In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [4]:
```
df=pd.read_csv('train.csv')
df.head()
```

Out[4]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_decelerations	prolongued_decelerations	abnormal_short_
0	142.0	0.000	0.000	0.007	0.000	0.0	0.0	
1	122.0	0.000	0.000	0.006	0.002	0.0	0.0	
2	129.0	0.005	0.003	0.001	0.000	0.0	0.0	
3	136.0	0.006	0.000	0.008	0.000	0.0	0.0	
4	144.0	0.000	0.000	0.006	0.000	0.0	0.0	

5 rows × 22 columns

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1700 entries, 0 to 1699
Data columns (total 22 columns):
 #   Column                                                  Non-Null Count  Dtype  
---  ------                                                  --------------  -----  
 0   baseline value                                          1700 non-null   float64
 1   accelerations                                           1700 non-null   float64
 2   fetal_movement                                          1700 non-null   float64
 3   uterine_contractions                                    1700 non-null   float64
 4   light_decelerations                                     1700 non-null   float64
 5   severe_decelerations                                    1700 non-null   float64
 6   prolongued_decelerations                                1700 non-null   float64
 7   abnormal_short_term_variability                         1700 non-null   float64
 8   mean_value_of_short_term_variability                    1700 non-null   float64
 9   percentage_of_time_with_abnormal_long_term_variability  1700 non-null   float64
 10  mean_value_of_long_term_variability                     1700 non-null   float64
 11  histogram_width                                         1700 non-null   float64
 12  histogram_min                                           1700 non-null   float64
 13  histogram_max                                           1700 non-null   float64
 14  histogram_number_of_peaks                               1700 non-null   float64
 15  histogram_number_of_zeroes                              1700 non-null   float64
 16  histogram_mode                                          1700 non-null   float64
 17  histogram_mean                                          1700 non-null   float64
 18  histogram_median                                        1700 non-null   float64
 19  histogram_variance                                      1700 non-null   float64
 20  histogram_tendency                                      1700 non-null   float64
 21  fetal_health                                            1700 non-null   float64
dtypes: float64(22)
memory usage: 292.3 KB
```

In [6]:
```
df.shape
```

Out[6]:
```
(1700, 22)
```

In [7]:
```
df.isna().sum()
```

Out[7]:
```
baseline value                                            0
accelerations                                             0
fetal_movement                                            0
uterine_contractions                                      0
light_decelerations                                       0
severe_decelerations                                      0
prolongued_decelerations                                  0
abnormal_short_term_variability                           0
mean_value_of_short_term_variability                      0
percentage_of_time_with_abnormal_long_term_variability    0
mean_value_of_long_term_variability                       0
histogram_width                                           0
histogram_min                                             0
histogram_max                                             0
histogram_number_of_peaks                                 0
histogram_number_of_zeroes                                0
histogram_mode                                            0
histogram_mean                                            0
histogram_median                                          0
histogram_variance                                        0
histogram_tendency                                        0
fetal_health                                              0
dtype: int64
```

In [8]:
```
df.describe()
```

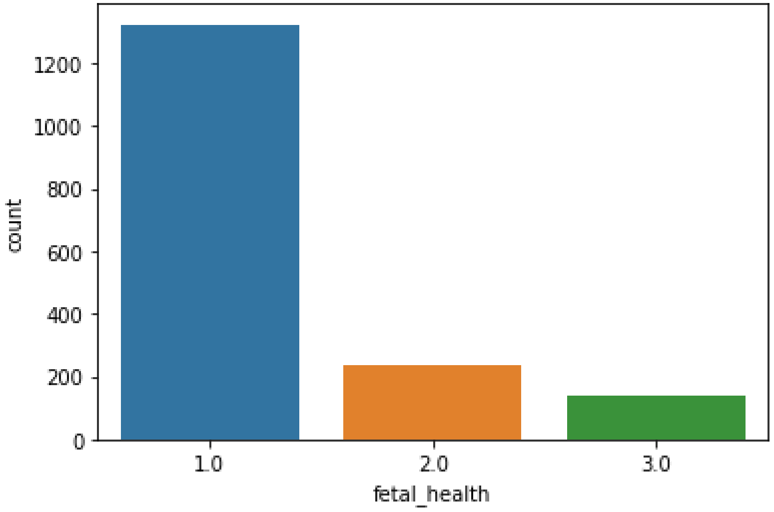
Out[8]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_decelerations	prolongued_decelerations	abnorma
count	1700.000000	1700.000000	1700.000000	1700.000000	1700.000000	1700.000000	1700.000000	
mean	133.213529	0.003212	0.010211	0.004356	0.001899	0.000004	0.000158	
std	9.873344	0.003888	0.050124	0.002943	0.002976	0.000059	0.000587	
min	106.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	126.000000	0.000000	0.000000	0.002000	0.000000	0.000000	0.000000	
50%	133.000000	0.002000	0.000000	0.004000	0.000000	0.000000	0.000000	
75%	140.000000	0.006000	0.003000	0.006000	0.003000	0.000000	0.000000	
max	159.000000	0.019000	0.481000	0.015000	0.015000	0.001000	0.005000	

8 rows × 22 columns

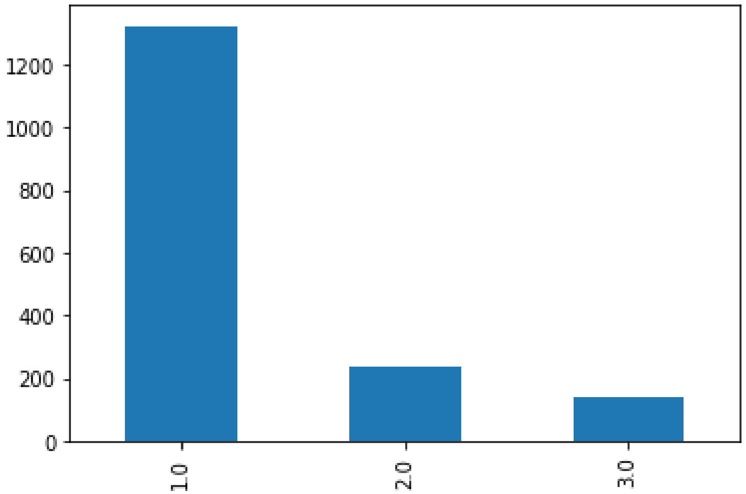
In [9]:
```
#To understand the different types in our target variable (fatal_health)
sns.countplot(df.fetal_health)
```

Out[9]: <AxesSubplot:xlabel='fetal_health', ylabel='count'>



In [10]:
```
df['fetal_health'].value_counts().plot(kind='bar')
```

Out[10]: <AxesSubplot:>

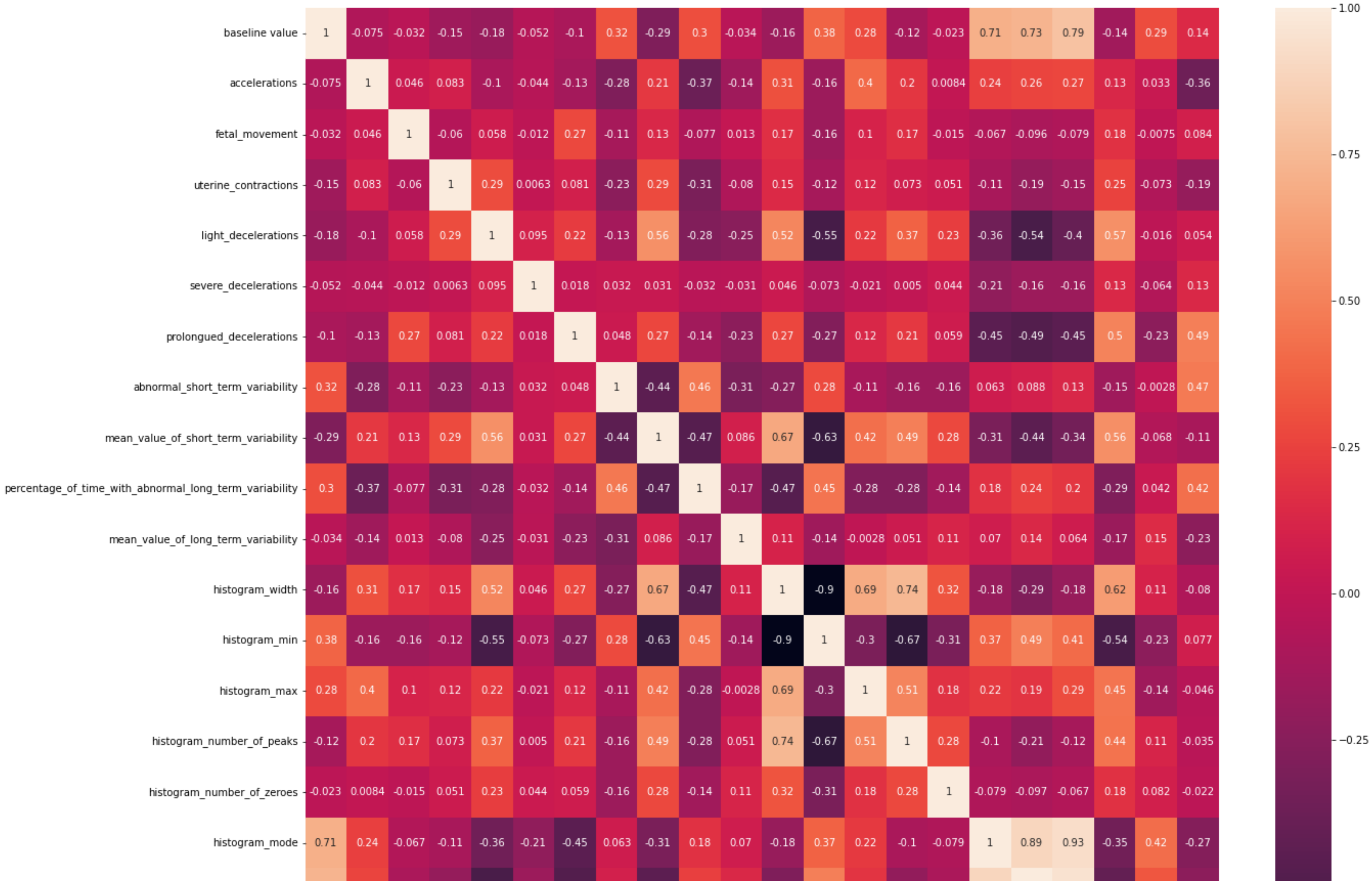


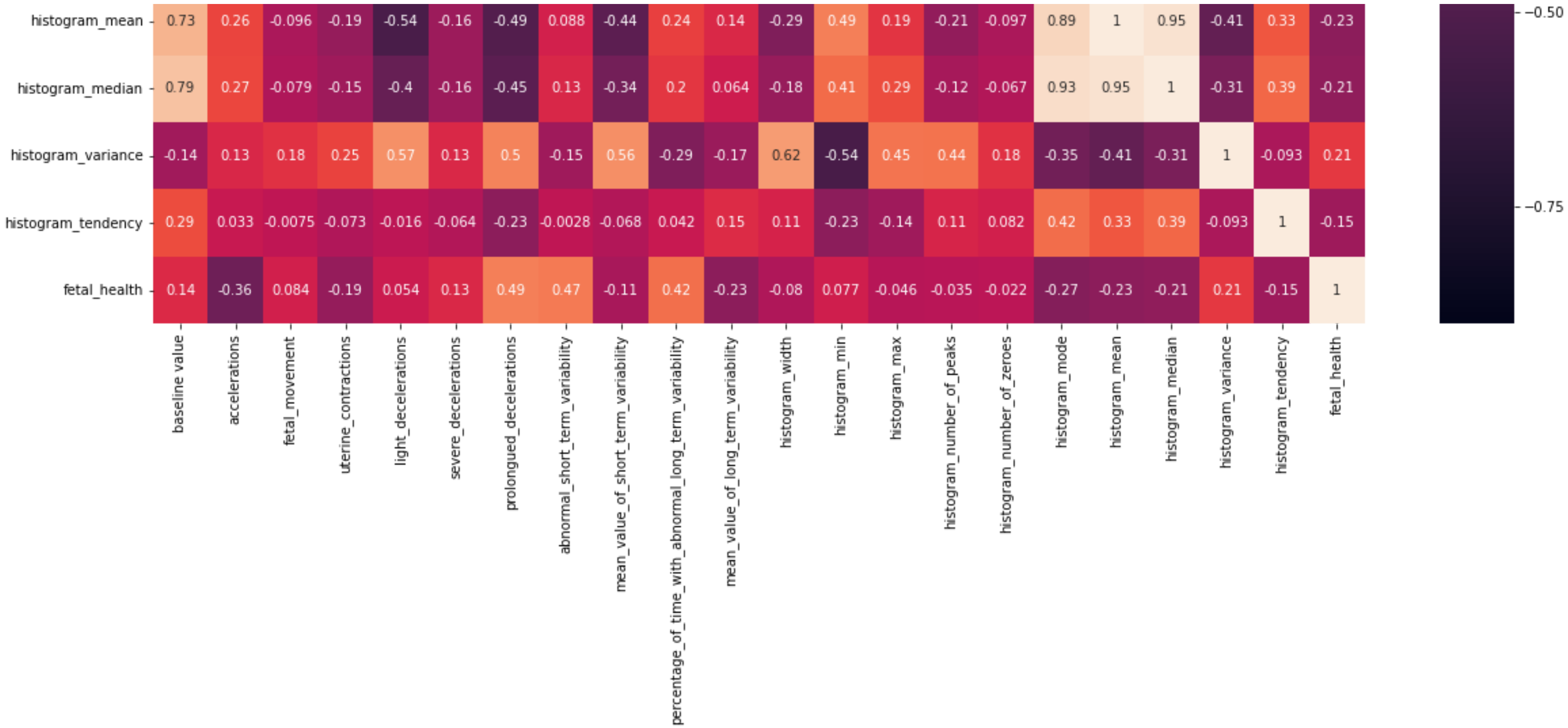
In [12]:
```
#to find all datatypes in our file
cats = list(df.select_dtypes(include=['object','bool']) )
nums = list(df.select_dtypes(include=['int64','float64']))
print(cats)
print(nums)
[]
['baseline value', 'accelerations', 'fetal_movement',
'uterine_contractions', 'light_decelerations', 'severe_decelerations',
'prolongued_decelerations', 'abnormal_short_term_variability',
'mean_value_of_short_term_variability',
'percentage_of_time_with_abnormal_long_term_variability',
'mean_value_of_long_term_variability', 'histogram_width',
'histogram_min', 'histogram_max', 'histogram_number_of_peaks',
'histogram_number_of_zeroes', 'histogram_mode', 'histogram_mean',
'histogram_median', 'histogram_variance', 'histogram_tendency',
'fetal_health']
#splitting the dataset X,y
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=21)
```

```
[]
['baseline value', 'accelerations', 'fetal_movement', 'uterine_contractions', 'light_decelerations', 'severe_decelerati
ons', 'prolongued_decelerations', 'abnormal_short_term_variability', 'mean_value_of_short_term_variability', 'percentag
e_of_time_with_abnormal_long_term_variability', 'mean_value_of_long_term_variability', 'histogram_width', 'histogram_mi
n', 'histogram_max', 'histogram_number_of_peaks', 'histogram_number_of_zeroes', 'histogram_mode', 'histogram_mean', 'hi
stogram_median', 'histogram_variance', 'histogram_tendency', 'fetal_health']
```

In [13]:
```python
#Finding the correlation of df
corr=df.corr()
plt.figure(figsize=(20,20))
sns.heatmap(corr,annot=True)
```

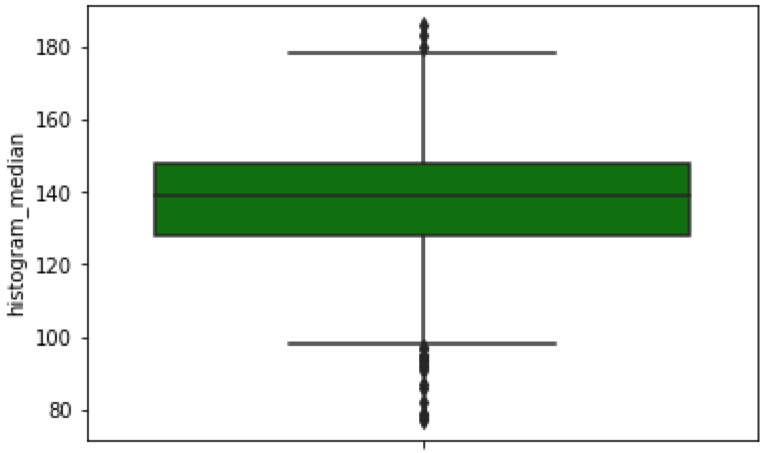
Out[13]: <AxesSubplot:>



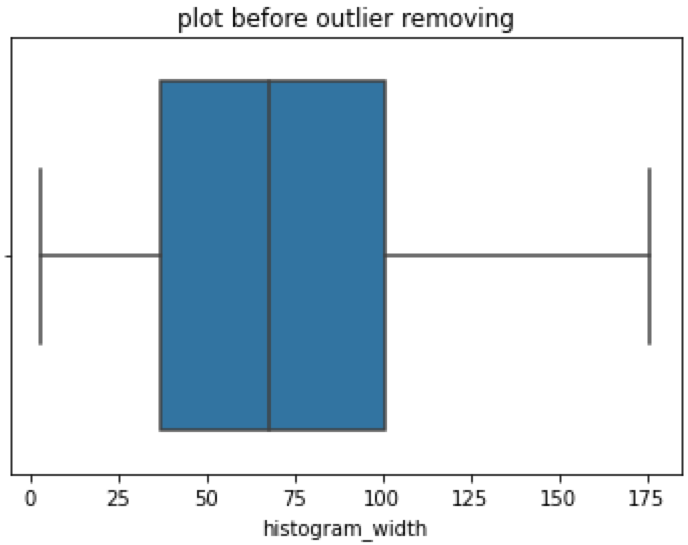

histogram_mean
histogram_median
histogram_variance
histogram_tendency
fetal_health
0.73 0.26 -0.096 -0.19 -0.54 -0.16 -0.49 0.088 -0.44 0.24 0.14 -0.29 0.49 0.19 -0.21 -0.097 0.89 1 0.95 -0.41 0.33 -0.23
0.79 0.27 -0.079 -0.15 -0.4 -0.16 -0.45 0.13 -0.34 0.2 0.064 -0.18 0.41 0.29 -0.12 -0.067 0.93 0.95 1 -0.31 0.39 -0.21
-0.14 0.13 0.18 0.25 0.57 0.13 0.5 -0.15 0.56 -0.29 -0.17 0.62 -0.54 0.45 0.44 0.18 -0.35 -0.41 -0.31 1 -0.093 0.21
0.29 0.033 -0.0075 -0.073 -0.016 -0.064 -0.23 -0.0028 -0.068 0.042 0.15 0.11 -0.23 -0.14 0.11 0.082 0.42 0.33 0.39 -0.093 1 -0.15
0.14 -0.36 0.084 -0.19 0.054 0.13 0.49 0.47 -0.11 0.42 -0.23 -0.08 0.077 -0.046 -0.035 -0.022 -0.27 -0.23 -0.21 0.21 -0.15 1
baseline value
accelerations
fetal_movement
uterine_contractions
light_decelerations
severe_decelerations
prolongued_decelerations
abnormal_short_term_variability
mean_value_of_short_term_variability
percentage_of_time_with_abnormal_long_term_variability
mean_value_of_long_term_variability
histogram_width
histogram_min
histogram_max
histogram_number_of_peaks
histogram_number_of_zeroes
histogram_mode
histogram_mean
histogram_median
histogram_variance
histogram_tendency
fetal_health
−0.50
−0.75

In [14]:

```
#Finding the outlier using boxplot
for i in range(0, len(nums )):
 sns.boxplot(y=df[nums[i]],color='green',orient='v')
 plt.show()
```



In [15]:
```
#Removing the outliers
for i in range(len(nums)):
    sns.boxplot(df[nums[i]])
    plt.title(nums[i])
    plt.title("plot before outlier removing")
    plt.show()
    def drop_outliers(df, field_name):
        iqr = 1.5 * (np.percentile(df[field_name], 75) - np.percentile(df[field_name], 25))
        df.drop(df[df[field_name] > (iqr + np.percentile(df[field_name], 75))].index, inplace=True)
        df.drop(df[df[field_name] < (np.percentile(df[field_name], 25)- iqr)].index, inplace=True)
        iqr = 1.5 * (np.percentile(df[field_name], 75) - np.percentile(df[field_name], 25))
        df.drop(df[df[field_name] > (iqr + np.percentile(df[field_name], 75))].index, inplace=True)
        df.drop(df[df[field_name] < (np.percentile(df[field_name], 25)- iqr)].index, inplace=True)
        drop_outliers(df, nums[i])
        sns.boxplot(df[nums[i]])
        plt.title("plot after outlier removing")
        plt.show()
```



In [16]:
```
#Feature scaling
from sklearn.preprocessing import StandardScaler
SC=StandardScaler()
X_train=SC.fit_transform(X_train)
X_test=SC.fit_transform(X_test)
print(X_train)
```

```
[[-1.46513509 -0.83485626 -0.20820521 ... -1.18281773 -0.60623421
  -2.14821249]
 [ 0.16393359 -0.83485626 -0.0746156  ... -0.08025974 -0.53911612
   1.13129086]
 [-1.2615015   2.2251405  -0.00782079 ... -0.42480911 -0.37132092
  -2.14821249]
 ...
 [ 1.28391831  1.71514104 -0.20820521 ...  1.160118   -0.50555708
  -0.50846081]
 [ 1.08028472 -0.83485626  0.6823922  ...  0.74665876 -0.06928955
   1.13129086]
 [ 0.97846793 -0.3248568   0.94957142 ...  0.74665876  1.00459976
  -0.50846081]]
```

In [17]:
```
print(X_test)
```

```
[[ 0.92243937 -0.80741306  0.01293697 ...  0.39981121 -0.48257105
   1.06361291]
 [-1.28386413 -0.80741306 -0.20203067 ... -2.47496427  1.27487407
  -0.59523291]
 [-1.08329109 -0.80741306 -0.20203067 ... -0.83223542 -0.37919193
   1.06361291]
 ...
 [-1.18357761  0.24506147 -0.20203067 ... -0.76378839  0.34446195
   1.06361291]
 [-0.78243152  1.297536   -0.08627887 ... -0.010871    1.58501145
   1.06361291]
 [ 0.01986066 -0.80741306 -0.1854947  ... -0.010871   -0.65486959
  -0.59523291]]
```

In [18]:

```
#Building the Model
from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)
classifier.fit(X_train,y_train)
KNeighborsClassifier()
y_pred=classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1),y_test.reshape(len
(y_test),1)),1))
```

```
[[1. 1.]
 [3. 3.]
 [1. 1.]
 ...
 [1. 1.]
 [1. 1.]
 [2. 1.]]
```

In [19]:

```
#Cross validation
parameters ={ 'n_neighbors' : [5,7,9,11,13,15],
 'weights' : ['uniform','distance'],
 'metric' : ['minkowski','euclidean','manhattan']}
from sklearn.model_selection import RandomizedSearchCV
cv = RandomizedSearchCV(classifier,parameters ,cv=5)
cv.fit(X_train,y_train)
RandomizedSearchCV(cv=5, estimator=KNeighborsClassifier(),param_distributions={'metric': ['minkowski','euclidean','manha
y_pred = cv.predict(X_test)
```

In [21]:

```
from sklearn.metrics import accuracy_score
print('\n Hyperparametric tuned knn accuracy:',accuracy_score(y_pred,y_test))
```

```
 Hyperparametric tuned knn accuracy: 0.8960784313725491
```

```
In [22]: test_set=pd.read_csv('test.csv')
         y_pre =classifier.predict(test_set)
         print(y_pre )
```

```
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

```
In [25]: #Saving the file in csv format
         y = pd.DataFrame(y_pred).astype(int)
         y.to_csv('Result.csv')
```

```
In [ ]:
```