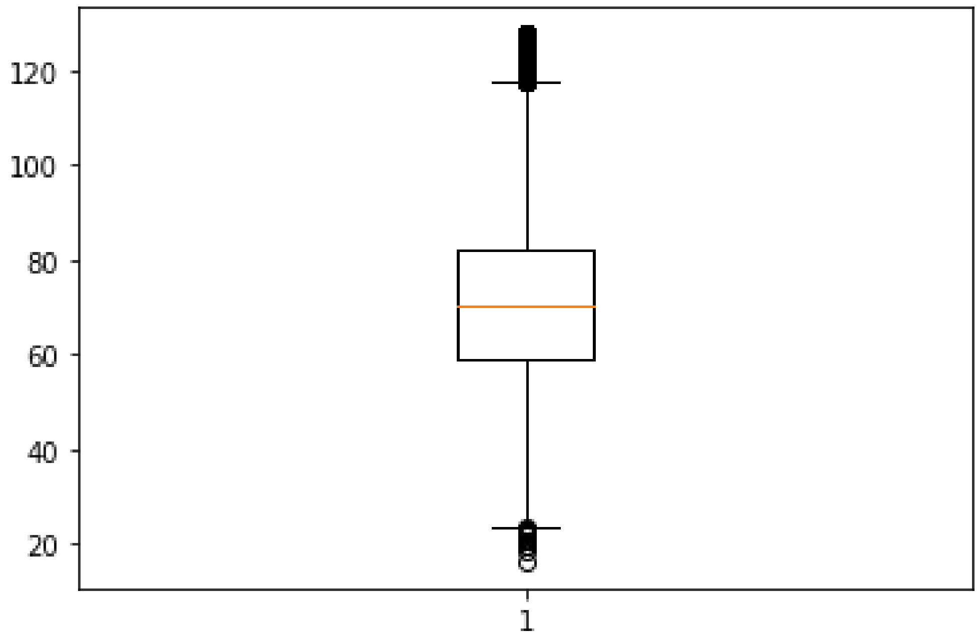


[66]:

```
plt.boxplot(df.GR)
```

[66]:

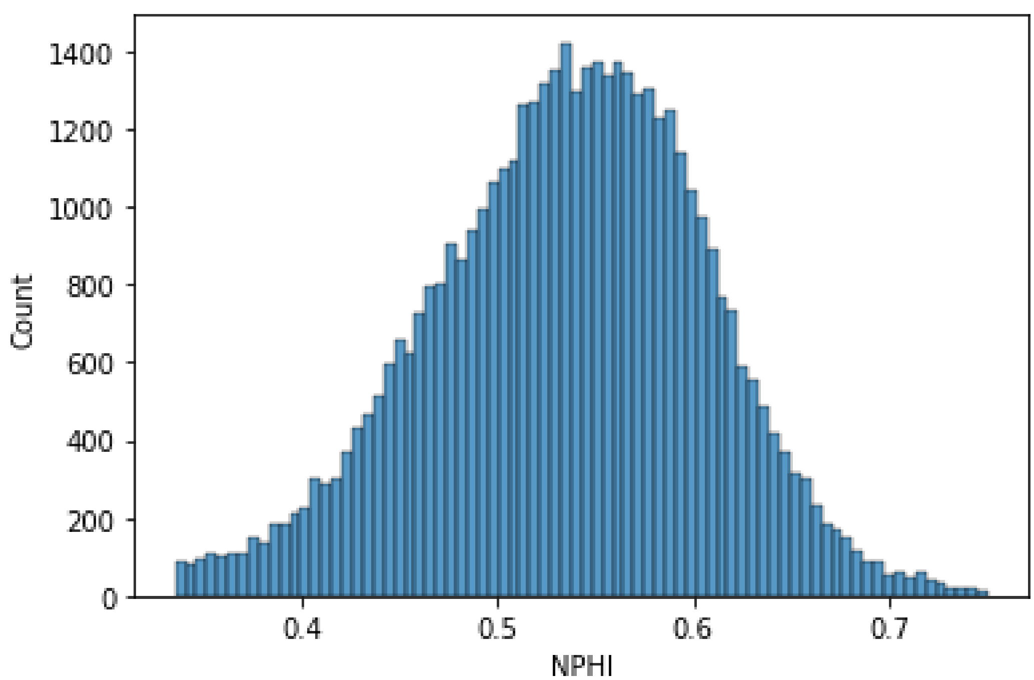
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e2d6b400>,
  <matplotlib.lines.Line2D at 0x7f43e2d6b790>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e2d6bb20>,
  <matplotlib.lines.Line2D at 0x7f43e2d6beb0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e2d6b070>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43e2d76280>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e2d76610>],
 'means': []}
```



4.9 NPHI AFTER REMOVING OUTLIER

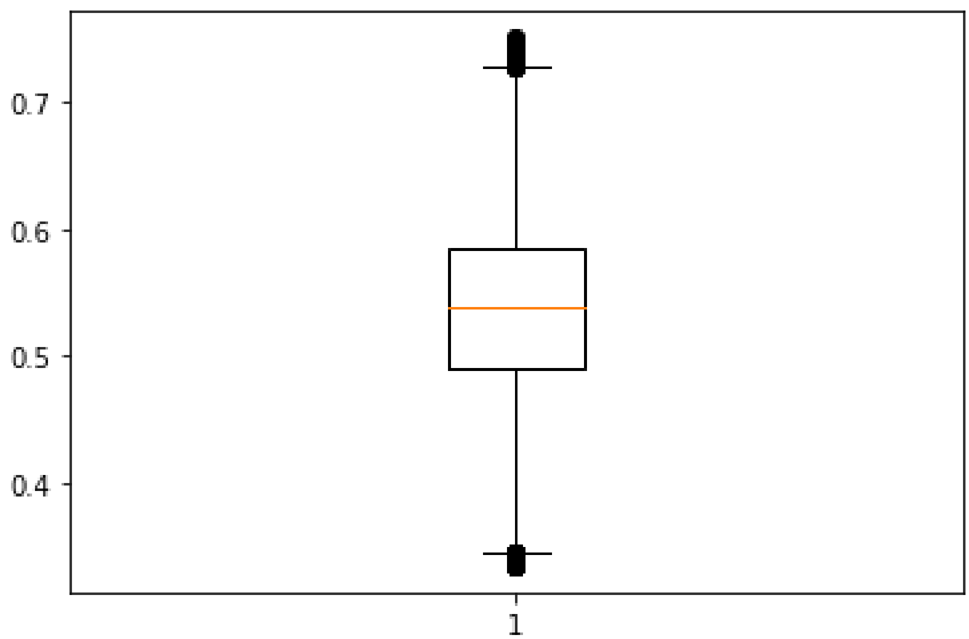
[67]:
```
sns.histplot(df.NPHI)
```

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



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

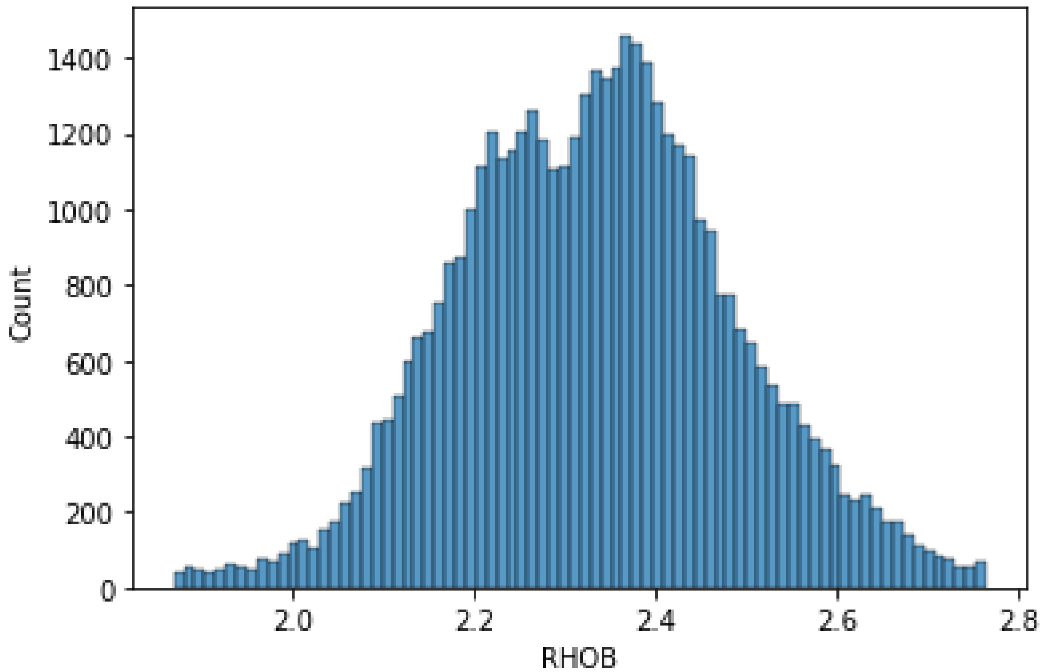
[68]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e2c3ffd0>,
  <matplotlib.lines.Line2D at 0x7f43e2c4d3a0>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e2c4d730>,
  <matplotlib.lines.Line2D at 0x7f43e2c4dac0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e2c3fc40>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43e2c4de50>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e2c57220>],
 'means': []}
```



4.10 RHOB AFTER REMOVING OUTLIER

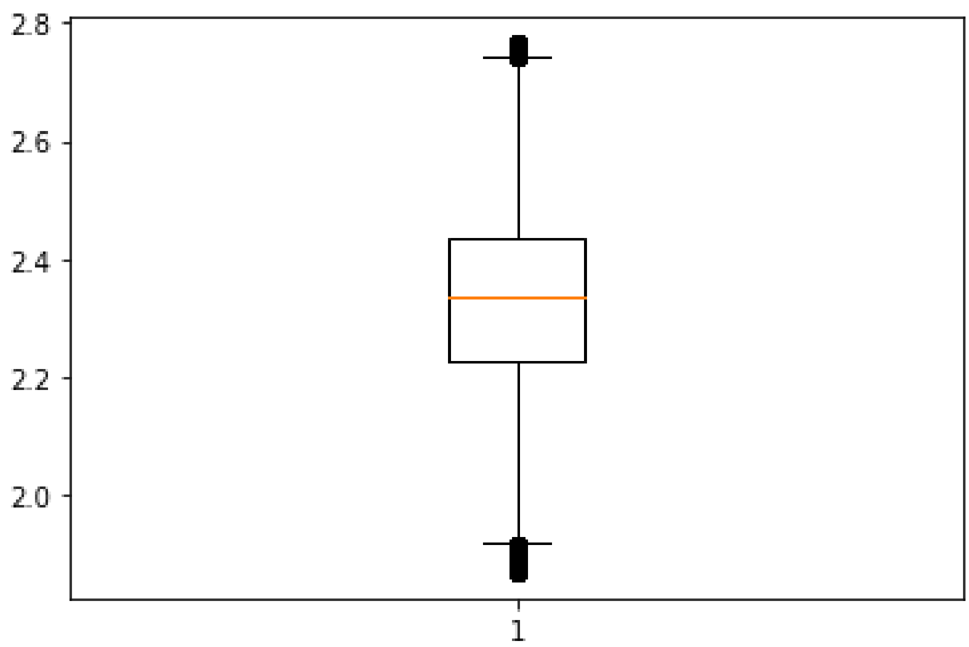
[69]:
```
sns.histplot(df.RHOB)
```

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



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

[70]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f43e2a6aa90>,
  <matplotlib.lines.Line2D at 0x7f43e2a6ae20>],
 'caps': [<matplotlib.lines.Line2D at 0x7f43e2a791f0>,
  <matplotlib.lines.Line2D at 0x7f43e2a79580>],
 'boxes': [<matplotlib.lines.Line2D at 0x7f43e2a6a700>],
 'medians': [<matplotlib.lines.Line2D at 0x7f43e2a79910>],
 'fliers': [<matplotlib.lines.Line2D at 0x7f43e2a79ca0>],
 '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
…	…	…	…	…	…
58494	123.7404	80.913653	0.4993	2.4639	0
58495	123.8728	82.952576	0.5313	2.4660	0
58496	123.3722	84.044079	0.5448	2.4714	0
58497	122.6038	83.725389	0.5364	2.4750	0
58498	122.3045	83.329152	0.5331	2.4709	0

[45392 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]:
```
(45392, 5)
```

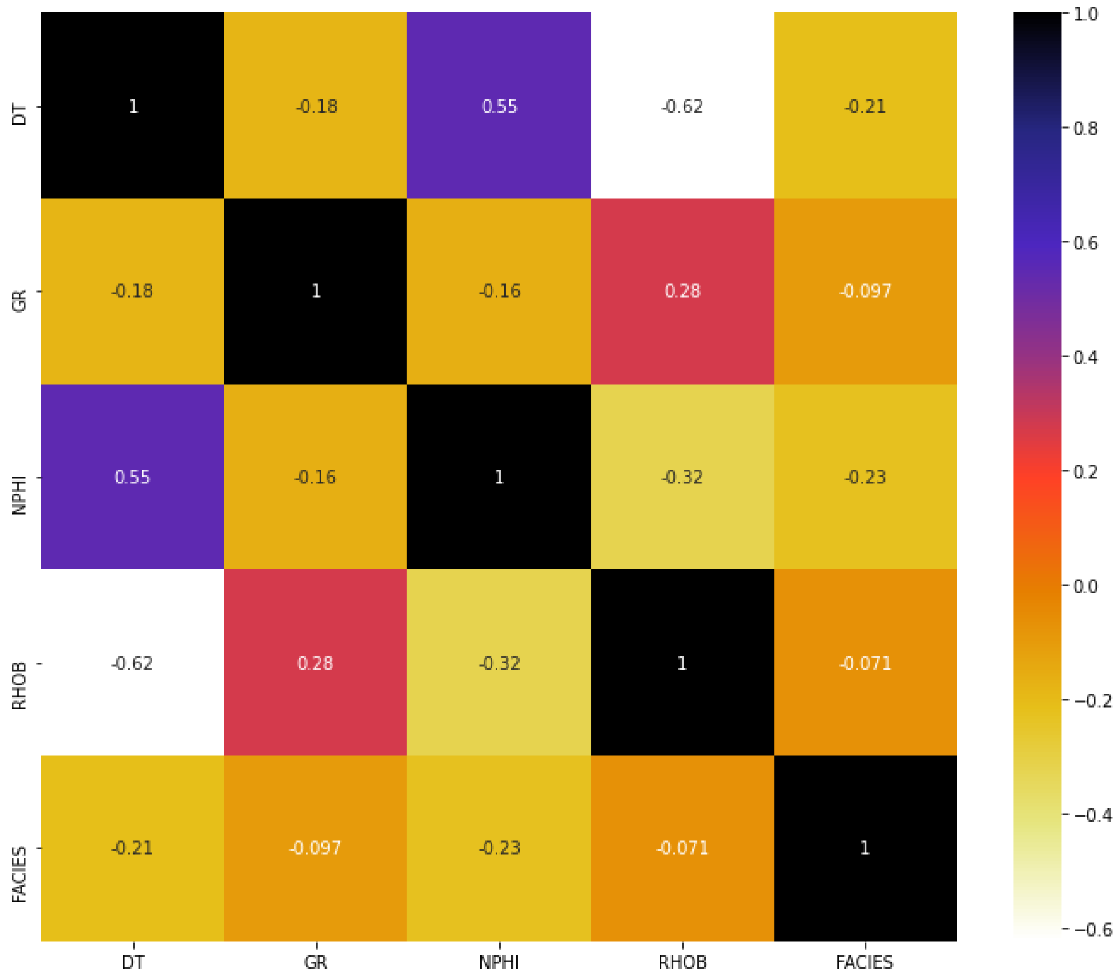
[74]:
```
features = df.shape[1]
features
```

[74]:
```
5
```

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

[75]:
```
DT        230.988291
GR        312.562035
NPHI        0.004939
RHOB        0.023286
FACIES      1.026135
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()
```

```python
        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
…          …          …       …       …
58494  123.7404  80.913653  0.4993  2.4639       0
58495  123.8728  82.952576  0.5313  2.4660       0
58496  123.3722  84.044079  0.5448  2.4714       0
58497  122.6038  83.725389  0.5364  2.4750       0
58498  122.3045  83.329152  0.5331  2.4709       0

[45392 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   -2.123320 -0.960120 -0.908901  0.465184       0
219   -2.107499 -0.987602 -0.921466  0.513056       0
2250   0.615845 -0.379535  0.260733 -0.722921       0
2251   0.694561 -0.351170  0.227225 -0.768859       0
2252   0.713622 -0.286443  0.247120 -0.679884       0
…           …         …         …         …     …
58494 -0.005948  0.452634 -0.419895  0.622340       0
58495 -0.000095  0.539265 -0.084817  0.632495       0
58496 -0.022224  0.585641  0.056545  0.658607       0
58497 -0.056191  0.572100 -0.031414  0.676015       0
58498 -0.069421  0.555265 -0.065969  0.656190       0

[45392 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 -2.123320 -0.960120 -0.908901  0.465184       0
1 -2.107499 -0.987602 -0.921466  0.513056       0
2  0.615845 -0.379535  0.260733 -0.722921       0
3  0.694561 -0.351170  0.227225 -0.768859       0
4  0.713622 -0.286443  0.247120 -0.679884       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]: (31774, 4)

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

[90]: (13618, 4)

[91]:
```
X_test
```

[91]:
```
             DT        GR      NPHI      RHOB
13107  0.585512  0.618141  0.524607  1.169729
24761 -0.492187 -0.387412  0.144503  1.200193
44043  0.314996  0.928135  0.108901  0.179400
17707  0.409511  0.461954  0.432461  0.064797
39859  0.550480  0.761862 -0.366492 -0.692456
…           …         …         …         …
17881  0.034243 -0.048364  0.727749  0.693907
43199  0.641064  0.074669  0.531937 -0.492263
1059  -0.678947 -0.525402 -1.783246 -0.685203
9662  -0.699114 -0.065674 -0.014660 -0.001934
6669  -0.587117  1.145858  1.061780  0.933752

[13618 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: 0.902321 using {'C': 0.1, 'penalty': 'l2', 'solver': 'newton-cg'}
0.902216 (0.002300) with: {'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'}
0.902205 (0.002308) with: {'C': 100, 'penalty': 'l2', 'solver': 'lbfgs'}
0.900401 (0.001923) with: {'C': 100, 'penalty': 'l2', 'solver': 'liblinear'}
0.902226 (0.002296) with: {'C': 10, 'penalty': 'l2', 'solver': 'newton-cg'}
0.902216 (0.002304) with: {'C': 10, 'penalty': 'l2', 'solver': 'lbfgs'}
0.900390 (0.001977) with: {'C': 10, 'penalty': 'l2', 'solver': 'liblinear'}
0.902279 (0.002317) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'newton-cg'}
0.902279 (0.002317) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'lbfgs'}
0.900275 (0.001884) with: {'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}
0.902321 (0.002006) with: {'C': 0.1, 'penalty': 'l2', 'solver': 'newton-cg'}
0.902321 (0.002006) with: {'C': 0.1, 'penalty': 'l2', 'solver': 'lbfgs'}
0.899624 (0.001890) with: {'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
0.900716 (0.001635) with: {'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}
0.900716 (0.001635) with: {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
0.899541 (0.001054) 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 0.094391
0:      learn: 1.3265685        total: 53.9ms   remaining: 53.9s
1:      learn: 1.1441447        total: 60.2ms   remaining: 30s
2:      learn: 1.0098936        total: 66.1ms   remaining: 22s
3:      learn: 0.9070909        total: 72.2ms   remaining: 18s
4:      learn: 0.8233265        total: 78.8ms   remaining: 15.7s
5:      learn: 0.7555721        total: 84.8ms   remaining: 14s
6:      learn: 0.7003256        total: 90.8ms   remaining: 12.9s
7:      learn: 0.6518884        total: 97.1ms   remaining: 12s
8:      learn: 0.6107570        total: 103ms    remaining: 11.4s
9:      learn: 0.5748652        total: 109ms    remaining: 10.8s
10:     learn: 0.5440267        total: 115ms    remaining: 10.4s
11:     learn: 0.5172645        total: 121ms    remaining: 9.97s
12:     learn: 0.4945976        total: 127ms    remaining: 9.63s
13:     learn: 0.4744959        total: 133ms    remaining: 9.39s
14:     learn: 0.4563637        total: 139ms    remaining: 9.14s
15:     learn: 0.4395808        total: 146ms    remaining: 8.95s
16:     learn: 0.4252144        total: 152ms    remaining: 8.78s
17:     learn: 0.4125212        total: 158ms    remaining: 8.61s
18:     learn: 0.4014292        total: 164ms    remaining: 8.46s
19:     learn: 0.3912449        total: 170ms    remaining: 8.33s
20:     learn: 0.3819971        total: 176ms    remaining: 8.21s
21:     learn: 0.3739274        total: 182ms    remaining: 8.1s
22:     learn: 0.3671660        total: 189ms    remaining: 8.04s
23:     learn: 0.3607492        total: 195ms    remaining: 7.94s
24:     learn: 0.3546127        total: 202ms    remaining: 7.87s
25:     learn: 0.3493236        total: 208ms    remaining: 7.78s
26:     learn: 0.3445168        total: 214ms    remaining: 7.7s
27:     learn: 0.3396638        total: 221ms    remaining: 7.66s
28:     learn: 0.3348014        total: 227ms    remaining: 7.6s
29:     learn: 0.3314200        total: 233ms    remaining: 7.54s
30:     learn: 0.3277385        total: 240ms    remaining: 7.49s
31:     learn: 0.3249679        total: 246ms    remaining: 7.43s
32:     learn: 0.3221409        total: 252ms    remaining: 7.38s
33:     learn: 0.3194326        total: 258ms    remaining: 7.32s
34:     learn: 0.3163122        total: 264ms    remaining: 7.29s
35:     learn: 0.3139723        total: 271ms    remaining: 7.25s
36:     learn: 0.3118089        total: 277ms    remaining: 7.2s
37:     learn: 0.3099771        total: 283ms    remaining: 7.16s
```

```
38:	learn: 0.3081699	total: 289ms	remaining: 7.12s
39:	learn: 0.3065539	total: 295ms	remaining: 7.08s
40:	learn: 0.3053255	total: 301ms	remaining: 7.05s
41:	learn: 0.3039021	total: 307ms	remaining: 7.01s
42:	learn: 0.3025759	total: 313ms	remaining: 6.97s
43:	learn: 0.3014233	total: 320ms	remaining: 6.94s
44:	learn: 0.2997517	total: 326ms	remaining: 6.91s
45:	learn: 0.2985515	total: 332ms	remaining: 6.88s
46:	learn: 0.2976736	total: 338ms	remaining: 6.85s
47:	learn: 0.2962642	total: 344ms	remaining: 6.82s
48:	learn: 0.2952980	total: 350ms	remaining: 6.79s
49:	learn: 0.2944012	total: 356ms	remaining: 6.76s
50:	learn: 0.2937763	total: 362ms	remaining: 6.74s
51:	learn: 0.2930726	total: 369ms	remaining: 6.72s
52:	learn: 0.2922601	total: 375ms	remaining: 6.69s
53:	learn: 0.2913897	total: 381ms	remaining: 6.68s
54:	learn: 0.2908381	total: 388ms	remaining: 6.66s
55:	learn: 0.2901412	total: 394ms	remaining: 6.64s
56:	learn: 0.2894886	total: 401ms	remaining: 6.63s
57:	learn: 0.2887200	total: 407ms	remaining: 6.61s
58:	learn: 0.2879917	total: 414ms	remaining: 6.61s
59:	learn: 0.2873456	total: 421ms	remaining: 6.59s
60:	learn: 0.2868916	total: 427ms	remaining: 6.58s
61:	learn: 0.2862890	total: 434ms	remaining: 6.57s
62:	learn: 0.2857701	total: 441ms	remaining: 6.55s
63:	learn: 0.2851188	total: 447ms	remaining: 6.53s
64:	learn: 0.2845449	total: 453ms	remaining: 6.51s
65:	learn: 0.2841480	total: 459ms	remaining: 6.5s
66:	learn: 0.2836590	total: 465ms	remaining: 6.47s
67:	learn: 0.2824413	total: 471ms	remaining: 6.46s
68:	learn: 0.2820536	total: 477ms	remaining: 6.44s
69:	learn: 0.2815608	total: 484ms	remaining: 6.43s
70:	learn: 0.2811326	total: 490ms	remaining: 6.41s
71:	learn: 0.2803515	total: 497ms	remaining: 6.4s
72:	learn: 0.2798132	total: 503ms	remaining: 6.38s
73:	learn: 0.2793368	total: 509ms	remaining: 6.37s
74:	learn: 0.2787849	total: 515ms	remaining: 6.36s
75:	learn: 0.2783549	total: 521ms	remaining: 6.33s
76:	learn: 0.2778664	total: 527ms	remaining: 6.32s
77:	learn: 0.2774651	total: 533ms	remaining: 6.3s
78:	learn: 0.2771077	total: 539ms	remaining: 6.28s
79:	learn: 0.2767508	total: 545ms	remaining: 6.27s
80:	learn: 0.2764223	total: 551ms	remaining: 6.25s
81:	learn: 0.2760075	total: 558ms	remaining: 6.24s
82:	learn: 0.2754035	total: 564ms	remaining: 6.23s
83:	learn: 0.2751105	total: 570ms	remaining: 6.21s
84:	learn: 0.2748215	total: 577ms	remaining: 6.21s
85:	learn: 0.2745139	total: 582ms	remaining: 6.19s
```

```
86:     learn: 0.2740358        total: 589ms    remaining: 6.18s
87:     learn: 0.2735845        total: 595ms    remaining: 6.16s
88:     learn: 0.2732176        total: 601ms    remaining: 6.15s
89:     learn: 0.2728186        total: 608ms    remaining: 6.14s
90:     learn: 0.2724885        total: 614ms    remaining: 6.13s
91:     learn: 0.2718692        total: 621ms    remaining: 6.13s
92:     learn: 0.2716402        total: 628ms    remaining: 6.12s
93:     learn: 0.2713018        total: 634ms    remaining: 6.11s
94:     learn: 0.2710079        total: 640ms    remaining: 6.1s
95:     learn: 0.2707728        total: 647ms    remaining: 6.09s
96:     learn: 0.2705703        total: 653ms    remaining: 6.08s
97:     learn: 0.2701709        total: 660ms    remaining: 6.08s
98:     learn: 0.2698516        total: 667ms    remaining: 6.07s
99:     learn: 0.2695801        total: 673ms    remaining: 6.06s
100:    learn: 0.2692980        total: 679ms    remaining: 6.04s
101:    learn: 0.2689943        total: 686ms    remaining: 6.04s
102:    learn: 0.2686165        total: 692ms    remaining: 6.03s
103:    learn: 0.2683034        total: 699ms    remaining: 6.02s
104:    learn: 0.2679960        total: 706ms    remaining: 6.01s
105:    learn: 0.2676761        total: 712ms    remaining: 6s
106:    learn: 0.2673879        total: 718ms    remaining: 5.99s
107:    learn: 0.2672502        total: 724ms    remaining: 5.98s
108:    learn: 0.2669644        total: 731ms    remaining: 5.97s
109:    learn: 0.2665695        total: 738ms    remaining: 5.97s
110:    learn: 0.2660552        total: 744ms    remaining: 5.96s
111:    learn: 0.2659019        total: 750ms    remaining: 5.95s
112:    learn: 0.2656889        total: 756ms    remaining: 5.94s
113:    learn: 0.2655331        total: 763ms    remaining: 5.93s
114:    learn: 0.2652994        total: 769ms    remaining: 5.92s
115:    learn: 0.2650709        total: 776ms    remaining: 5.91s
116:    learn: 0.2648800        total: 782ms    remaining: 5.9s
117:    learn: 0.2646628        total: 788ms    remaining: 5.89s
118:    learn: 0.2644057        total: 794ms    remaining: 5.88s
119:    learn: 0.2642081        total: 801ms    remaining: 5.87s
120:    learn: 0.2637376        total: 807ms    remaining: 5.86s
121:    learn: 0.2635814        total: 813ms    remaining: 5.85s
122:    learn: 0.2633007        total: 819ms    remaining: 5.84s
123:    learn: 0.2630463        total: 825ms    remaining: 5.83s
124:    learn: 0.2625436        total: 831ms    remaining: 5.82s
125:    learn: 0.2623534        total: 837ms    remaining: 5.81s
126:    learn: 0.2620983        total: 843ms    remaining: 5.8s
127:    learn: 0.2619562        total: 850ms    remaining: 5.79s
128:    learn: 0.2616396        total: 856ms    remaining: 5.78s
129:    learn: 0.2614650        total: 862ms    remaining: 5.77s
130:    learn: 0.2613639        total: 867ms    remaining: 5.75s
131:    learn: 0.2611495        total: 873ms    remaining: 5.74s
132:    learn: 0.2609170        total: 880ms    remaining: 5.74s
133:    learn: 0.2606897        total: 886ms    remaining: 5.73s
```

```
134:    learn: 0.2602264        total: 893ms    remaining: 5.72s
135:    learn: 0.2600640        total: 899ms    remaining: 5.71s
136:    learn: 0.2597188        total: 905ms    remaining: 5.7s
137:    learn: 0.2595349        total: 911ms    remaining: 5.69s
138:    learn: 0.2592843        total: 917ms    remaining: 5.68s
139:    learn: 0.2590630        total: 923ms    remaining: 5.67s
140:    learn: 0.2588638        total: 930ms    remaining: 5.66s
141:    learn: 0.2585916        total: 936ms    remaining: 5.66s
142:    learn: 0.2581374        total: 943ms    remaining: 5.65s
143:    learn: 0.2579750        total: 949ms    remaining: 5.64s
144:    learn: 0.2577894        total: 956ms    remaining: 5.63s
145:    learn: 0.2575293        total: 962ms    remaining: 5.63s
146:    learn: 0.2573961        total: 968ms    remaining: 5.62s
147:    learn: 0.2571889        total: 975ms    remaining: 5.61s
148:    learn: 0.2567704        total: 981ms    remaining: 5.6s
149:    learn: 0.2566223        total: 987ms    remaining: 5.59s
150:    learn: 0.2563600        total: 994ms    remaining: 5.59s
151:    learn: 0.2561895        total: 1000ms   remaining: 5.58s
152:    learn: 0.2558380        total: 1s       remaining: 5.57s
153:    learn: 0.2555814        total: 1.01s    remaining: 5.56s
154:    learn: 0.2554083        total: 1.02s    remaining: 5.55s
155:    learn: 0.2552984        total: 1.02s    remaining: 5.54s
156:    learn: 0.2551474        total: 1.03s    remaining: 5.53s
157:    learn: 0.2549385        total: 1.04s    remaining: 5.52s
158:    learn: 0.2547784        total: 1.04s    remaining: 5.51s
159:    learn: 0.2544535        total: 1.05s    remaining: 5.51s
160:    learn: 0.2541922        total: 1.05s    remaining: 5.5s
161:    learn: 0.2539903        total: 1.06s    remaining: 5.49s
162:    learn: 0.2537715        total: 1.07s    remaining: 5.48s
163:    learn: 0.2535303        total: 1.07s    remaining: 5.47s
164:    learn: 0.2533616        total: 1.08s    remaining: 5.46s
165:    learn: 0.2532041        total: 1.08s    remaining: 5.46s
166:    learn: 0.2529416        total: 1.09s    remaining: 5.45s
167:    learn: 0.2525866        total: 1.1s     remaining: 5.44s
168:    learn: 0.2525007        total: 1.1s     remaining: 5.43s
169:    learn: 0.2523330        total: 1.11s    remaining: 5.42s
170:    learn: 0.2521521        total: 1.12s    remaining: 5.41s
171:    learn: 0.2519040        total: 1.12s    remaining: 5.4s
172:    learn: 0.2516926        total: 1.13s    remaining: 5.39s
173:    learn: 0.2514995        total: 1.13s    remaining: 5.39s
174:    learn: 0.2512817        total: 1.14s    remaining: 5.38s
175:    learn: 0.2511407        total: 1.15s    remaining: 5.37s
176:    learn: 0.2510292        total: 1.15s    remaining: 5.36s
177:    learn: 0.2509016        total: 1.16s    remaining: 5.36s
178:    learn: 0.2507685        total: 1.17s    remaining: 5.35s
179:    learn: 0.2505808        total: 1.17s    remaining: 5.34s
180:    learn: 0.2504170        total: 1.18s    remaining: 5.33s
181:    learn: 0.2502518        total: 1.18s    remaining: 5.33s
```

```
182:	learn: 0.2500816	total: 1.19s	remaining: 5.32s
183:	learn: 0.2498906	total: 1.2s	remaining: 5.31s
184:	learn: 0.2497381	total: 1.2s	remaining: 5.3s
185:	learn: 0.2494736	total: 1.21s	remaining: 5.29s
186:	learn: 0.2492397	total: 1.21s	remaining: 5.28s
187:	learn: 0.2490249	total: 1.22s	remaining: 5.27s
188:	learn: 0.2488902	total: 1.23s	remaining: 5.26s
189:	learn: 0.2487012	total: 1.23s	remaining: 5.26s
190:	learn: 0.2484632	total: 1.24s	remaining: 5.25s
191:	learn: 0.2483583	total: 1.24s	remaining: 5.24s
192:	learn: 0.2482601	total: 1.25s	remaining: 5.23s
193:	learn: 0.2481310	total: 1.26s	remaining: 5.22s
194:	learn: 0.2480393	total: 1.26s	remaining: 5.21s
195:	learn: 0.2478056	total: 1.27s	remaining: 5.2s
196:	learn: 0.2476306	total: 1.27s	remaining: 5.2s
197:	learn: 0.2474531	total: 1.28s	remaining: 5.19s
198:	learn: 0.2473607	total: 1.29s	remaining: 5.18s
199:	learn: 0.2472163	total: 1.29s	remaining: 5.17s
200:	learn: 0.2470767	total: 1.3s	remaining: 5.16s
201:	learn: 0.2469456	total: 1.3s	remaining: 5.16s
202:	learn: 0.2467708	total: 1.31s	remaining: 5.15s
203:	learn: 0.2465132	total: 1.32s	remaining: 5.14s
204:	learn: 0.2463698	total: 1.32s	remaining: 5.13s
205:	learn: 0.2462758	total: 1.33s	remaining: 5.13s
206:	learn: 0.2460621	total: 1.34s	remaining: 5.12s
207:	learn: 0.2459294	total: 1.34s	remaining: 5.11s
208:	learn: 0.2457998	total: 1.35s	remaining: 5.11s
209:	learn: 0.2457208	total: 1.35s	remaining: 5.1s
210:	learn: 0.2456269	total: 1.36s	remaining: 5.09s
211:	learn: 0.2455251	total: 1.37s	remaining: 5.08s
212:	learn: 0.2454213	total: 1.37s	remaining: 5.07s
213:	learn: 0.2452242	total: 1.38s	remaining: 5.06s
214:	learn: 0.2451108	total: 1.38s	remaining: 5.05s
215:	learn: 0.2450393	total: 1.39s	remaining: 5.05s
216:	learn: 0.2449533	total: 1.4s	remaining: 5.04s
217:	learn: 0.2448301	total: 1.4s	remaining: 5.03s
218:	learn: 0.2445710	total: 1.41s	remaining: 5.02s
219:	learn: 0.2444952	total: 1.41s	remaining: 5.01s
220:	learn: 0.2443948	total: 1.42s	remaining: 5.01s
221:	learn: 0.2443132	total: 1.43s	remaining: 5s
222:	learn: 0.2441354	total: 1.43s	remaining: 4.99s
223:	learn: 0.2438815	total: 1.44s	remaining: 4.98s
224:	learn: 0.2437099	total: 1.45s	remaining: 4.98s
225:	learn: 0.2436042	total: 1.45s	remaining: 4.97s
226:	learn: 0.2433821	total: 1.46s	remaining: 4.96s
227:	learn: 0.2431384	total: 1.46s	remaining: 4.96s
228:	learn: 0.2429660	total: 1.47s	remaining: 4.95s
229:	learn: 0.2427872	total: 1.48s	remaining: 4.94s
```

```
230:	learn: 0.2425999	total: 1.48s	remaining: 4.93s
231:	learn: 0.2425140	total: 1.49s	remaining: 4.92s
232:	learn: 0.2422695	total: 1.49s	remaining: 4.92s
233:	learn: 0.2421520	total: 1.5s	remaining: 4.91s
234:	learn: 0.2420232	total: 1.51s	remaining: 4.91s
235:	learn: 0.2418425	total: 1.51s	remaining: 4.9s
236:	learn: 0.2417145	total: 1.52s	remaining: 4.89s
237:	learn: 0.2415322	total: 1.52s	remaining: 4.88s
238:	learn: 0.2413079	total: 1.53s	remaining: 4.88s
239:	learn: 0.2410921	total: 1.54s	remaining: 4.87s
240:	learn: 0.2406247	total: 1.54s	remaining: 4.86s
241:	learn: 0.2405484	total: 1.55s	remaining: 4.87s
242:	learn: 0.2403969	total: 1.56s	remaining: 4.86s
243:	learn: 0.2402799	total: 1.57s	remaining: 4.85s
244:	learn: 0.2402060	total: 1.57s	remaining: 4.85s
245:	learn: 0.2401171	total: 1.58s	remaining: 4.84s
246:	learn: 0.2399426	total: 1.59s	remaining: 4.84s
247:	learn: 0.2397320	total: 1.59s	remaining: 4.83s
248:	learn: 0.2396716	total: 1.6s	remaining: 4.83s
249:	learn: 0.2395485	total: 1.6s	remaining: 4.82s
250:	learn: 0.2394654	total: 1.61s	remaining: 4.81s
251:	learn: 0.2393852	total: 1.62s	remaining: 4.8s
252:	learn: 0.2392479	total: 1.62s	remaining: 4.8s
253:	learn: 0.2391280	total: 1.63s	remaining: 4.79s
254:	learn: 0.2390591	total: 1.64s	remaining: 4.78s
255:	learn: 0.2388890	total: 1.64s	remaining: 4.78s
256:	learn: 0.2386931	total: 1.65s	remaining: 4.77s
257:	learn: 0.2385642	total: 1.66s	remaining: 4.76s
258:	learn: 0.2383739	total: 1.66s	remaining: 4.76s
259:	learn: 0.2381171	total: 1.67s	remaining: 4.75s
260:	learn: 0.2379443	total: 1.68s	remaining: 4.74s
261:	learn: 0.2378468	total: 1.68s	remaining: 4.74s
262:	learn: 0.2375667	total: 1.69s	remaining: 4.73s
263:	learn: 0.2373650	total: 1.69s	remaining: 4.72s
264:	learn: 0.2372930	total: 1.7s	remaining: 4.72s
265:	learn: 0.2371216	total: 1.71s	remaining: 4.71s
266:	learn: 0.2370675	total: 1.71s	remaining: 4.7s
267:	learn: 0.2368375	total: 1.72s	remaining: 4.7s
268:	learn: 0.2367015	total: 1.73s	remaining: 4.69s
269:	learn: 0.2364558	total: 1.73s	remaining: 4.68s
270:	learn: 0.2363646	total: 1.74s	remaining: 4.67s
271:	learn: 0.2363036	total: 1.74s	remaining: 4.67s
272:	learn: 0.2360773	total: 1.75s	remaining: 4.66s
273:	learn: 0.2359868	total: 1.76s	remaining: 4.65s
274:	learn: 0.2358526	total: 1.76s	remaining: 4.64s
275:	learn: 0.2357256	total: 1.77s	remaining: 4.64s
276:	learn: 0.2355491	total: 1.77s	remaining: 4.63s
277:	learn: 0.2354738	total: 1.78s	remaining: 4.62s
```

```
278:    learn: 0.2352435        total: 1.79s    remaining: 4.62s
279:    learn: 0.2351142        total: 1.79s    remaining: 4.61s
280:    learn: 0.2350460        total: 1.8s     remaining: 4.6s
281:    learn: 0.2349202        total: 1.8s     remaining: 4.59s
282:    learn: 0.2348326        total: 1.81s    remaining: 4.59s
283:    learn: 0.2346826        total: 1.82s    remaining: 4.58s
284:    learn: 0.2346046        total: 1.82s    remaining: 4.57s
285:    learn: 0.2345158        total: 1.83s    remaining: 4.57s
286:    learn: 0.2344149        total: 1.83s    remaining: 4.56s
287:    learn: 0.2343082        total: 1.84s    remaining: 4.55s
288:    learn: 0.2341964        total: 1.85s    remaining: 4.54s
289:    learn: 0.2340399        total: 1.85s    remaining: 4.54s
290:    learn: 0.2339014        total: 1.86s    remaining: 4.53s
291:    learn: 0.2338514        total: 1.86s    remaining: 4.52s
292:    learn: 0.2337334        total: 1.87s    remaining: 4.52s
293:    learn: 0.2335168        total: 1.88s    remaining: 4.51s
294:    learn: 0.2334100        total: 1.88s    remaining: 4.5s
295:    learn: 0.2332535        total: 1.89s    remaining: 4.5s
296:    learn: 0.2331561        total: 1.9s     remaining: 4.49s
297:    learn: 0.2330767        total: 1.9s     remaining: 4.49s
298:    learn: 0.2329214        total: 1.91s    remaining: 4.48s
299:    learn: 0.2328660        total: 1.92s    remaining: 4.47s
300:    learn: 0.2327600        total: 1.92s    remaining: 4.47s
301:    learn: 0.2325710        total: 1.93s    remaining: 4.46s
302:    learn: 0.2324730        total: 1.94s    remaining: 4.46s
303:    learn: 0.2322987        total: 1.94s    remaining: 4.45s
304:    learn: 0.2322007        total: 1.95s    remaining: 4.45s
305:    learn: 0.2321026        total: 1.96s    remaining: 4.44s
306:    learn: 0.2319058        total: 1.96s    remaining: 4.43s
307:    learn: 0.2318392        total: 1.97s    remaining: 4.42s
308:    learn: 0.2317533        total: 1.98s    remaining: 4.42s
309:    learn: 0.2316300        total: 1.98s    remaining: 4.41s
310:    learn: 0.2315107        total: 1.99s    remaining: 4.4s
311:    learn: 0.2314088        total: 1.99s    remaining: 4.4s
312:    learn: 0.2312854        total: 2s       remaining: 4.39s
313:    learn: 0.2312056        total: 2.01s    remaining: 4.38s
314:    learn: 0.2310590        total: 2.01s    remaining: 4.38s
315:    learn: 0.2309510        total: 2.02s    remaining: 4.37s
316:    learn: 0.2308020        total: 2.02s    remaining: 4.36s
317:    learn: 0.2307294        total: 2.03s    remaining: 4.36s
318:    learn: 0.2306022        total: 2.04s    remaining: 4.35s
319:    learn: 0.2304457        total: 2.04s    remaining: 4.34s
320:    learn: 0.2303706        total: 2.05s    remaining: 4.33s
321:    learn: 0.2301776        total: 2.06s    remaining: 4.33s
322:    learn: 0.2300760        total: 2.06s    remaining: 4.32s
323:    learn: 0.2299269        total: 2.07s    remaining: 4.31s
324:    learn: 0.2298565        total: 2.07s    remaining: 4.31s
325:    learn: 0.2297747        total: 2.08s    remaining: 4.3s
```

```
326:	learn: 0.2296641	total: 2.09s	remaining: 4.29s
327:	learn: 0.2295321	total: 2.09s	remaining: 4.29s
328:	learn: 0.2293651	total: 2.1s	remaining: 4.28s
329:	learn: 0.2292387	total: 2.1s	remaining: 4.27s
330:	learn: 0.2291054	total: 2.11s	remaining: 4.27s
331:	learn: 0.2289381	total: 2.12s	remaining: 4.26s
332:	learn: 0.2288299	total: 2.12s	remaining: 4.25s
333:	learn: 0.2286605	total: 2.13s	remaining: 4.25s
334:	learn: 0.2285349	total: 2.14s	remaining: 4.24s
335:	learn: 0.2284535	total: 2.14s	remaining: 4.23s
336:	learn: 0.2283536	total: 2.15s	remaining: 4.23s
337:	learn: 0.2282592	total: 2.15s	remaining: 4.22s
338:	learn: 0.2279756	total: 2.16s	remaining: 4.21s
339:	learn: 0.2278969	total: 2.17s	remaining: 4.21s
340:	learn: 0.2278449	total: 2.17s	remaining: 4.2s
341:	learn: 0.2277673	total: 2.18s	remaining: 4.19s
342:	learn: 0.2276390	total: 2.18s	remaining: 4.18s
343:	learn: 0.2275845	total: 2.19s	remaining: 4.18s
344:	learn: 0.2274582	total: 2.2s	remaining: 4.17s
345:	learn: 0.2274200	total: 2.2s	remaining: 4.16s
346:	learn: 0.2273298	total: 2.21s	remaining: 4.16s
347:	learn: 0.2272161	total: 2.21s	remaining: 4.15s
348:	learn: 0.2270976	total: 2.22s	remaining: 4.14s
349:	learn: 0.2270232	total: 2.23s	remaining: 4.14s
350:	learn: 0.2268901	total: 2.23s	remaining: 4.13s
351:	learn: 0.2267488	total: 2.24s	remaining: 4.12s
352:	learn: 0.2266618	total: 2.25s	remaining: 4.11s
353:	learn: 0.2265505	total: 2.25s	remaining: 4.11s
354:	learn: 0.2264771	total: 2.26s	remaining: 4.1s
355:	learn: 0.2263983	total: 2.26s	remaining: 4.09s
356:	learn: 0.2263505	total: 2.27s	remaining: 4.09s
357:	learn: 0.2262671	total: 2.27s	remaining: 4.08s
358:	learn: 0.2261118	total: 2.28s	remaining: 4.07s
359:	learn: 0.2260468	total: 2.29s	remaining: 4.07s
360:	learn: 0.2259924	total: 2.29s	remaining: 4.06s
361:	learn: 0.2258496	total: 2.3s	remaining: 4.05s
362:	learn: 0.2257988	total: 2.31s	remaining: 4.05s
363:	learn: 0.2256857	total: 2.31s	remaining: 4.04s
364:	learn: 0.2256119	total: 2.32s	remaining: 4.03s
365:	learn: 0.2254031	total: 2.33s	remaining: 4.03s
366:	learn: 0.2251951	total: 2.33s	remaining: 4.02s
367:	learn: 0.2251213	total: 2.34s	remaining: 4.02s
368:	learn: 0.2250513	total: 2.35s	remaining: 4.01s
369:	learn: 0.2249376	total: 2.35s	remaining: 4s
370:	learn: 0.2248535	total: 2.36s	remaining: 4s
371:	learn: 0.2247426	total: 2.36s	remaining: 3.99s
372:	learn: 0.2246487	total: 2.37s	remaining: 3.98s
373:	learn: 0.2245474	total: 2.38s	remaining: 3.98s
```

```
374:    learn: 0.2243712        total: 2.38s    remaining: 3.97s
375:    learn: 0.2243207        total: 2.39s    remaining: 3.96s
376:    learn: 0.2242267        total: 2.39s    remaining: 3.96s
377:    learn: 0.2241480        total: 2.4s     remaining: 3.95s
378:    learn: 0.2240364        total: 2.41s    remaining: 3.94s
379:    learn: 0.2239902        total: 2.41s    remaining: 3.94s
380:    learn: 0.2239568        total: 2.42s    remaining: 3.93s
381:    learn: 0.2238940        total: 2.42s    remaining: 3.92s
382:    learn: 0.2238190        total: 2.43s    remaining: 3.91s
383:    learn: 0.2237454        total: 2.44s    remaining: 3.91s
384:    learn: 0.2236104        total: 2.44s    remaining: 3.9s
385:    learn: 0.2235448        total: 2.45s    remaining: 3.89s
386:    learn: 0.2234451        total: 2.45s    remaining: 3.89s
387:    learn: 0.2233694        total: 2.46s    remaining: 3.88s
388:    learn: 0.2231778        total: 2.46s    remaining: 3.87s
389:    learn: 0.2230817        total: 2.47s    remaining: 3.87s
390:    learn: 0.2229973        total: 2.48s    remaining: 3.86s
391:    learn: 0.2228756        total: 2.48s    remaining: 3.85s
392:    learn: 0.2227837        total: 2.49s    remaining: 3.85s
393:    learn: 0.2226789        total: 2.5s     remaining: 3.84s
394:    learn: 0.2225873        total: 2.5s     remaining: 3.84s
395:    learn: 0.2224891        total: 2.51s    remaining: 3.83s
396:    learn: 0.2223921        total: 2.52s    remaining: 3.83s
397:    learn: 0.2223288        total: 2.52s    remaining: 3.82s
398:    learn: 0.2221911        total: 2.53s    remaining: 3.81s
399:    learn: 0.2220904        total: 2.54s    remaining: 3.81s
400:    learn: 0.2220063        total: 2.55s    remaining: 3.8s
401:    learn: 0.2218881        total: 2.55s    remaining: 3.8s
402:    learn: 0.2218155        total: 2.56s    remaining: 3.79s
403:    learn: 0.2216986        total: 2.56s    remaining: 3.79s
404:    learn: 0.2216620        total: 2.57s    remaining: 3.78s
405:    learn: 0.2215218        total: 2.58s    remaining: 3.77s
406:    learn: 0.2214495        total: 2.58s    remaining: 3.76s
407:    learn: 0.2213840        total: 2.59s    remaining: 3.76s
408:    learn: 0.2212938        total: 2.6s     remaining: 3.75s
409:    learn: 0.2211536        total: 2.6s     remaining: 3.75s
410:    learn: 0.2210424        total: 2.61s    remaining: 3.74s
411:    learn: 0.2209620        total: 2.62s    remaining: 3.73s
412:    learn: 0.2208763        total: 2.62s    remaining: 3.73s
413:    learn: 0.2207857        total: 2.63s    remaining: 3.72s
414:    learn: 0.2207325        total: 2.63s    remaining: 3.71s
415:    learn: 0.2206513        total: 2.64s    remaining: 3.71s
416:    learn: 0.2205462        total: 2.65s    remaining: 3.7s
417:    learn: 0.2204792        total: 2.65s    remaining: 3.69s
418:    learn: 0.2204360        total: 2.66s    remaining: 3.69s
419:    learn: 0.2202566        total: 2.67s    remaining: 3.68s
420:    learn: 0.2201897        total: 2.67s    remaining: 3.67s
421:    learn: 0.2200053        total: 2.68s    remaining: 3.67s
```

```
422:	learn: 0.2199119	total: 2.68s	remaining: 3.66s
423:	learn: 0.2198538	total: 2.69s	remaining: 3.65s
424:	learn: 0.2196806	total: 2.7s	remaining: 3.65s
425:	learn: 0.2195446	total: 2.7s	remaining: 3.64s
426:	learn: 0.2194325	total: 2.71s	remaining: 3.64s
427:	learn: 0.2193998	total: 2.71s	remaining: 3.63s
428:	learn: 0.2192857	total: 2.72s	remaining: 3.62s
429:	learn: 0.2191101	total: 2.73s	remaining: 3.62s
430:	learn: 0.2189795	total: 2.73s	remaining: 3.61s
431:	learn: 0.2189000	total: 2.74s	remaining: 3.6s
432:	learn: 0.2187997	total: 2.75s	remaining: 3.6s
433:	learn: 0.2187450	total: 2.75s	remaining: 3.59s
434:	learn: 0.2186719	total: 2.76s	remaining: 3.58s
435:	learn: 0.2185494	total: 2.77s	remaining: 3.58s
436:	learn: 0.2183277	total: 2.77s	remaining: 3.57s
437:	learn: 0.2182696	total: 2.78s	remaining: 3.56s
438:	learn: 0.2180881	total: 2.78s	remaining: 3.56s
439:	learn: 0.2179389	total: 2.79s	remaining: 3.55s
440:	learn: 0.2178397	total: 2.8s	remaining: 3.54s
441:	learn: 0.2177451	total: 2.8s	remaining: 3.54s
442:	learn: 0.2175836	total: 2.81s	remaining: 3.53s
443:	learn: 0.2174961	total: 2.82s	remaining: 3.53s
444:	learn: 0.2174097	total: 2.82s	remaining: 3.52s
445:	learn: 0.2172442	total: 2.83s	remaining: 3.51s
446:	learn: 0.2171339	total: 2.83s	remaining: 3.51s
447:	learn: 0.2170263	total: 2.84s	remaining: 3.5s
448:	learn: 0.2168717	total: 2.85s	remaining: 3.49s
449:	learn: 0.2167817	total: 2.85s	remaining: 3.49s
450:	learn: 0.2166920	total: 2.86s	remaining: 3.48s
451:	learn: 0.2166008	total: 2.86s	remaining: 3.47s
452:	learn: 0.2165132	total: 2.87s	remaining: 3.47s
453:	learn: 0.2163766	total: 2.88s	remaining: 3.46s
454:	learn: 0.2161928	total: 2.88s	remaining: 3.46s
455:	learn: 0.2160550	total: 2.89s	remaining: 3.45s
456:	learn: 0.2159512	total: 2.9s	remaining: 3.44s
457:	learn: 0.2158766	total: 2.9s	remaining: 3.44s
458:	learn: 0.2157677	total: 2.91s	remaining: 3.43s
459:	learn: 0.2156778	total: 2.92s	remaining: 3.42s
460:	learn: 0.2156335	total: 2.92s	remaining: 3.42s
461:	learn: 0.2155545	total: 2.93s	remaining: 3.41s
462:	learn: 0.2154547	total: 2.94s	remaining: 3.4s
463:	learn: 0.2153413	total: 2.94s	remaining: 3.4s
464:	learn: 0.2152417	total: 2.95s	remaining: 3.39s
465:	learn: 0.2151660	total: 2.95s	remaining: 3.38s
466:	learn: 0.2150671	total: 2.96s	remaining: 3.38s
467:	learn: 0.2149759	total: 2.97s	remaining: 3.37s
468:	learn: 0.2148777	total: 2.97s	remaining: 3.37s
469:	learn: 0.2148264	total: 2.98s	remaining: 3.36s
```

```
470:    learn: 0.2147377        total: 2.98s    remaining: 3.35s
471:    learn: 0.2146307        total: 2.99s    remaining: 3.35s
472:    learn: 0.2145233        total: 3s       remaining: 3.34s
473:    learn: 0.2144755        total: 3s       remaining: 3.33s
474:    learn: 0.2143416        total: 3.01s    remaining: 3.33s
475:    learn: 0.2142608        total: 3.01s    remaining: 3.32s
476:    learn: 0.2141369        total: 3.02s    remaining: 3.31s
477:    learn: 0.2140400        total: 3.03s    remaining: 3.31s
478:    learn: 0.2139744        total: 3.03s    remaining: 3.3s
479:    learn: 0.2138870        total: 3.04s    remaining: 3.29s
480:    learn: 0.2137787        total: 3.04s    remaining: 3.29s
481:    learn: 0.2137266        total: 3.05s    remaining: 3.28s
482:    learn: 0.2136642        total: 3.06s    remaining: 3.27s
483:    learn: 0.2136163        total: 3.06s    remaining: 3.27s
484:    learn: 0.2134952        total: 3.07s    remaining: 3.26s
485:    learn: 0.2134634        total: 3.08s    remaining: 3.25s
486:    learn: 0.2134009        total: 3.08s    remaining: 3.25s
487:    learn: 0.2132641        total: 3.09s    remaining: 3.24s
488:    learn: 0.2132161        total: 3.1s     remaining: 3.23s
489:    learn: 0.2131495        total: 3.1s     remaining: 3.23s
490:    learn: 0.2130005        total: 3.11s    remaining: 3.22s
491:    learn: 0.2128442        total: 3.11s    remaining: 3.22s
492:    learn: 0.2127390        total: 3.12s    remaining: 3.21s
493:    learn: 0.2126422        total: 3.13s    remaining: 3.2s
494:    learn: 0.2125131        total: 3.13s    remaining: 3.2s
495:    learn: 0.2123999        total: 3.14s    remaining: 3.19s
496:    learn: 0.2123438        total: 3.15s    remaining: 3.18s
497:    learn: 0.2122668        total: 3.15s    remaining: 3.18s
498:    learn: 0.2121514        total: 3.16s    remaining: 3.17s
499:    learn: 0.2120527        total: 3.16s    remaining: 3.16s
500:    learn: 0.2119999        total: 3.17s    remaining: 3.16s
501:    learn: 0.2119371        total: 3.18s    remaining: 3.15s
502:    learn: 0.2118923        total: 3.18s    remaining: 3.14s
503:    learn: 0.2118240        total: 3.19s    remaining: 3.14s
504:    learn: 0.2117593        total: 3.19s    remaining: 3.13s
505:    learn: 0.2116512        total: 3.2s     remaining: 3.12s
506:    learn: 0.2115866        total: 3.21s    remaining: 3.12s
507:    learn: 0.2115161        total: 3.21s    remaining: 3.11s
508:    learn: 0.2113917        total: 3.22s    remaining: 3.1s
509:    learn: 0.2113214        total: 3.22s    remaining: 3.1s
510:    learn: 0.2112496        total: 3.23s    remaining: 3.09s
511:    learn: 0.2111018        total: 3.24s    remaining: 3.08s
512:    learn: 0.2109492        total: 3.24s    remaining: 3.08s
513:    learn: 0.2108762        total: 3.25s    remaining: 3.07s
514:    learn: 0.2108162        total: 3.25s    remaining: 3.07s
515:    learn: 0.2107745        total: 3.26s    remaining: 3.06s
516:    learn: 0.2107079        total: 3.27s    remaining: 3.05s
517:    learn: 0.2106499        total: 3.28s    remaining: 3.05s
```

```
518:	learn: 0.2106001	total: 3.28s	remaining: 3.04s
519:	learn: 0.2105139	total: 3.29s	remaining: 3.04s
520:	learn: 0.2104720	total: 3.3s	remaining: 3.03s
521:	learn: 0.2103790	total: 3.3s	remaining: 3.02s
522:	learn: 0.2103389	total: 3.31s	remaining: 3.02s
523:	learn: 0.2102202	total: 3.31s	remaining: 3.01s
524:	learn: 0.2100753	total: 3.32s	remaining: 3s
525:	learn: 0.2099908	total: 3.33s	remaining: 3s
526:	learn: 0.2098948	total: 3.33s	remaining: 2.99s
527:	learn: 0.2097867	total: 3.34s	remaining: 2.98s
528:	learn: 0.2097522	total: 3.34s	remaining: 2.98s
529:	learn: 0.2096978	total: 3.35s	remaining: 2.97s
530:	learn: 0.2096082	total: 3.36s	remaining: 2.96s
531:	learn: 0.2095653	total: 3.36s	remaining: 2.96s
532:	learn: 0.2094684	total: 3.37s	remaining: 2.95s
533:	learn: 0.2093117	total: 3.37s	remaining: 2.94s
534:	learn: 0.2092354	total: 3.38s	remaining: 2.94s
535:	learn: 0.2091436	total: 3.39s	remaining: 2.93s
536:	learn: 0.2090864	total: 3.39s	remaining: 2.92s
537:	learn: 0.2089512	total: 3.4s	remaining: 2.92s
538:	learn: 0.2088497	total: 3.4s	remaining: 2.91s
539:	learn: 0.2087798	total: 3.41s	remaining: 2.91s
540:	learn: 0.2086202	total: 3.42s	remaining: 2.9s
541:	learn: 0.2085208	total: 3.42s	remaining: 2.89s
542:	learn: 0.2084385	total: 3.43s	remaining: 2.89s
543:	learn: 0.2083634	total: 3.44s	remaining: 2.88s
544:	learn: 0.2082971	total: 3.44s	remaining: 2.88s
545:	learn: 0.2081810	total: 3.45s	remaining: 2.87s
546:	learn: 0.2079915	total: 3.46s	remaining: 2.87s
547:	learn: 0.2079285	total: 3.47s	remaining: 2.86s
548:	learn: 0.2078454	total: 3.47s	remaining: 2.85s
549:	learn: 0.2077833	total: 3.48s	remaining: 2.85s
550:	learn: 0.2077211	total: 3.49s	remaining: 2.84s
551:	learn: 0.2076747	total: 3.49s	remaining: 2.83s
552:	learn: 0.2076155	total: 3.5s	remaining: 2.83s
553:	learn: 0.2075145	total: 3.51s	remaining: 2.82s
554:	learn: 0.2074651	total: 3.51s	remaining: 2.82s
555:	learn: 0.2073812	total: 3.52s	remaining: 2.81s
556:	learn: 0.2072993	total: 3.52s	remaining: 2.8s
557:	learn: 0.2071930	total: 3.53s	remaining: 2.8s
558:	learn: 0.2071185	total: 3.54s	remaining: 2.79s
559:	learn: 0.2070510	total: 3.54s	remaining: 2.79s
560:	learn: 0.2068170	total: 3.55s	remaining: 2.78s
561:	learn: 0.2067595	total: 3.56s	remaining: 2.77s
562:	learn: 0.2066856	total: 3.56s	remaining: 2.77s
563:	learn: 0.2065821	total: 3.57s	remaining: 2.76s
564:	learn: 0.2065028	total: 3.58s	remaining: 2.75s
565:	learn: 0.2064569	total: 3.58s	remaining: 2.75s
```

```
566:    learn: 0.2063768        total: 3.59s    remaining: 2.74s
567:    learn: 0.2063010        total: 3.59s    remaining: 2.73s
568:    learn: 0.2061286        total: 3.6s     remaining: 2.73s
569:    learn: 0.2060233        total: 3.61s    remaining: 2.72s
570:    learn: 0.2059153        total: 3.61s    remaining: 2.71s
571:    learn: 0.2058480        total: 3.62s    remaining: 2.71s
572:    learn: 0.2056988        total: 3.63s    remaining: 2.7s
573:    learn: 0.2055973        total: 3.63s    remaining: 2.7s
574:    learn: 0.2055389        total: 3.64s    remaining: 2.69s
575:    learn: 0.2054888        total: 3.65s    remaining: 2.68s
576:    learn: 0.2054406        total: 3.65s    remaining: 2.68s
577:    learn: 0.2053376        total: 3.66s    remaining: 2.67s
578:    learn: 0.2053066        total: 3.67s    remaining: 2.67s
579:    learn: 0.2052102        total: 3.67s    remaining: 2.66s
580:    learn: 0.2051722        total: 3.68s    remaining: 2.65s
581:    learn: 0.2051036        total: 3.68s    remaining: 2.65s
582:    learn: 0.2050620        total: 3.69s    remaining: 2.64s
583:    learn: 0.2050266        total: 3.69s    remaining: 2.63s
584:    learn: 0.2049465        total: 3.7s     remaining: 2.63s
585:    learn: 0.2048190        total: 3.71s    remaining: 2.62s
586:    learn: 0.2047287        total: 3.71s    remaining: 2.61s
587:    learn: 0.2046807        total: 3.72s    remaining: 2.61s
588:    learn: 0.2045474        total: 3.73s    remaining: 2.6s
589:    learn: 0.2044338        total: 3.73s    remaining: 2.59s
590:    learn: 0.2043946        total: 3.74s    remaining: 2.59s
591:    learn: 0.2043238        total: 3.74s    remaining: 2.58s
592:    learn: 0.2042665        total: 3.75s    remaining: 2.57s
593:    learn: 0.2041527        total: 3.76s    remaining: 2.57s
594:    learn: 0.2040958        total: 3.76s    remaining: 2.56s
595:    learn: 0.2039872        total: 3.77s    remaining: 2.56s
596:    learn: 0.2038833        total: 3.77s    remaining: 2.55s
597:    learn: 0.2038104        total: 3.78s    remaining: 2.54s
598:    learn: 0.2037615        total: 3.79s    remaining: 2.54s
599:    learn: 0.2036613        total: 3.79s    remaining: 2.53s
600:    learn: 0.2035668        total: 3.8s     remaining: 2.52s
601:    learn: 0.2034867        total: 3.81s    remaining: 2.52s
602:    learn: 0.2034659        total: 3.81s    remaining: 2.51s
603:    learn: 0.2033558        total: 3.82s    remaining: 2.5s
604:    learn: 0.2032621        total: 3.82s    remaining: 2.5s
605:    learn: 0.2031786        total: 3.83s    remaining: 2.49s
606:    learn: 0.2030911        total: 3.84s    remaining: 2.48s
607:    learn: 0.2029989        total: 3.84s    remaining: 2.48s
608:    learn: 0.2029045        total: 3.85s    remaining: 2.47s
609:    learn: 0.2028255        total: 3.86s    remaining: 2.46s
610:    learn: 0.2027479        total: 3.86s    remaining: 2.46s
611:    learn: 0.2026813        total: 3.87s    remaining: 2.45s
612:    learn: 0.2026205        total: 3.87s    remaining: 2.45s
613:    learn: 0.2025210        total: 3.88s    remaining: 2.44s
```

```
614:    learn: 0.2024570        total: 3.89s    remaining: 2.43s
615:    learn: 0.2023572        total: 3.89s    remaining: 2.43s
616:    learn: 0.2022534        total: 3.9s     remaining: 2.42s
617:    learn: 0.2021801        total: 3.9s     remaining: 2.41s
618:    learn: 0.2021383        total: 3.91s    remaining: 2.41s
619:    learn: 0.2020917        total: 3.92s    remaining: 2.4s
620:    learn: 0.2019920        total: 3.92s    remaining: 2.39s
621:    learn: 0.2018590        total: 3.93s    remaining: 2.39s
622:    learn: 0.2017607        total: 3.94s    remaining: 2.38s
623:    learn: 0.2016948        total: 3.94s    remaining: 2.38s
624:    learn: 0.2015345        total: 3.95s    remaining: 2.37s
625:    learn: 0.2014788        total: 3.95s    remaining: 2.36s
626:    learn: 0.2014067        total: 3.96s    remaining: 2.36s
627:    learn: 0.2013413        total: 3.97s    remaining: 2.35s
628:    learn: 0.2012455        total: 3.97s    remaining: 2.34s
629:    learn: 0.2012018        total: 3.98s    remaining: 2.34s
630:    learn: 0.2011478        total: 3.98s    remaining: 2.33s
631:    learn: 0.2010356        total: 3.99s    remaining: 2.32s
632:    learn: 0.2009691        total: 4s       remaining: 2.32s
633:    learn: 0.2009022        total: 4s       remaining: 2.31s
634:    learn: 0.2008306        total: 4.01s    remaining: 2.3s
635:    learn: 0.2007926        total: 4.01s    remaining: 2.3s
636:    learn: 0.2007445        total: 4.02s    remaining: 2.29s
637:    learn: 0.2006610        total: 4.03s    remaining: 2.29s
638:    learn: 0.2005908        total: 4.03s    remaining: 2.28s
639:    learn: 0.2005211        total: 4.04s    remaining: 2.27s
640:    learn: 0.2004693        total: 4.05s    remaining: 2.27s
641:    learn: 0.2004018        total: 4.05s    remaining: 2.26s
642:    learn: 0.2003394        total: 4.06s    remaining: 2.25s
643:    learn: 0.2002736        total: 4.06s    remaining: 2.25s
644:    learn: 0.2002289        total: 4.07s    remaining: 2.24s
645:    learn: 0.2001750        total: 4.08s    remaining: 2.23s
646:    learn: 0.2001379        total: 4.08s    remaining: 2.23s
647:    learn: 0.2000654        total: 4.09s    remaining: 2.22s
648:    learn: 0.1999964        total: 4.09s    remaining: 2.21s
649:    learn: 0.1999407        total: 4.1s     remaining: 2.21s
650:    learn: 0.1998783        total: 4.11s    remaining: 2.2s
651:    learn: 0.1997949        total: 4.11s    remaining: 2.19s
652:    learn: 0.1997280        total: 4.12s    remaining: 2.19s
653:    learn: 0.1996323        total: 4.12s    remaining: 2.18s
654:    learn: 0.1995662        total: 4.13s    remaining: 2.17s
655:    learn: 0.1995313        total: 4.14s    remaining: 2.17s
656:    learn: 0.1994082        total: 4.14s    remaining: 2.16s
657:    learn: 0.1993488        total: 4.15s    remaining: 2.16s
658:    learn: 0.1992498        total: 4.16s    remaining: 2.15s
659:    learn: 0.1992097        total: 4.16s    remaining: 2.14s
660:    learn: 0.1991774        total: 4.17s    remaining: 2.14s
661:    learn: 0.1990894        total: 4.17s    remaining: 2.13s
```

```
662:    learn: 0.1990365        total: 4.18s    remaining: 2.13s
663:    learn: 0.1989641        total: 4.19s    remaining: 2.12s
664:    learn: 0.1989173        total: 4.19s    remaining: 2.11s
665:    learn: 0.1988548        total: 4.2s     remaining: 2.11s
666:    learn: 0.1988125        total: 4.21s    remaining: 2.1s
667:    learn: 0.1986869        total: 4.21s    remaining: 2.09s
668:    learn: 0.1986417        total: 4.22s    remaining: 2.09s
669:    learn: 0.1986112        total: 4.22s    remaining: 2.08s
670:    learn: 0.1985652        total: 4.23s    remaining: 2.07s
671:    learn: 0.1984814        total: 4.24s    remaining: 2.07s
672:    learn: 0.1984098        total: 4.24s    remaining: 2.06s
673:    learn: 0.1983605        total: 4.25s    remaining: 2.06s
674:    learn: 0.1982421        total: 4.25s    remaining: 2.05s
675:    learn: 0.1980866        total: 4.26s    remaining: 2.04s
676:    learn: 0.1980079        total: 4.27s    remaining: 2.04s
677:    learn: 0.1978641        total: 4.27s    remaining: 2.03s
678:    learn: 0.1977948        total: 4.28s    remaining: 2.02s
679:    learn: 0.1976980        total: 4.29s    remaining: 2.02s
680:    learn: 0.1976036        total: 4.29s    remaining: 2.01s
681:    learn: 0.1975447        total: 4.3s     remaining: 2s
682:    learn: 0.1974716        total: 4.31s    remaining: 2s
683:    learn: 0.1974157        total: 4.31s    remaining: 1.99s
684:    learn: 0.1973268        total: 4.32s    remaining: 1.99s
685:    learn: 0.1972638        total: 4.32s    remaining: 1.98s
686:    learn: 0.1972139        total: 4.33s    remaining: 1.97s
687:    learn: 0.1971788        total: 4.34s    remaining: 1.97s
688:    learn: 0.1971374        total: 4.34s    remaining: 1.96s
689:    learn: 0.1970828        total: 4.35s    remaining: 1.95s
690:    learn: 0.1970310        total: 4.35s    remaining: 1.95s
691:    learn: 0.1969840        total: 4.36s    remaining: 1.94s
692:    learn: 0.1969072        total: 4.37s    remaining: 1.94s
693:    learn: 0.1968492        total: 4.38s    remaining: 1.93s
694:    learn: 0.1966983        total: 4.38s    remaining: 1.92s
695:    learn: 0.1966088        total: 4.39s    remaining: 1.92s
696:    learn: 0.1965760        total: 4.39s    remaining: 1.91s
697:    learn: 0.1965173        total: 4.4s     remaining: 1.9s
698:    learn: 0.1964285        total: 4.41s    remaining: 1.9s
699:    learn: 0.1963442        total: 4.42s    remaining: 1.89s
700:    learn: 0.1961963        total: 4.42s    remaining: 1.89s
701:    learn: 0.1961272        total: 4.43s    remaining: 1.88s
702:    learn: 0.1960599        total: 4.44s    remaining: 1.87s
703:    learn: 0.1959964        total: 4.44s    remaining: 1.87s
704:    learn: 0.1959371        total: 4.45s    remaining: 1.86s
705:    learn: 0.1958750        total: 4.46s    remaining: 1.86s
706:    learn: 0.1958235        total: 4.46s    remaining: 1.85s
707:    learn: 0.1957617        total: 4.47s    remaining: 1.84s
708:    learn: 0.1956646        total: 4.48s    remaining: 1.84s
709:    learn: 0.1956086        total: 4.48s    remaining: 1.83s
```

```
710:    learn: 0.1955558        total: 4.49s    remaining: 1.82s
711:    learn: 0.1954574        total: 4.5s     remaining: 1.82s
712:    learn: 0.1953871        total: 4.5s     remaining: 1.81s
713:    learn: 0.1953299        total: 4.51s    remaining: 1.8s
714:    learn: 0.1952887        total: 4.51s    remaining: 1.8s
715:    learn: 0.1952297        total: 4.52s    remaining: 1.79s
716:    learn: 0.1951415        total: 4.53s    remaining: 1.79s
717:    learn: 0.1950738        total: 4.53s    remaining: 1.78s
718:    learn: 0.1950161        total: 4.54s    remaining: 1.77s
719:    learn: 0.1949749        total: 4.54s    remaining: 1.77s
720:    learn: 0.1948462        total: 4.55s    remaining: 1.76s
721:    learn: 0.1947736        total: 4.56s    remaining: 1.75s
722:    learn: 0.1947256        total: 4.56s    remaining: 1.75s
723:    learn: 0.1946693        total: 4.57s    remaining: 1.74s
724:    learn: 0.1945913        total: 4.58s    remaining: 1.74s
725:    learn: 0.1945653        total: 4.58s    remaining: 1.73s
726:    learn: 0.1945068        total: 4.59s    remaining: 1.72s
727:    learn: 0.1944264        total: 4.59s    remaining: 1.72s
728:    learn: 0.1943877        total: 4.6s     remaining: 1.71s
729:    learn: 0.1943467        total: 4.61s    remaining: 1.7s
730:    learn: 0.1942933        total: 4.61s    remaining: 1.7s
731:    learn: 0.1942501        total: 4.62s    remaining: 1.69s
732:    learn: 0.1941951        total: 4.63s    remaining: 1.69s
733:    learn: 0.1941275        total: 4.63s    remaining: 1.68s
734:    learn: 0.1940436        total: 4.64s    remaining: 1.67s
735:    learn: 0.1939957        total: 4.65s    remaining: 1.67s
736:    learn: 0.1939473        total: 4.65s    remaining: 1.66s
737:    learn: 0.1938955        total: 4.66s    remaining: 1.65s
738:    learn: 0.1938517        total: 4.66s    remaining: 1.65s
739:    learn: 0.1937896        total: 4.67s    remaining: 1.64s
740:    learn: 0.1937389        total: 4.68s    remaining: 1.63s
741:    learn: 0.1936824        total: 4.68s    remaining: 1.63s
742:    learn: 0.1936308        total: 4.69s    remaining: 1.62s
743:    learn: 0.1935848        total: 4.7s     remaining: 1.61s
744:    learn: 0.1935553        total: 4.7s     remaining: 1.61s
745:    learn: 0.1934806        total: 4.71s    remaining: 1.6s
746:    learn: 0.1934124        total: 4.71s    remaining: 1.6s
747:    learn: 0.1933741        total: 4.72s    remaining: 1.59s
748:    learn: 0.1932505        total: 4.73s    remaining: 1.58s
749:    learn: 0.1932073        total: 4.73s    remaining: 1.58s
750:    learn: 0.1931790        total: 4.74s    remaining: 1.57s
751:    learn: 0.1931243        total: 4.75s    remaining: 1.56s
752:    learn: 0.1930409        total: 4.75s    remaining: 1.56s
753:    learn: 0.1929782        total: 4.76s    remaining: 1.55s
754:    learn: 0.1929231        total: 4.76s    remaining: 1.54s
755:    learn: 0.1928767        total: 4.77s    remaining: 1.54s
756:    learn: 0.1927564        total: 4.78s    remaining: 1.53s
757:    learn: 0.1926995        total: 4.78s    remaining: 1.53s
```

```
758:	learn: 0.1926468	total: 4.79s	remaining: 1.52s
759:	learn: 0.1925954	total: 4.79s	remaining: 1.51s
760:	learn: 0.1925582	total: 4.8s	remaining: 1.51s
761:	learn: 0.1925032	total: 4.81s	remaining: 1.5s
762:	learn: 0.1924362	total: 4.81s	remaining: 1.5s
763:	learn: 0.1923957	total: 4.82s	remaining: 1.49s
764:	learn: 0.1923231	total: 4.83s	remaining: 1.48s
765:	learn: 0.1922752	total: 4.83s	remaining: 1.48s
766:	learn: 0.1922166	total: 4.84s	remaining: 1.47s
767:	learn: 0.1921601	total: 4.84s	remaining: 1.46s
768:	learn: 0.1921294	total: 4.85s	remaining: 1.46s
769:	learn: 0.1920539	total: 4.86s	remaining: 1.45s
770:	learn: 0.1919381	total: 4.86s	remaining: 1.44s
771:	learn: 0.1918772	total: 4.87s	remaining: 1.44s
772:	learn: 0.1918145	total: 4.88s	remaining: 1.43s
773:	learn: 0.1917487	total: 4.88s	remaining: 1.43s
774:	learn: 0.1916815	total: 4.89s	remaining: 1.42s
775:	learn: 0.1916249	total: 4.89s	remaining: 1.41s
776:	learn: 0.1915275	total: 4.9s	remaining: 1.41s
777:	learn: 0.1914847	total: 4.91s	remaining: 1.4s
778:	learn: 0.1914462	total: 4.91s	remaining: 1.39s
779:	learn: 0.1913790	total: 4.92s	remaining: 1.39s
780:	learn: 0.1913073	total: 4.93s	remaining: 1.38s
781:	learn: 0.1912337	total: 4.93s	remaining: 1.38s
782:	learn: 0.1911011	total: 4.94s	remaining: 1.37s
783:	learn: 0.1909886	total: 4.95s	remaining: 1.36s
784:	learn: 0.1909409	total: 4.95s	remaining: 1.36s
785:	learn: 0.1908991	total: 4.96s	remaining: 1.35s
786:	learn: 0.1908315	total: 4.96s	remaining: 1.34s
787:	learn: 0.1908022	total: 4.97s	remaining: 1.34s
788:	learn: 0.1907172	total: 4.98s	remaining: 1.33s
789:	learn: 0.1906736	total: 4.98s	remaining: 1.32s
790:	learn: 0.1905327	total: 4.99s	remaining: 1.32s
791:	learn: 0.1904500	total: 5s	remaining: 1.31s
792:	learn: 0.1904218	total: 5s	remaining: 1.31s
793:	learn: 0.1903906	total: 5.01s	remaining: 1.3s
794:	learn: 0.1902471	total: 5.02s	remaining: 1.29s
795:	learn: 0.1901935	total: 5.02s	remaining: 1.29s
796:	learn: 0.1900416	total: 5.03s	remaining: 1.28s
797:	learn: 0.1900122	total: 5.04s	remaining: 1.27s
798:	learn: 0.1899747	total: 5.04s	remaining: 1.27s
799:	learn: 0.1899227	total: 5.05s	remaining: 1.26s
800:	learn: 0.1898807	total: 5.05s	remaining: 1.25s
801:	learn: 0.1898421	total: 5.06s	remaining: 1.25s
802:	learn: 0.1897930	total: 5.07s	remaining: 1.24s
803:	learn: 0.1897610	total: 5.07s	remaining: 1.24s
804:	learn: 0.1896710	total: 5.08s	remaining: 1.23s
805:	learn: 0.1896135	total: 5.08s	remaining: 1.22s
```

```
806:	learn: 0.1895601	total: 5.09s	remaining: 1.22s
807:	learn: 0.1895369	total: 5.1s	remaining: 1.21s
808:	learn: 0.1894627	total: 5.1s	remaining: 1.2s
809:	learn: 0.1893839	total: 5.11s	remaining: 1.2s
810:	learn: 0.1893088	total: 5.12s	remaining: 1.19s
811:	learn: 0.1892817	total: 5.12s	remaining: 1.19s
812:	learn: 0.1891631	total: 5.13s	remaining: 1.18s
813:	learn: 0.1891230	total: 5.13s	remaining: 1.17s
814:	learn: 0.1890854	total: 5.14s	remaining: 1.17s
815:	learn: 0.1889219	total: 5.15s	remaining: 1.16s
816:	learn: 0.1888709	total: 5.15s	remaining: 1.15s
817:	learn: 0.1888247	total: 5.16s	remaining: 1.15s
818:	learn: 0.1887144	total: 5.17s	remaining: 1.14s
819:	learn: 0.1886630	total: 5.17s	remaining: 1.14s
820:	learn: 0.1886179	total: 5.18s	remaining: 1.13s
821:	learn: 0.1885825	total: 5.18s	remaining: 1.12s
822:	learn: 0.1885123	total: 5.19s	remaining: 1.12s
823:	learn: 0.1884517	total: 5.2s	remaining: 1.11s
824:	learn: 0.1883777	total: 5.21s	remaining: 1.1s
825:	learn: 0.1883313	total: 5.21s	remaining: 1.1s
826:	learn: 0.1882942	total: 5.22s	remaining: 1.09s
827:	learn: 0.1882472	total: 5.22s	remaining: 1.08s
828:	learn: 0.1881965	total: 5.23s	remaining: 1.08s
829:	learn: 0.1880825	total: 5.24s	remaining: 1.07s
830:	learn: 0.1880234	total: 5.24s	remaining: 1.06s
831:	learn: 0.1879763	total: 5.25s	remaining: 1.06s
832:	learn: 0.1879248	total: 5.25s	remaining: 1.05s
833:	learn: 0.1878405	total: 5.26s	remaining: 1.05s
834:	learn: 0.1877591	total: 5.26s	remaining: 1.04s
835:	learn: 0.1877115	total: 5.27s	remaining: 1.03s
836:	learn: 0.1876571	total: 5.28s	remaining: 1.03s
837:	learn: 0.1876166	total: 5.28s	remaining: 1.02s
838:	learn: 0.1875605	total: 5.29s	remaining: 1.01s
839:	learn: 0.1874875	total: 5.3s	remaining: 1.01s
840:	learn: 0.1874378	total: 5.31s	remaining: 1s
841:	learn: 0.1873431	total: 5.31s	remaining: 997ms
842:	learn: 0.1873156	total: 5.32s	remaining: 991ms
843:	learn: 0.1872790	total: 5.33s	remaining: 984ms
844:	learn: 0.1872330	total: 5.33s	remaining: 978ms
845:	learn: 0.1871534	total: 5.34s	remaining: 972ms
846:	learn: 0.1870521	total: 5.35s	remaining: 966ms
847:	learn: 0.1869520	total: 5.35s	remaining: 959ms
848:	learn: 0.1869020	total: 5.36s	remaining: 953ms
849:	learn: 0.1868210	total: 5.37s	remaining: 947ms
850:	learn: 0.1867953	total: 5.37s	remaining: 941ms
851:	learn: 0.1867432	total: 5.38s	remaining: 935ms
852:	learn: 0.1867011	total: 5.39s	remaining: 929ms
853:	learn: 0.1866362	total: 5.39s	remaining: 922ms
```

```
854:	learn: 0.1865869	total: 5.4s	remaining: 916ms
855:	learn: 0.1865473	total: 5.41s	remaining: 910ms
856:	learn: 0.1864878	total: 5.42s	remaining: 904ms
857:	learn: 0.1863979	total: 5.42s	remaining: 897ms
858:	learn: 0.1863061	total: 5.43s	remaining: 891ms
859:	learn: 0.1862172	total: 5.43s	remaining: 885ms
860:	learn: 0.1861699	total: 5.44s	remaining: 878ms
861:	learn: 0.1861276	total: 5.45s	remaining: 872ms
862:	learn: 0.1860910	total: 5.45s	remaining: 866ms
863:	learn: 0.1860256	total: 5.46s	remaining: 859ms
864:	learn: 0.1859875	total: 5.46s	remaining: 853ms
865:	learn: 0.1858766	total: 5.47s	remaining: 847ms
866:	learn: 0.1858215	total: 5.48s	remaining: 840ms
867:	learn: 0.1857580	total: 5.48s	remaining: 834ms
868:	learn: 0.1857137	total: 5.49s	remaining: 828ms
869:	learn: 0.1856185	total: 5.5s	remaining: 821ms
870:	learn: 0.1855621	total: 5.5s	remaining: 815ms
871:	learn: 0.1854975	total: 5.51s	remaining: 809ms
872:	learn: 0.1854594	total: 5.51s	remaining: 802ms
873:	learn: 0.1853966	total: 5.52s	remaining: 796ms
874:	learn: 0.1853544	total: 5.53s	remaining: 790ms
875:	learn: 0.1852546	total: 5.53s	remaining: 783ms
876:	learn: 0.1852105	total: 5.54s	remaining: 777ms
877:	learn: 0.1851749	total: 5.55s	remaining: 771ms
878:	learn: 0.1851255	total: 5.55s	remaining: 764ms
879:	learn: 0.1850571	total: 5.56s	remaining: 758ms
880:	learn: 0.1849887	total: 5.57s	remaining: 752ms
881:	learn: 0.1849271	total: 5.58s	remaining: 746ms
882:	learn: 0.1848930	total: 5.58s	remaining: 740ms
883:	learn: 0.1848156	total: 5.59s	remaining: 733ms
884:	learn: 0.1847537	total: 5.59s	remaining: 727ms
885:	learn: 0.1847240	total: 5.6s	remaining: 721ms
886:	learn: 0.1846561	total: 5.61s	remaining: 714ms
887:	learn: 0.1846367	total: 5.61s	remaining: 708ms
888:	learn: 0.1846023	total: 5.62s	remaining: 702ms
889:	learn: 0.1845636	total: 5.63s	remaining: 695ms
890:	learn: 0.1845252	total: 5.63s	remaining: 689ms
891:	learn: 0.1844703	total: 5.64s	remaining: 683ms
892:	learn: 0.1844277	total: 5.64s	remaining: 676ms
893:	learn: 0.1843491	total: 5.65s	remaining: 670ms
894:	learn: 0.1842706	total: 5.66s	remaining: 664ms
895:	learn: 0.1841790	total: 5.66s	remaining: 657ms
896:	learn: 0.1841322	total: 5.67s	remaining: 651ms
897:	learn: 0.1840940	total: 5.67s	remaining: 645ms
898:	learn: 0.1840513	total: 5.68s	remaining: 638ms
899:	learn: 0.1839500	total: 5.69s	remaining: 632ms
900:	learn: 0.1839051	total: 5.69s	remaining: 625ms
901:	learn: 0.1838690	total: 5.7s	remaining: 619ms
```

```
902:	learn: 0.1838340	total: 5.7s	remaining: 613ms
903:	learn: 0.1838107	total: 5.71s	remaining: 606ms
904:	learn: 0.1837625	total: 5.72s	remaining: 600ms
905:	learn: 0.1836721	total: 5.72s	remaining: 594ms
906:	learn: 0.1835911	total: 5.73s	remaining: 588ms
907:	learn: 0.1834980	total: 5.74s	remaining: 581ms
908:	learn: 0.1834220	total: 5.74s	remaining: 575ms
909:	learn: 0.1833429	total: 5.75s	remaining: 569ms
910:	learn: 0.1832808	total: 5.75s	remaining: 562ms
911:	learn: 0.1832510	total: 5.76s	remaining: 556ms
912:	learn: 0.1832091	total: 5.77s	remaining: 550ms
913:	learn: 0.1830306	total: 5.77s	remaining: 543ms
914:	learn: 0.1829526	total: 5.78s	remaining: 537ms
915:	learn: 0.1828862	total: 5.79s	remaining: 531ms
916:	learn: 0.1828156	total: 5.79s	remaining: 524ms
917:	learn: 0.1827643	total: 5.8s	remaining: 518ms
918:	learn: 0.1827074	total: 5.81s	remaining: 512ms
919:	learn: 0.1826764	total: 5.81s	remaining: 505ms
920:	learn: 0.1826378	total: 5.82s	remaining: 499ms
921:	learn: 0.1825923	total: 5.82s	remaining: 493ms
922:	learn: 0.1825187	total: 5.83s	remaining: 486ms
923:	learn: 0.1824676	total: 5.84s	remaining: 480ms
924:	learn: 0.1824365	total: 5.84s	remaining: 474ms
925:	learn: 0.1824101	total: 5.85s	remaining: 467ms
926:	learn: 0.1823213	total: 5.86s	remaining: 461ms
927:	learn: 0.1822724	total: 5.86s	remaining: 455ms
928:	learn: 0.1822111	total: 5.87s	remaining: 448ms
929:	learn: 0.1821301	total: 5.87s	remaining: 442ms
930:	learn: 0.1820535	total: 5.88s	remaining: 436ms
931:	learn: 0.1819781	total: 5.89s	remaining: 430ms
932:	learn: 0.1819033	total: 5.89s	remaining: 423ms
933:	learn: 0.1818579	total: 5.9s	remaining: 417ms
934:	learn: 0.1817432	total: 5.91s	remaining: 411ms
935:	learn: 0.1817179	total: 5.91s	remaining: 404ms
936:	learn: 0.1816418	total: 5.92s	remaining: 398ms
937:	learn: 0.1816005	total: 5.92s	remaining: 392ms
938:	learn: 0.1815598	total: 5.93s	remaining: 385ms
939:	learn: 0.1815208	total: 5.94s	remaining: 379ms
940:	learn: 0.1814883	total: 5.94s	remaining: 373ms
941:	learn: 0.1814144	total: 5.95s	remaining: 366ms
942:	learn: 0.1813459	total: 5.96s	remaining: 360ms
943:	learn: 0.1813011	total: 5.96s	remaining: 354ms
944:	learn: 0.1812364	total: 5.97s	remaining: 347ms
945:	learn: 0.1811484	total: 5.97s	remaining: 341ms
946:	learn: 0.1811265	total: 5.98s	remaining: 335ms
947:	learn: 0.1810627	total: 5.99s	remaining: 328ms
948:	learn: 0.1810274	total: 5.99s	remaining: 322ms
949:	learn: 0.1809470	total: 6s	remaining: 316ms
```

```
950:    learn: 0.1808604        total: 6.01s    remaining: 309ms
951:    learn: 0.1808094        total: 6.01s    remaining: 303ms
952:    learn: 0.1807565        total: 6.02s    remaining: 297ms
953:    learn: 0.1806931        total: 6.02s    remaining: 290ms
954:    learn: 0.1806400        total: 6.03s    remaining: 284ms
955:    learn: 0.1806084        total: 6.04s    remaining: 278ms
956:    learn: 0.1805417        total: 6.04s    remaining: 272ms
957:    learn: 0.1804648        total: 6.05s    remaining: 265ms
958:    learn: 0.1804367        total: 6.05s    remaining: 259ms
959:    learn: 0.1803765        total: 6.06s    remaining: 253ms
960:    learn: 0.1803410        total: 6.07s    remaining: 246ms
961:    learn: 0.1803168        total: 6.07s    remaining: 240ms
962:    learn: 0.1802625        total: 6.08s    remaining: 234ms
963:    learn: 0.1801915        total: 6.09s    remaining: 227ms
964:    learn: 0.1801313        total: 6.09s    remaining: 221ms
965:    learn: 0.1801005        total: 6.1s     remaining: 215ms
966:    learn: 0.1799676        total: 6.11s    remaining: 208ms
967:    learn: 0.1799060        total: 6.11s    remaining: 202ms
968:    learn: 0.1798571        total: 6.12s    remaining: 196ms
969:    learn: 0.1798203        total: 6.12s    remaining: 189ms
970:    learn: 0.1797556        total: 6.13s    remaining: 183ms
971:    learn: 0.1797036        total: 6.14s    remaining: 177ms
972:    learn: 0.1796488        total: 6.14s    remaining: 170ms
973:    learn: 0.1796218        total: 6.15s    remaining: 164ms
974:    learn: 0.1795801        total: 6.16s    remaining: 158ms
975:    learn: 0.1795017        total: 6.16s    remaining: 152ms
976:    learn: 0.1794816        total: 6.17s    remaining: 145ms
977:    learn: 0.1794352        total: 6.18s    remaining: 139ms
978:    learn: 0.1793843        total: 6.18s    remaining: 133ms
979:    learn: 0.1793591        total: 6.19s    remaining: 126ms
980:    learn: 0.1792623        total: 6.2s     remaining: 120ms
981:    learn: 0.1792027        total: 6.2s     remaining: 114ms
982:    learn: 0.1790638        total: 6.21s    remaining: 107ms
983:    learn: 0.1790086        total: 6.21s    remaining: 101ms
984:    learn: 0.1789560        total: 6.22s    remaining: 94.7ms
985:    learn: 0.1789085        total: 6.23s    remaining: 88.4ms
986:    learn: 0.1788723        total: 6.23s    remaining: 82.1ms
987:    learn: 0.1788502        total: 6.24s    remaining: 75.8ms
988:    learn: 0.1787975        total: 6.25s    remaining: 69.5ms
989:    learn: 0.1787243        total: 6.25s    remaining: 63.2ms
990:    learn: 0.1786958        total: 6.26s    remaining: 56.8ms
991:    learn: 0.1786443        total: 6.27s    remaining: 50.5ms
992:    learn: 0.1786059        total: 6.27s    remaining: 44.2ms
993:    learn: 0.1785555        total: 6.28s    remaining: 37.9ms
994:    learn: 0.1785192        total: 6.29s    remaining: 31.6ms
995:    learn: 0.1784798        total: 6.29s    remaining: 25.3ms
996:    learn: 0.1784028        total: 6.3s     remaining: 19ms
997:    learn: 0.1783568        total: 6.3s     remaining: 12.6ms
```

```
998:    learn: 0.1783011        total: 6.31s    remaining: 6.32ms
999:    learn: 0.1782462        total: 6.32s    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',
                              <…
                                            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)),
                             ('KNN', KNeighborsClassifier(n_neighbors=3))],
                 voting='soft')
```

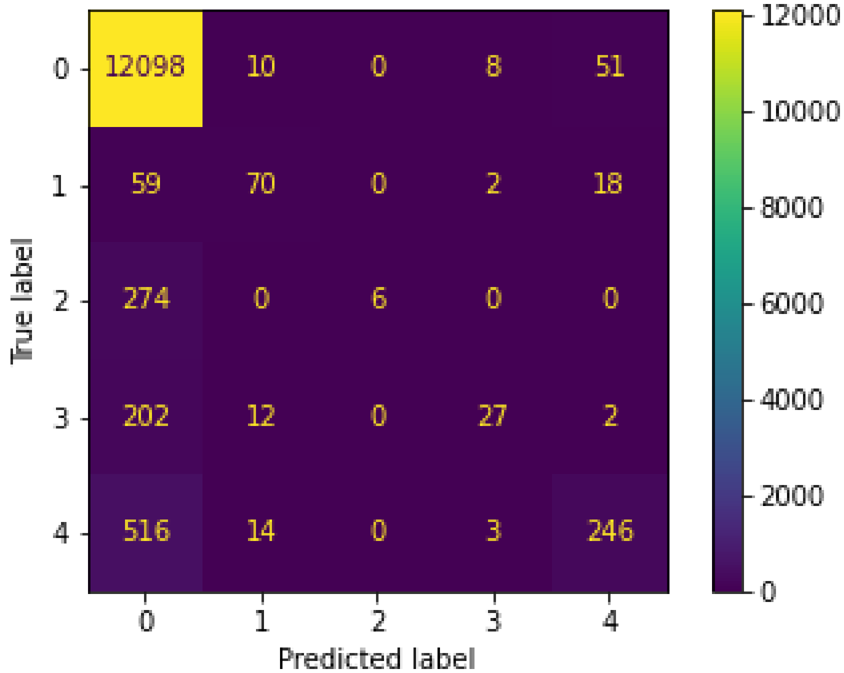
[104]:
```
y_pred = vot_soft.predict(X_test)
```

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

[105]: 91.40108679688647

[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
0x7f44b4c45790>
```



[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 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")
```

[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  102.326070  0.506785  2.4993       0
29561  127.9872  102.490830  0.510428  2.4997       0
29562  127.9657  102.498159  0.510361  2.4999       0
29563  128.9050  102.607440  0.512985  2.5000       0
29564  128.4784  102.560015  0.511792  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       1
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   94.713206  0.5044  2.3501       0
15680  173.8879   96.256515  0.4875  2.3948       0
15706  172.7409   97.818285  0.5074  2.4185       0
```

```
15707  174.8540   97.349782  0.4967  2.4147       0
15708  172.7833   96.989636  0.4784  2.4165       0
16123   76.3119  121.788437  0.3927  3.0026       0
16907  173.0850   78.984443  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  102.326070  0.506785  2.4993       0
29561  127.9872  102.490830  0.510428  2.4997       0
29562  127.9657  102.498159  0.510361  2.4999       0
29563  128.9050  102.607440  0.512985  2.5000       0
29564  128.4784  102.560015  0.511792  2.5000       0

[29534 rows x 5 columns]
column GR
Percentiles: 25th=77.174, 75th=102.911, IQR=25.736
InterQuartile Range Outliers-:
             DT       GR    NPHI    RHOB  FACIES
1342   144.1047  35.5685  0.6130  1.2752       3
1516   115.6053  37.5339  0.6715  1.1197       3
1517   116.5264  37.2150  0.6474  1.2269       3
1625   149.5008  36.3442  0.6133  1.1143       3
1626   150.9417  29.3642  0.6122  1.0951       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

[1851 rows x 5 columns]
(1851, 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  102.326070  0.506785  2.4993       0
29561  127.9872  102.490830  0.510428  2.4997       0
29562  127.9657  102.498159  0.510361  2.4999       0
29563  128.9050  102.607440  0.512985  2.5000       0
29564  128.4784  102.560015  0.511792  2.5000       0

[27683 rows x 5 columns]
column NPHI
Percentiles: 25th=0.468, 75th=0.550, IQR=0.083
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

[1568 rows x 5 columns]
(1568, 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  102.326070  0.506785  2.4993       0
29561  127.9872  102.490830  0.510428  2.4997       0
29562  127.9657  102.498159  0.510361  2.4999       0
29563  128.9050  102.607440  0.512985  2.5000       0
29564  128.4784  102.560015  0.511792  2.5000       0

[26115 rows x 5 columns]
column RHOB
Percentiles: 25th=2.207, 75th=2.416, IQR=0.209
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

[1440 rows x 5 columns]
(1440, 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  102.326070  0.506785  2.4993       0
29561  127.9872  102.490830  0.510428  2.4997       0
29562  127.9657  102.498159  0.510361  2.4999       0
29563  128.9050  102.607440  0.512985  2.5000       0
29564  128.4784  102.560015  0.511792  2.5000       0

[24675 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      0.417594 -0.265769 -0.554201 -0.069035       0
1      0.368665 -0.208540 -0.685637 -0.076038       0
2      0.413534 -0.253584 -0.742547 -0.091546       0
3      0.479706 -0.325497 -0.357724 -0.113057       0
4      0.529023 -0.372035  0.365854 -0.138569       0
…          …         …         …         …         …
24670  0.093904  0.435043 -0.031369  0.930465       0
24671  0.156481  0.442736  0.017991  0.932466       0
24672  0.155452  0.443078  0.017092  0.933467       0
24673  0.200417  0.448180  0.052639  0.933967       0
24674  0.179995  0.445966  0.036476  0.933967       0

[24675 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      0.417594 -0.265769 -0.554201 -0.069035
1      0.368665 -0.208540 -0.685637 -0.076038
2      0.413534 -0.253584 -0.742547 -0.091546
3      0.479706 -0.325497 -0.357724 -0.113057
4      0.529023 -0.372035  0.365854 -0.138569
…           …         …         …         …
24670  0.093904  0.435043 -0.031369  0.930465
24671  0.156481  0.442736  0.017991  0.932466
24672  0.155452  0.443078  0.017092  0.933467
24673  0.200417  0.448180  0.052639  0.933967
24674  0.179995  0.445966  0.036476  0.933967

[24675 rows x 4 columns]
```

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

[134]:
```
             FACIES
count  24675.000000
mean       0.327254
std        1.016689
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
```

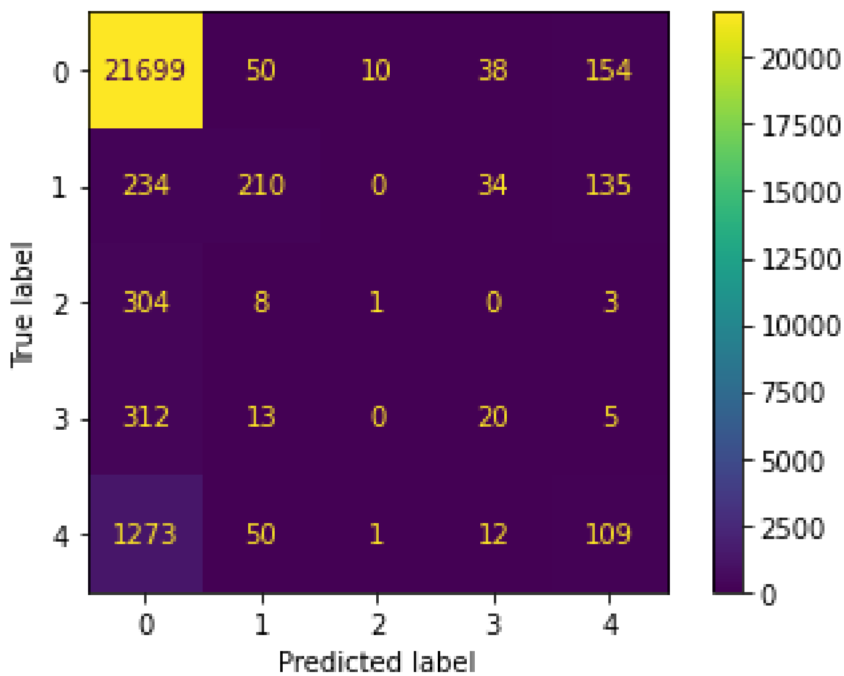
[137]: 89.31712259371834

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

[138]:
```
array([[21699,    50,    10,    38,   154],
       [  234,   210,     0,    34,   135],
       [  304,     8,     1,     0,     3],
       [  312,    13,     0,    20,     5],
       [ 1273,    50,     1,    12,   109]])
```

[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 0x7f44b47c0820>



[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)
```

[ ]:
```
```