Thyroid Disease Detection

Problem Statement: Thyroid disease is a common cause of medical diagnosis and prediction, with an onset that is difficult to forecast in medical research. The thyroid gland is one of our body's most vital organs. Thyroid hormone releases are responsible for metabolic regulation. Hyperthyroidism and hypothyroidism are one of the two common diseases of the thyroid that releases thyroid hormones in regulating the rate of body's metabolism. The main goal is to predict the estimated risk on a patient's chance of obtaining thyroid disease or not.

In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import seaborn as sns

%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

For classification problems

    Check Null Values
    Check Balance Data

In [3]:
```
df = pd.read_csv('datafiles/hypothyroid.csv')
df.head()
```

Out[3]:

	age	sex	on thyroxine	query on thyroxine	on antithyroid medication	sick	pregnant	thyroid surgery	I131 treatment	hypoth
0	41	F	f	f	f	f	f	f	f	
1	23	F	f	f	f	f	f	f	f	
2	46	M	f	f	f	f	f	f	f	
3	70	F	t	f	f	f	f	f	f	
4	70	F	f	f	f	f	f	f	f	

5 rows × 30 columns

In [4]:
```
df.shape
```

Out[4]:
```
(3772, 30)
```

In [5]:
```
df.describe().T
```

Out[5]:

	count	unique	top	freq
age	3772	94	59	95
sex	3772	3	F	2480
on thyroxine	3772	2	f	3308
query on thyroxine	3772	2	f	3722
on antithyroid medication	3772	2	f	3729
sick	3772	2	f	3625
pregnant	3772	2	f	3719
thyroid surgery	3772	2	f	3719
I131 treatment	3772	2	f	3713
query hypothyroid	3772	2	f	3538
query hyperthyroid	3772	2	f	3535
lithium	3772	2	f	3754
goitre	3772	2	f	3738
tumor	3772	2	f	3676
hypopituitary	3772	2	f	3771
psych	3772	2	f	3588
TSH measured	3772	2	t	3403
TSH	3772	288	?	369
T3 measured	3772	2	t	3003
T3	3772	70	?	769
TT4 measured	3772	2	t	3541
TT4	3772	242	?	231
T4U measured	3772	2	t	3385
T4U	3772	147	?	387
FTI measured	3772	2	t	3387
FTI	3772	235	?	385
TBG measured	3772	1	f	3772
TBG	3772	1	?	3772
referral source	3772	5	other	2201
binaryClass	3772	2	P	3481

Need to remove ? from the data

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3772 entries, 0 to 3771
Data columns (total 30 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   age                        3772 non-null   object
 1   sex                        3772 non-null   object
 2   on thyroxine               3772 non-null   object
 3   query on thyroxine         3772 non-null   object
 4   on antithyroid medication  3772 non-null   object
 5   sick                       3772 non-null   object
 6   pregnant                   3772 non-null   object
 7   thyroid surgery            3772 non-null   object
 8   I131 treatment             3772 non-null   object
 9   query hypothyroid          3772 non-null   object
 10  query hyperthyroid         3772 non-null   object
 11  lithium                    3772 non-null   object
 12  goitre                     3772 non-null   object
 13  tumor                      3772 non-null   object
 14  hypopituitary              3772 non-null   object
 15  psych                      3772 non-null   object
 16  TSH measured               3772 non-null   object
 17  TSH                        3772 non-null   object
 18  T3 measured                3772 non-null   object
 19  T3                         3772 non-null   object
 20  TT4 measured               3772 non-null   object
 21  TT4                        3772 non-null   object
 22  T4U measured               3772 non-null   object
 23  T4U                        3772 non-null   object
 24  FTI measured               3772 non-null   object
 25  FTI                        3772 non-null   object
 26  TBG measured               3772 non-null   object
 27  TBG                        3772 non-null   object
 28  referral source            3772 non-null   object
 29  binaryClass                3772 non-null   object
dtypes: object(30)
memory usage: 884.2+ KB
```

In [7]:
```
df["binaryClass"].value_counts()
```

Out[7]:
```
binaryClass
P    3481
N     291
Name: count, dtype: int64
```

Need to generate some N values to balance the dataset

In [8]:
```
df["binaryClass"]=df["binaryClass"].map({"P":0,"N":1})
```

replace t and f values with 1 and 0 respectively

In [9]:
```
df=df.replace({"t":1,"f":0})
```

replace ? with np.NAN in the dataset

In [10]:
```
df=df.replace({"?":np.NAN})
```

In [11]:
```
df.head()
```

Out[11]:

	age	sex	on thyroxine	query on thyroxine	on antithyroid medication	sick	pregnant	thyroid surgery	I131 treatment	hypoth
0	41	F	0	0	0	0	0	0	0	
1	23	F	0	0	0	0	0	0	0	
2	46	M	0	0	0	0	0	0	0	
3	70	F	1	0	0	0	0	0	0	
4	70	F	0	0	0	0	0	0	0	

5 rows × 30 columns

In [12]:
```
len(df.columns),df.columns
```

Out[12]:
```
(30,
 Index(['age', 'sex', 'on thyroxine', 'query on thyroxine',
        'on antithyroid medication', 'sick', 'pregnant', 'thyroid surgery',
        'I131 treatment', 'query hypothyroid', 'query hyperthyroid', 'lithium',
        'goitre', 'tumor', 'hypopituitary', 'psych', 'TSH measured', 'TSH',
        'T3 measured', 'T3', 'TT4 measured', 'TT4', 'T4U measured', 'T4U',
        'FTI measured', 'FTI', 'TBG measured', 'TBG', 'referral source',
        'binaryClass'],
       dtype='object'))
```

In [13]:
```
numerical_cols = df.select_dtypes(exclude='object').columns
categorical_cols = df.select_dtypes(include='object').columns
```

In [14]:
```
len(numerical_cols), numerical_cols
```

Out[14]:
```
(22,
 Index(['on thyroxine', 'query on thyroxine', 'on antithyroid medication',
        'sick', 'pregnant', 'thyroid surgery', 'I131 treatment',
        'query hypothyroid', 'query hyperthyroid', 'lithium', 'goitre', 'tumor',
        'hypopituitary', 'psych', 'TSH measured', 'T3 measured', 'TT4 measured',
        'T4U measured', 'FTI measured', 'TBG measured', 'TBG', 'binaryClass'],
       dtype='object'))
```

In [15]:
```
len(categorical_cols), categorical_cols
```

Out[15]:
```
(8,
 Index(['age', 'sex', 'TSH', 'T3', 'TT4', 'T4U', 'FTI', 'referral source'], dtype='
object'))
```

In [16]:
```
df.isnull().sum()
```

Out[16]:
```
age                            1
sex                          150
on thyroxine                   0
query on thyroxine             0
on antithyroid medication      0
sick                           0
pregnant                       0
thyroid surgery                0
I131 treatment                 0
query hypothyroid              0
query hyperthyroid             0
lithium                        0
goitre                         0
tumor                          0
hypopituitary                  0
psych                          0
TSH measured                   0
TSH                          369
T3 measured                    0
T3                           769
TT4 measured                   0
TT4                          231
T4U measured                   0
T4U                          387
FTI measured                   0
FTI                          385
TBG measured                   0
TBG                         3772
referral source                0
binaryClass                    0
dtype: int64
```

In [17]:
```
def printvalues(col):
    print(f"{col}")
    print(f"{df[col].value_counts()}")
    print(f"Total Null Values: {df[col].isnull().sum()}")
    print("*"*15)
```

In [18]:
```
for cat in categorical_cols:
    printvalues(cat)
```

```
age
age
59     95
60     91
70     90
73     81
55     81
       ..
10      1
4       1
5       1
455     1
6       1
Name: count, Length: 93, dtype: int64
Total Null Values: 1
***************
sex
sex
F    2480
M    1142
Name: count, dtype: int64
Total Null Values: 150
***************
TSH
TSH
0.2     116
1.3     105
1.1      97
1.4      91
1.5      80
       ...
86        1
18.4      1
89        1
29        1
40        1
Name: count, Length: 287, dtype: int64
Total Null Values: 369
***************
T3
T3
2      238
1.8    207
2.2    201
1.9    189
2.1    184
      ...
6.7      1
7.3      1
4.6      1
5.2      1
6.6      1
Name: count, Length: 69, dtype: int64
Total Null Values: 769
***************
TT4
```

```
TT4
101    71
93     67
98     63
103    63
102    59
       ..
289     1
240     1
43      1
34      1
258     1
Name: count, Length: 241, dtype: int64
Total Null Values: 231
***************
T4U
T4U
0.99    95
0.9     93
1.01    91
1       90
0.92    89
        ..
1.84     1
0.57     1
2.03     1
1.97     1
0.25     1
Name: count, Length: 146, dtype: int64
Total Null Values: 387
***************
FTI
FTI
100    73
93     70
114    65
98     64
107    64
       ..
349     1
221     1
187     1
39      1
227     1
Name: count, Length: 234, dtype: int64
Total Null Values: 385
***************
referral source
referral source
other    2201
SVI      1034
SVHC      386
STMW      112
SVHD       39
Name: count, dtype: int64
Total Null Values: 0
```

****

In [19]:
```
df=df.replace({"F":1,"M":0})
```

In [20]:
```
df["referral source"].value_counts()
```

Out[20]:
```
referral source
other    2201
SVI      1034
SVHC      386
STMW      112
SVHD       39
Name: count, dtype: int64
```

In [21]:
```
del df["referral source"]
```

In [22]:
```
df["TBG measured"].value_counts()
```

Out[22]:
```
TBG measured
0    3772
Name: count, dtype: int64
```

In [23]:
```
df.dtypes
```

Out[23]:
```
age                          object
sex                         float64
on thyroxine                  int64
query on thyroxine            int64
on antithyroid medication     int64
sick                          int64
pregnant                      int64
thyroid surgery               int64
I131 treatment                int64
query hypothyroid             int64
query hyperthyroid            int64
lithium                       int64
goitre                        int64
tumor                         int64
hypopituitary                 int64
psych                         int64
TSH measured                  int64
TSH                          object
T3 measured                   int64
T3                           object
TT4 measured                  int64
TT4                          object
T4U measured                  int64
T4U                          object
FTI measured                  int64
FTI                          object
TBG measured                  int64
TBG                         float64
binaryClass                   int64
dtype: object
```

In [24]:
```
cols = df.columns[df.dtypes.eq('object')]
df[cols] = df[cols].apply(pd.to_numeric, errors='coerce')
df.dtypes
```

Out[24]:
```
age                            float64
sex                            float64
on thyroxine                     int64
query on thyroxine               int64
on antithyroid medication        int64
sick                             int64
pregnant                         int64
thyroid surgery                  int64
I131 treatment                   int64
query hypothyroid                int64
query hyperthyroid               int64
lithium                          int64
goitre                           int64
tumor                            int64
hypopituitary                    int64
psych                            int64
TSH measured                     int64
TSH                            float64
T3 measured                      int64
T3                             float64
TT4 measured                     int64
TT4                            float64
T4U measured                     int64
T4U                            float64
FTI measured                     int64
FTI                            float64
TBG measured                     int64
TBG                            float64
binaryClass                      int64
dtype: object
```

In [25]:
```
df.isnull().sum()
```

Out[25]:
```
age                            1
sex                          150
on thyroxine                   0
query on thyroxine             0
on antithyroid medication      0
sick                           0
pregnant                       0
thyroid surgery                0
I131 treatment                 0
query hypothyroid              0
query hyperthyroid             0
lithium                        0
goitre                         0
tumor                          0
hypopituitary                  0
psych                          0
TSH measured                   0
TSH                          369
T3 measured                    0
T3                           769
TT4 measured                   0
TT4                          231
T4U measured                   0
T4U                          387
FTI measured                   0
FTI                          385
TBG measured                   0
TBG                         3772
binaryClass                    0
dtype: int64
```

In [26]:
```
df["TBG"].value_counts()
```

Out[26]:
```
Series([], Name: count, dtype: int64)
```

In [27]:
```
del df["TBG"]
```

In [28]:
```
cols
```

Out[28]:
```
Index(['age', 'TSH', 'T3', 'TT4', 'T4U', 'FTI'], dtype='object')
```

In [29]:
```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')
```

In [30]:
```
df['age'] = imputer.fit_transform(df[['age']])
#
df['T4U measured'] = imputer.fit_transform(df[['T4U measured']])

df['TSH'] = imputer.fit_transform(df[['TSH']])
df['T3'] = imputer.fit_transform(df[['T3']])
df['TT4'] = imputer.fit_transform(df[['TT4']])
df['T4U'] = imputer.fit_transform(df[['T4U']])
df['FTI'] = imputer.fit_transform(df[['FTI']])
```

In [31]:
```
df['sex'].fillna(df['sex'].mode()[0], inplace=True)
```

In [32]:
```
df.isnull().sum().sum()
```

Out[32]:
```
0
```

In [33]:
```
df.to_csv('datafiles/hypothyroid_train.csv',index=False)
```

In [34]:
```
df = pd.read_csv('datafiles/hypothyroid_train.csv')
df.head()
```

Out[34]:

	age	sex	on thyroxine	query on thyroxine	on antithyroid medication	sick	pregnant	thyroid surgery	I131 treatment	hypotl
0	41.0	1.0	0	0	0	0	0	0	0	
1	23.0	1.0	0	0	0	0	0	0	0	
2	46.0	0.0	0	0	0	0	0	0	0	
3	70.0	1.0	1	0	0	0	0	0	0	
4	70.0	1.0	0	0	0	0	0	0	0	

5 rows × 28 columns

In [35]:
```
df.isnull().sum().sum()
```

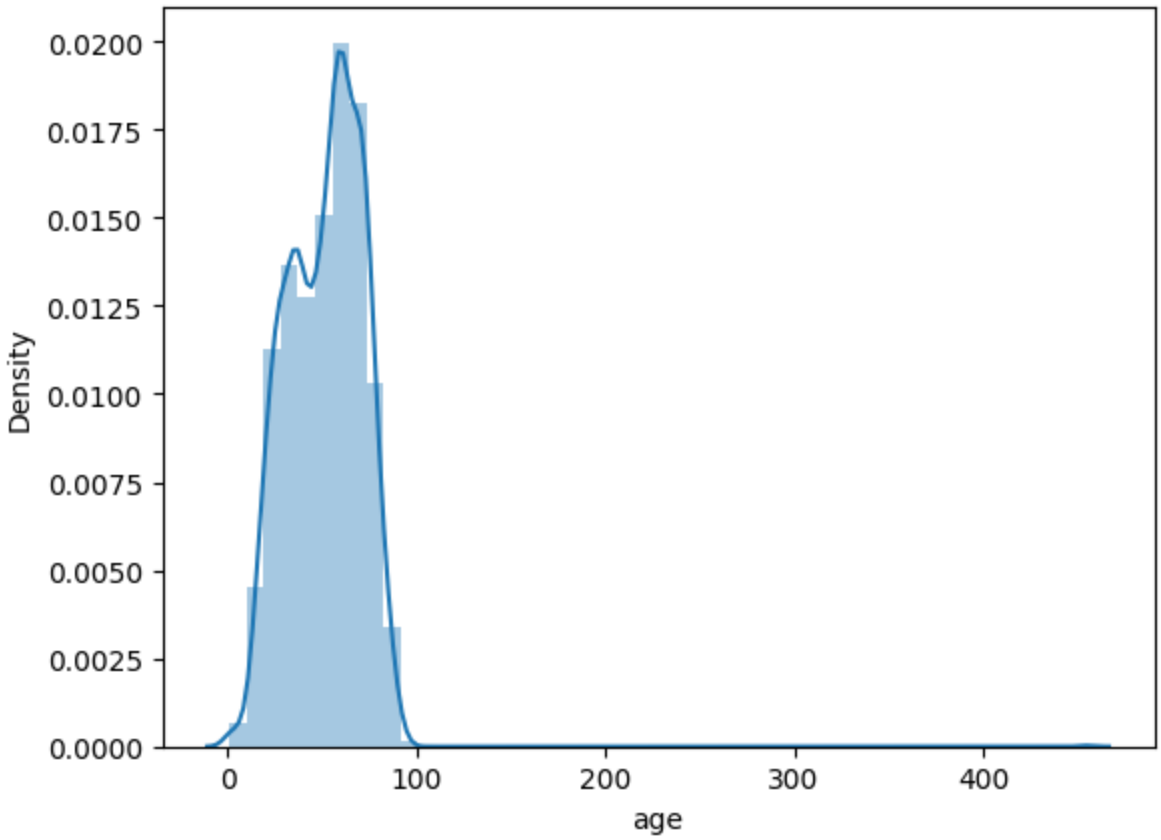
Out[35]:
```
0
```

In [36]:
```
df.columns
```

Out[36]:
```
Index(['age', 'sex', 'on thyroxine', 'query on thyroxine',
       'on antithyroid medication', 'sick', 'pregnant', 'thyroid surgery',
       'I131 treatment', 'query hypothyroid', 'query hyperthyroid', 'lithium',
       'goitre', 'tumor', 'hypopituitary', 'psych', 'TSH measured', 'TSH',
       'T3 measured', 'T3', 'TT4 measured', 'TT4', 'T4U measured', 'T4U',
       'FTI measured', 'FTI', 'TBG measured', 'binaryClass'],
      dtype='object')
```

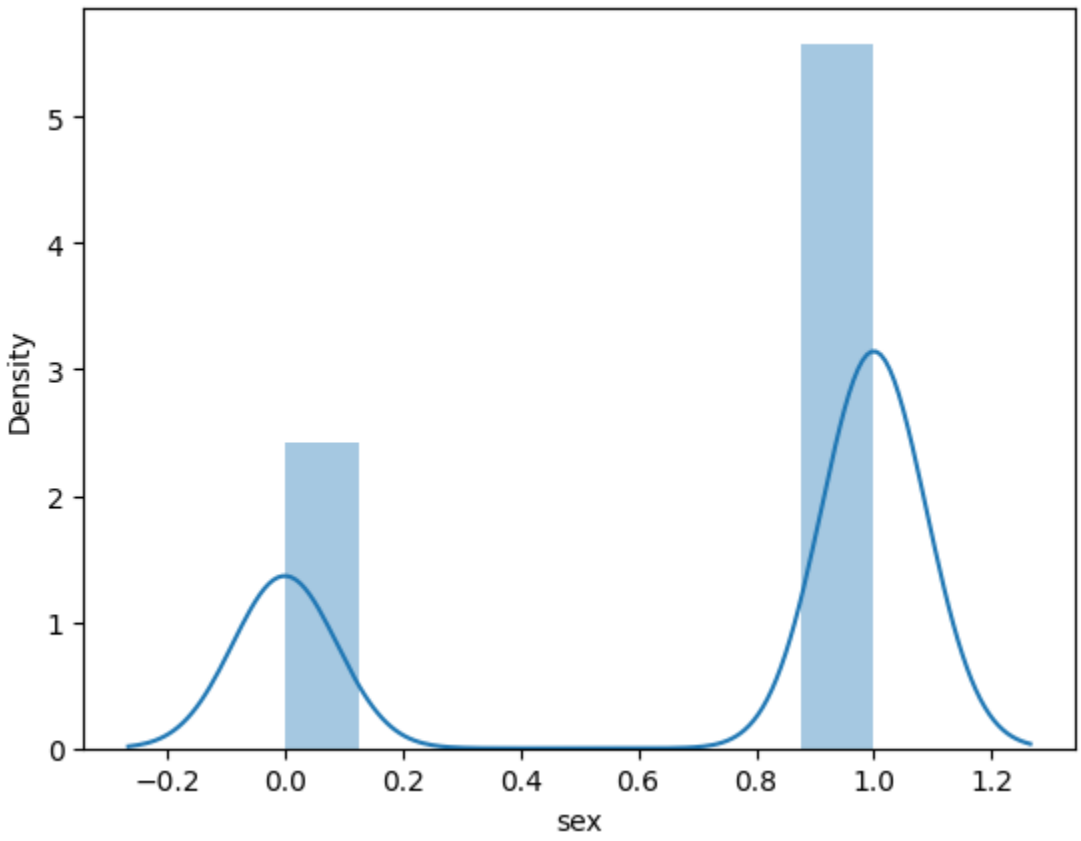
In [37]:
```
sns.distplot(df['age'])
```

Out[37]:
```
<Axes: xlabel='age', ylabel='Density'>
```



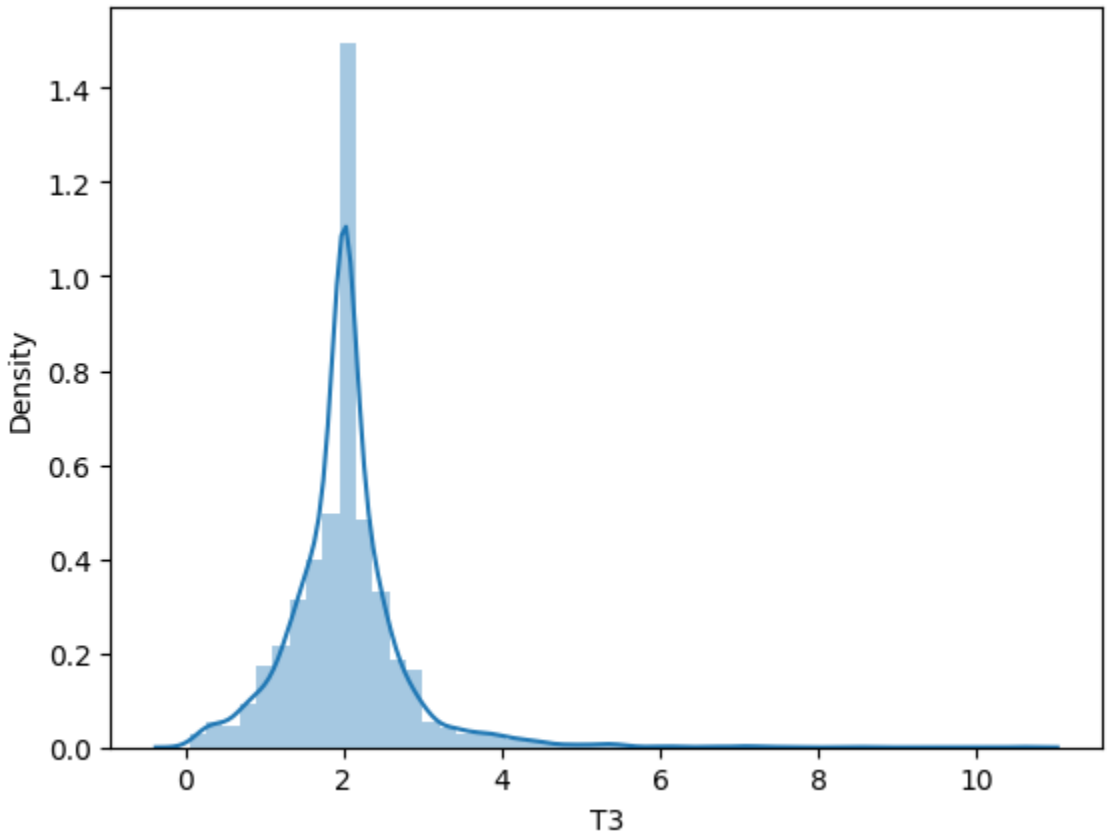
In [38]:
```
sns.distplot(df['sex'])
```

Out[38]: <Axes: xlabel='sex', ylabel='Density'>



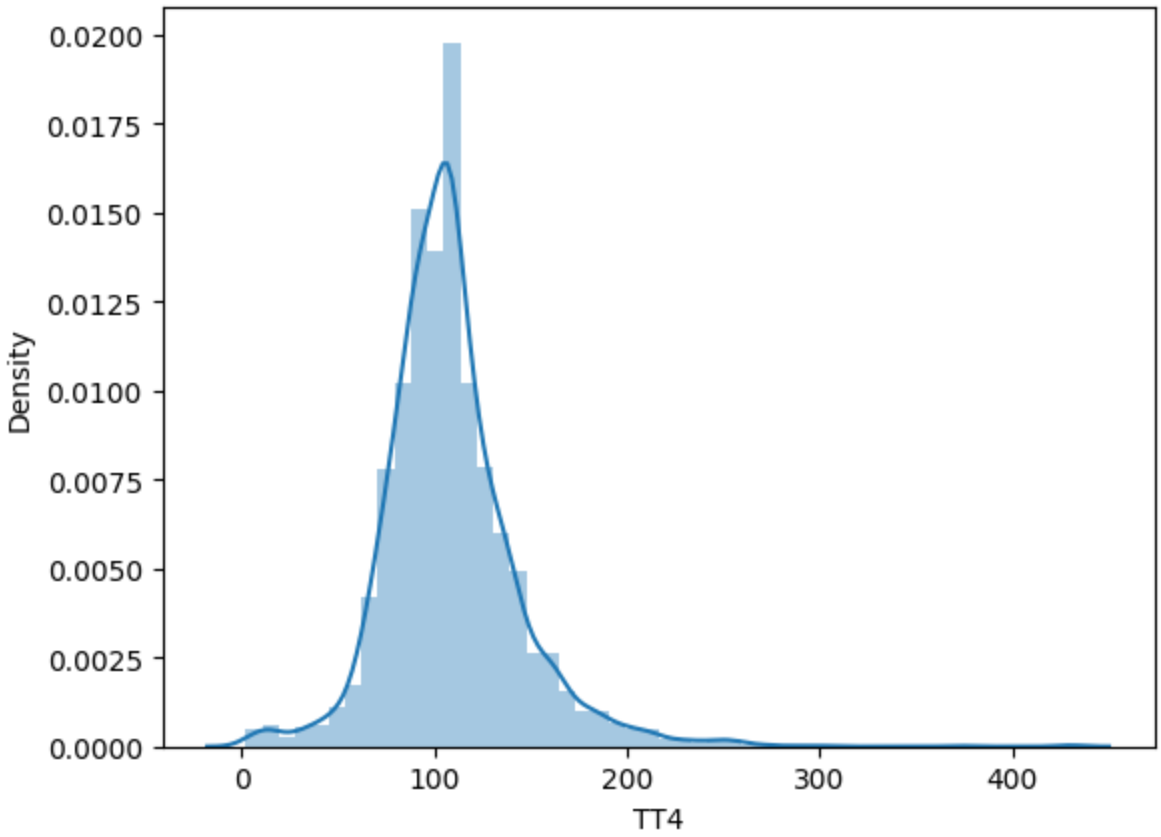
In [39]:
```
sns.distplot(df['T3'])
```

Out[39]: <Axes: xlabel='T3', ylabel='Density'>



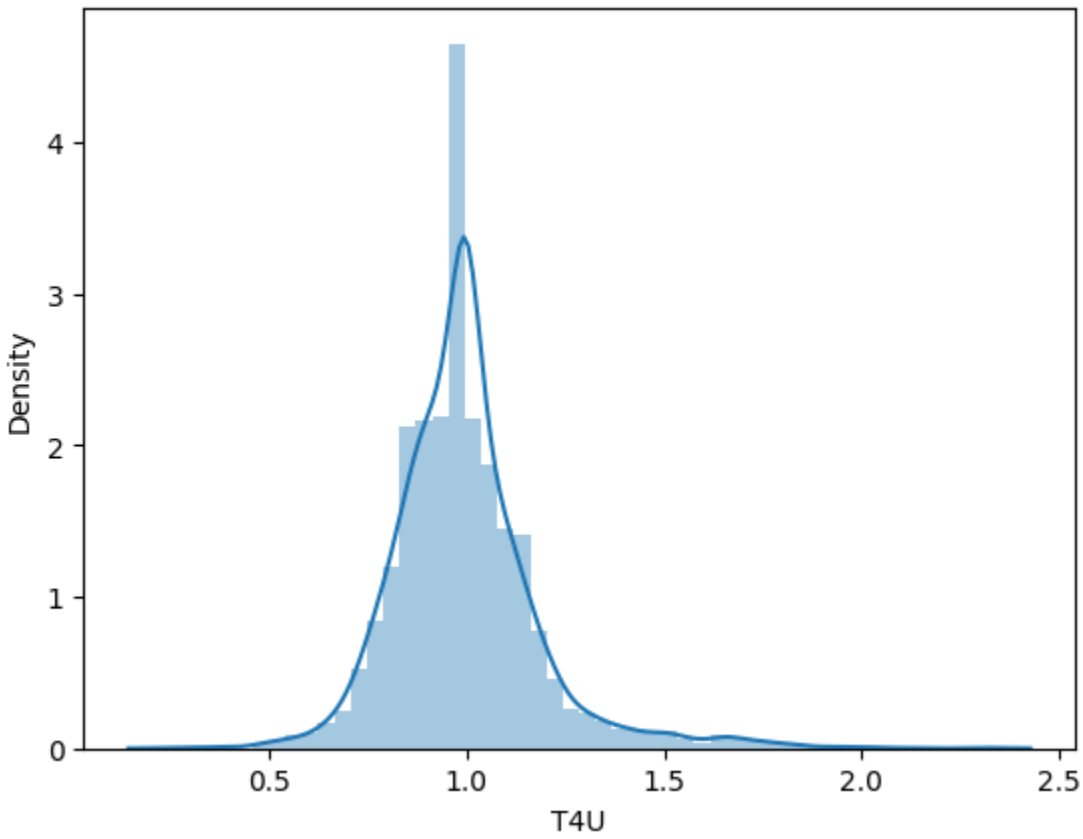
In [40]:
```
sns.distplot(df['TT4'])
```

Out[40]: <Axes: xlabel='TT4', ylabel='Density'>



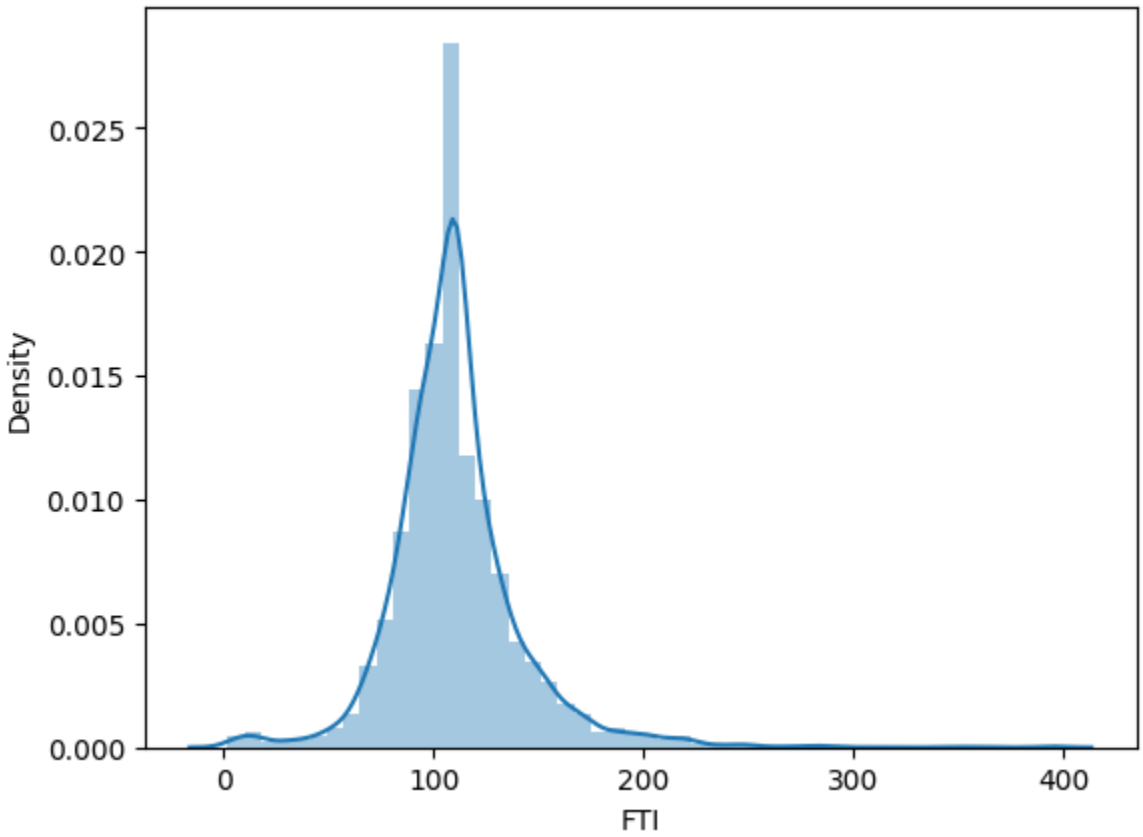
In [41]:
```
sns.distplot(df['T4U'])
```

Out[41]: <Axes: xlabel='T4U', ylabel='Density'>



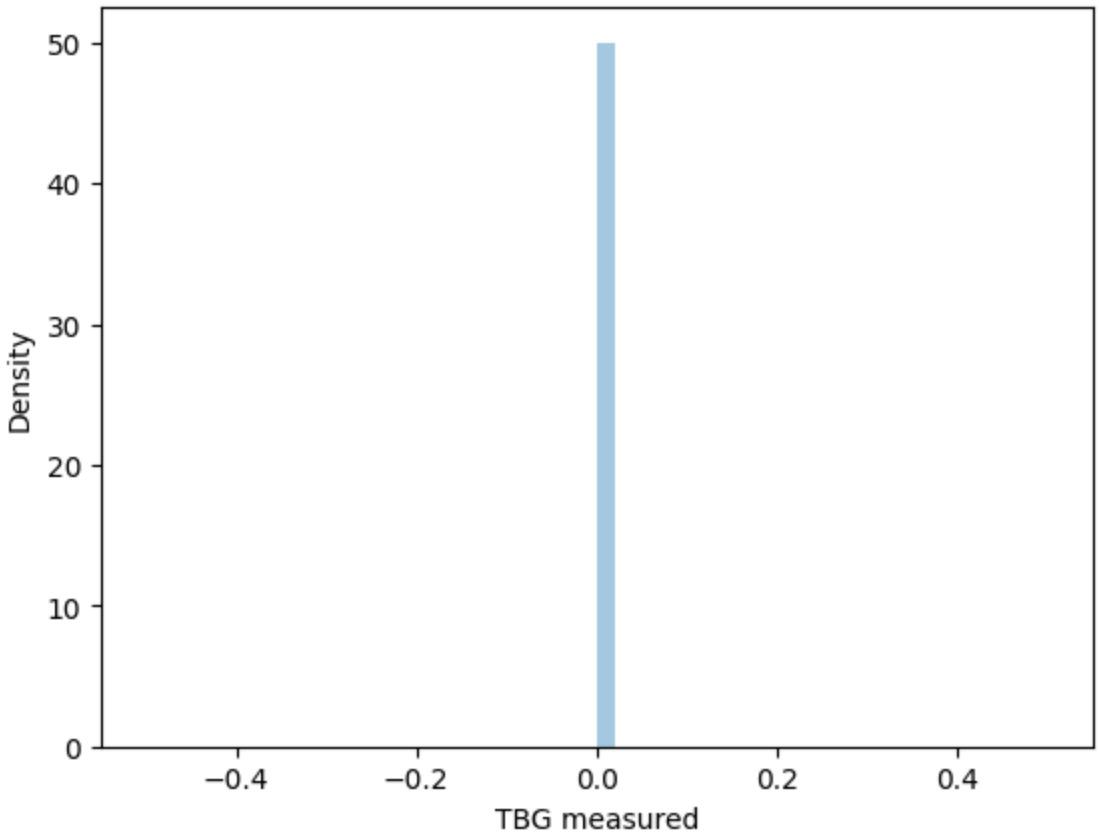
In [42]:
```
sns.distplot(df['FTI'])
```

Out[42]: <Axes: xlabel='FTI', ylabel='Density'>



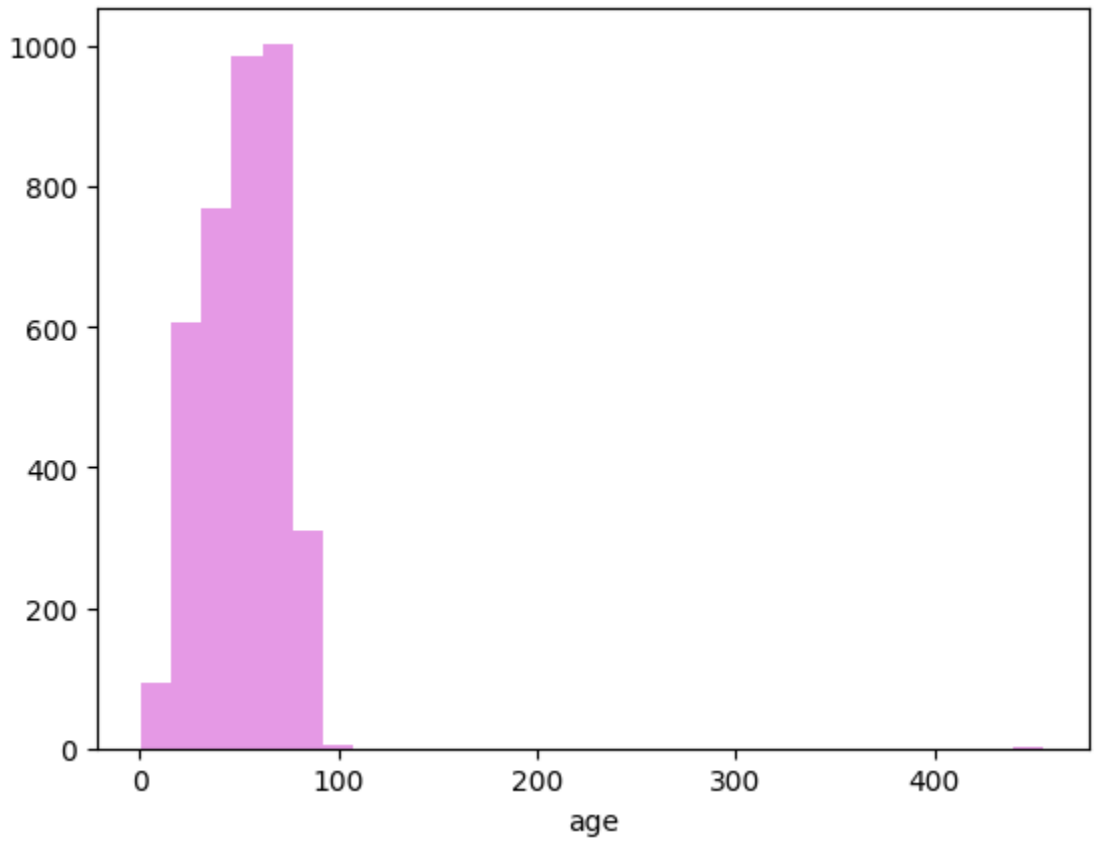
In [43]:
```
sns.distplot(df['TBG measured'])
```

Out[43]: <Axes: xlabel='TBG measured', ylabel='Density'>



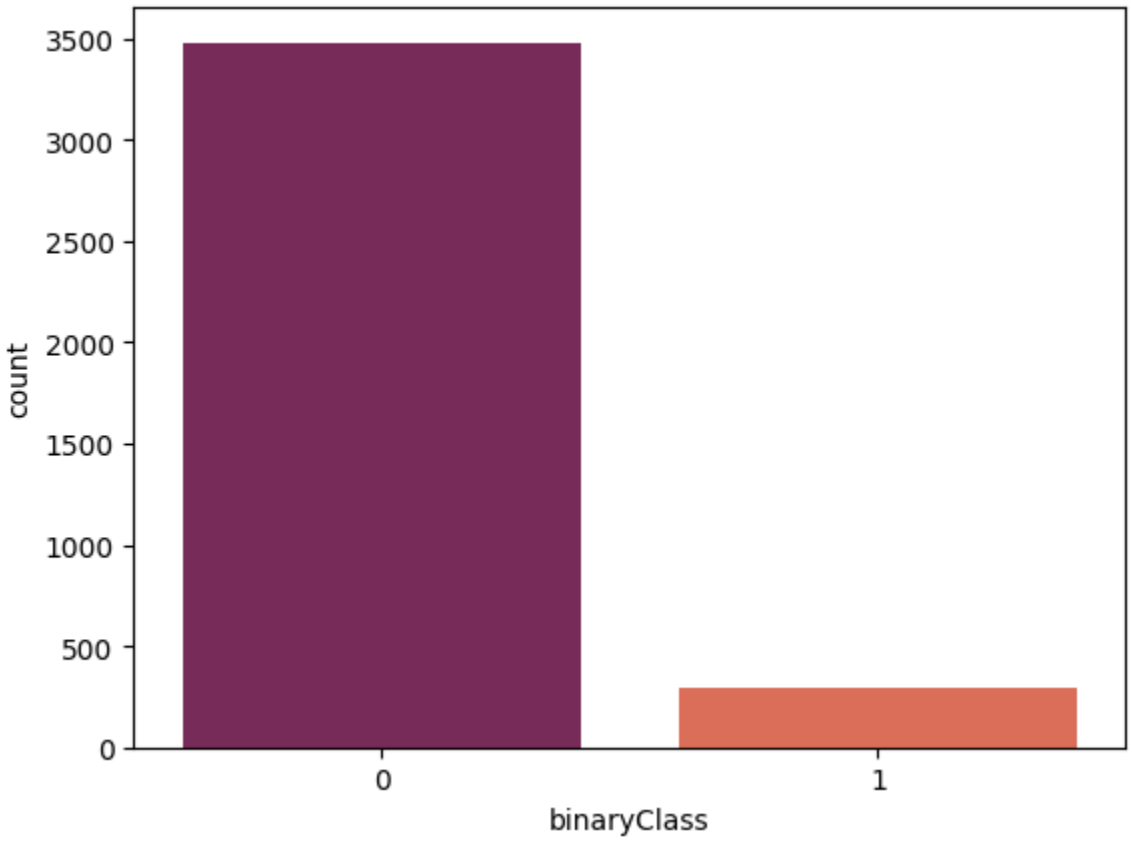
In [44]:
```
sns.distplot(df['age'], kde=False, bins=30, color='m')
```

Out[44]: <Axes: xlabel='age'>



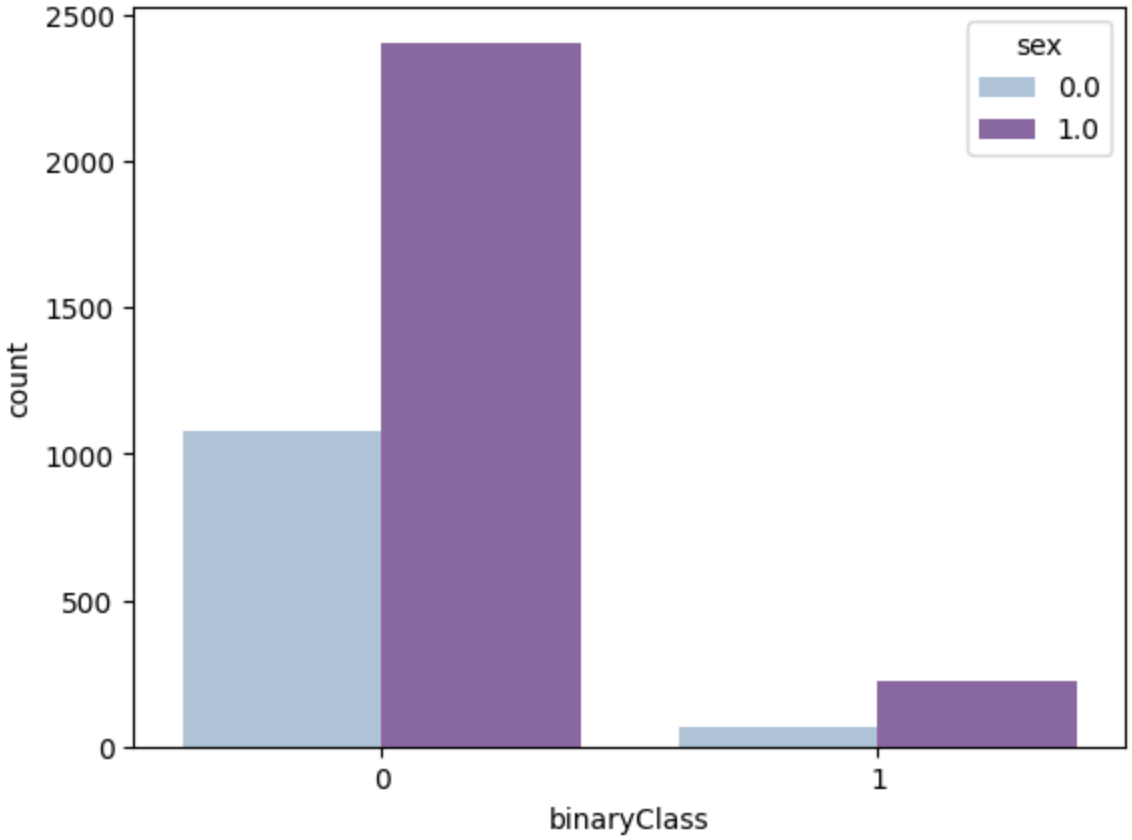
In [45]:
```
sns.countplot(x='binaryClass', data=df, palette='rocket')
```

Out[45]:
```
<Axes: xlabel='binaryClass', ylabel='count'>
```



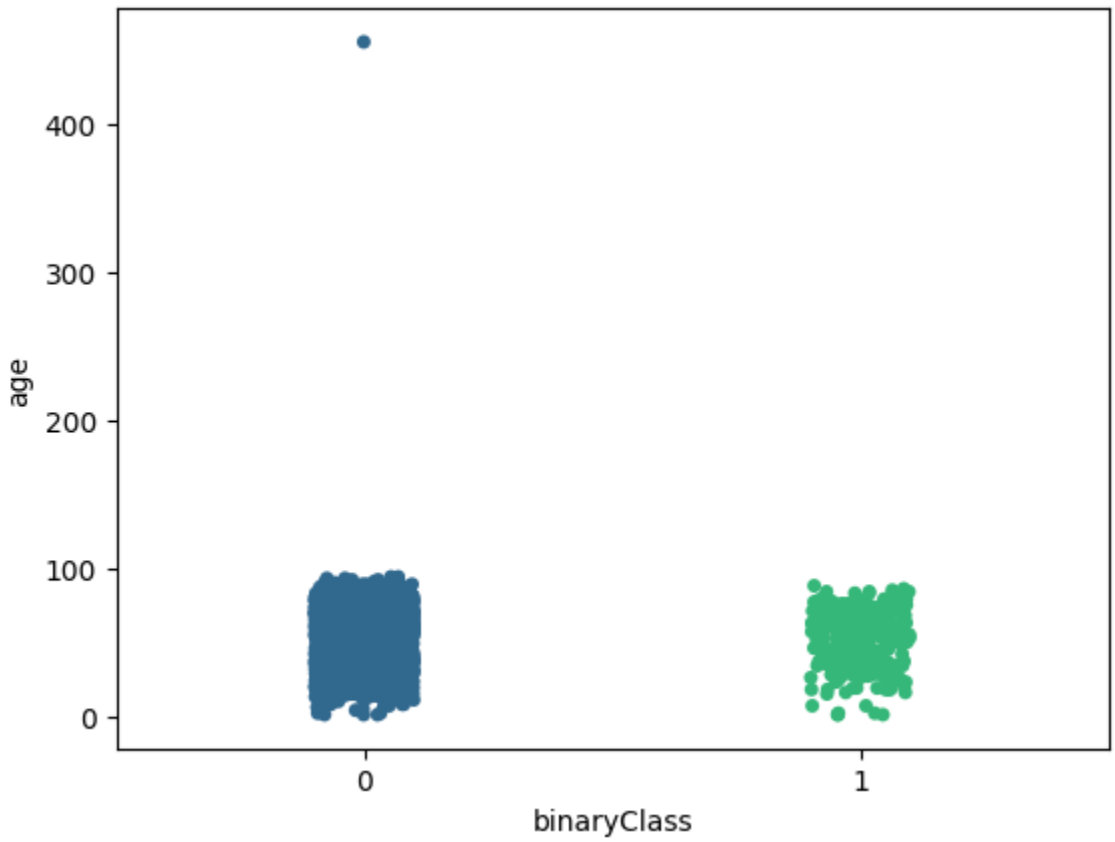
In [46]:
```
sns.countplot(x='binaryClass', data=df, hue='sex', palette='BuPu')
```

Out[46]:
```
<Axes: xlabel='binaryClass', ylabel='count'>
```



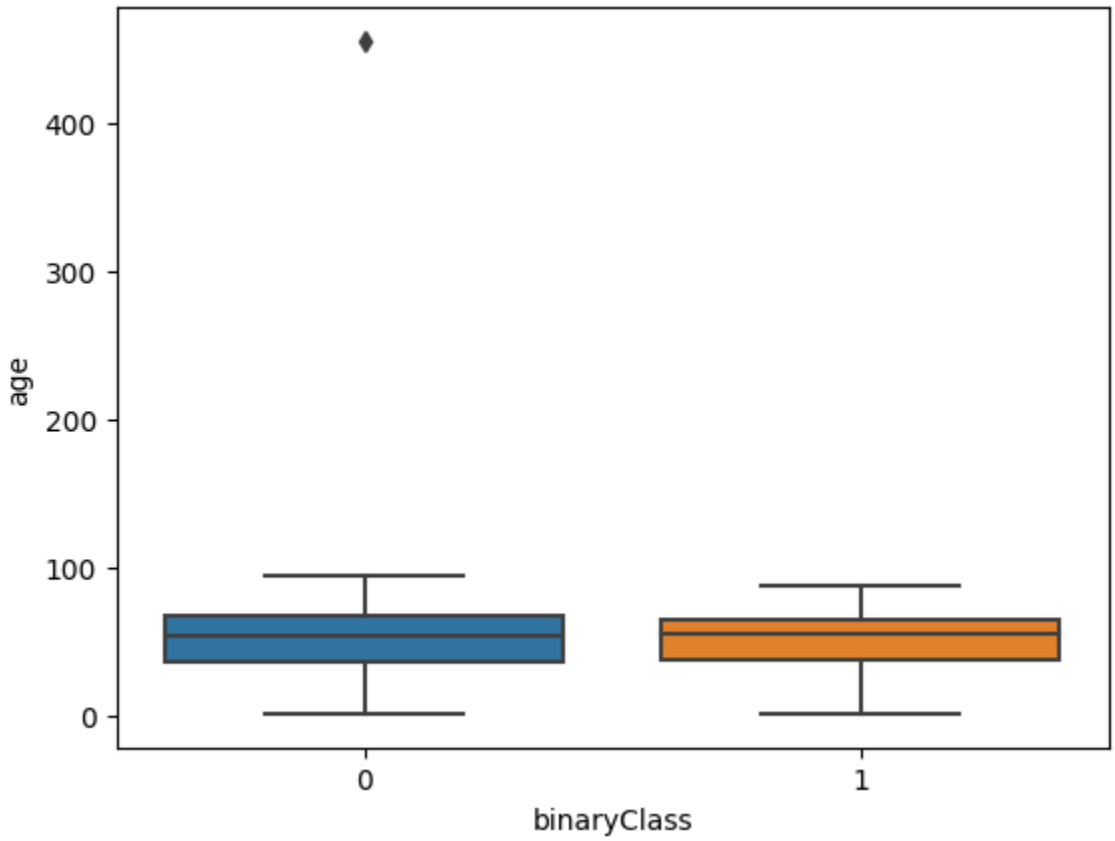
In [47]:
```
sns.stripplot(x="binaryClass", y="age", data=df, palette="viridis")
```

Out[47]:
```
<Axes: xlabel='binaryClass', ylabel='age'>
```



In [48]:
```
sns.boxplot(x='binaryClass', y='age', data=df)
```

Out[48]: <Axes: xlabel='binaryClass', ylabel='age'>



In [49]:
```
df_corr = df.corr()
df_corr
```

Out[49]:

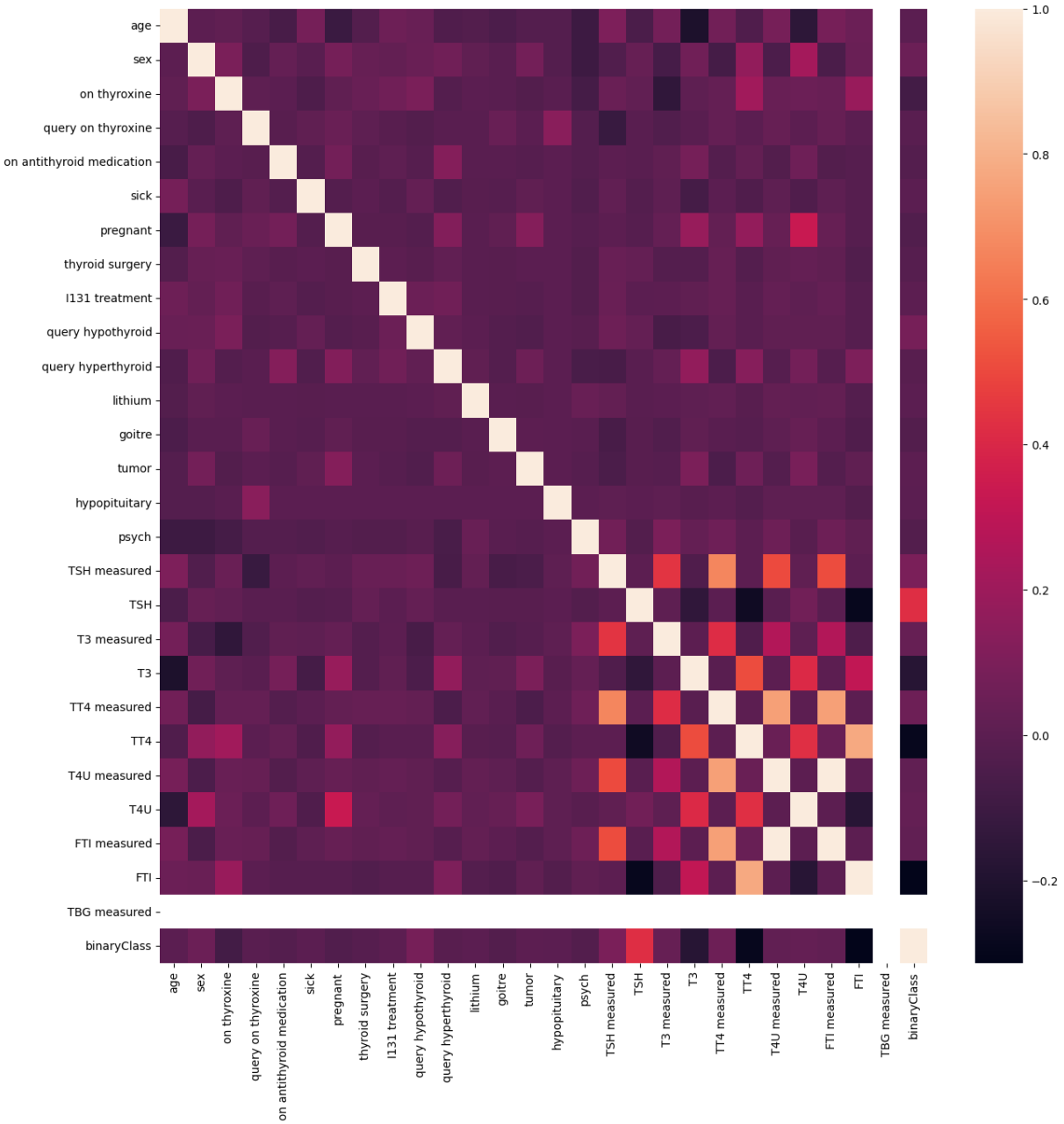
	age	sex	on thyroxine	query on thyroxine	on antithyroid medication	sick	pregnant
age	1.000000	0.000154	0.014563	-0.017870	-0.063881	0.079468	-0.113521
sex	0.000154	1.000000	0.095709	-0.044712	0.027278	-0.007438	0.078665
on thyroxine	0.014563	0.095709	1.000000	0.005995	-0.002201	-0.042053	0.010152
query on thyroxine	-0.017870	-0.044712	0.005995	1.000000	-0.012446	0.012594	0.045247
on antithyroid medication	-0.063881	0.027278	-0.002201	-0.012446	1.000000	-0.021624	0.072050
sick	0.079468	-0.007438	-0.042053	0.012594	-0.021624	1.000000	-0.024040
pregnant	-0.113521	0.078665	0.010152	0.045247	0.072050	-0.024040	1.000000
thyroid surgery	-0.029502	0.034543	0.037583	0.005858	-0.012819	-0.000762	-0.014251
I131 treatment	0.052704	0.022612	0.063373	-0.014610	0.006589	-0.025384	-0.015048
query hypothyroid	0.039562	0.042687	0.094412	-0.029808	-0.017264	0.027718	-0.021364
query hyperthyroid	-0.038054	0.063617	-0.023796	-0.010905	0.126566	-0.035206	0.117605
lithium	-0.030126	0.012138	-0.002509	-0.008026	-0.007436	-0.013944	-0.008266
goitre	-0.051830	-0.010417	-0.010098	0.038000	-0.010241	-0.019205	0.012447
tumor	-0.025037	0.073514	-0.029773	-0.004011	-0.017353	0.010949	0.123728
hypopituitary	-0.024927	-0.024712	-0.006099	0.140500	-0.001749	-0.003279	-0.001944
psych	-0.100116	-0.099896	-0.073571	-0.026247	-0.024318	-0.032883	-0.016577
TSH measured	0.105131	-0.036355	0.041818	-0.117891	0.001736	0.015588	0.001401
TSH	-0.056167	0.033615	0.017138	-0.009453	-0.010668	-0.022099	-0.019693
T3 measured	0.073111	-0.071355	-0.145061	-0.033409	0.010950	0.006695	0.026865
T3	-0.214925	0.064213	0.006485	-0.006466	0.079212	-0.076472	0.181147
TT4 measured	0.067509	-0.076857	0.024964	0.029603	-0.024649	-0.005700	0.021097
TT4	-0.037609	0.167546	0.212801	-0.004702	0.023811	-0.037006	0.172490
T4U measured	0.085361	-0.051662	0.038852	0.031550	-0.029532	0.009399	0.032942

	age	sex	on thyroxine	query on thyroxine	on antithyroid medication	sick	pregnant
T4U	-0.157523	0.221883	0.046368	0.000438	0.060365	-0.039069	0.334702
FTI measured	0.084534	-0.052532	0.038285	0.031420	-0.029788	0.009068	0.032808
FTI	0.050017	0.041900	0.185748	-0.003550	-0.016603	-0.021189	-0.016698
TBG measured	NaN	NaN	NaN	NaN	NaN	NaN	NaN
binaryClass	-0.003174	0.049960	-0.081060	-0.007448	-0.021689	-0.001749	-0.034516

28 rows × 28 columns

In [50]:
```
plt.figure(figsize = (15,15))
sns.heatmap(df_corr)
```

Out[50]: <Axes: >



In [51]:
```
x = df.drop('binaryClass', axis=1)
y = df['binaryClass']
```

In [52]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_st
```

In [53]:
```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
sc.fit(x_train)
x_train = sc.transform(x_train)
x_test = sc.transform(x_test)
```

In [54]:
```
x_train[0]
```

Out[54]:
```
array([-1.25724849,  0.66254235, -0.37144669, -0.11823068, -0.10696181,
       -0.20208625, -0.11668415, -0.11353134, -0.12569377, -0.25912251,
       -0.26298479, -0.07052203, -0.09053575, -0.16275031, -0.01880444,
       -0.22472494,  0.32615128, -0.12074446,  0.49790031, -0.01562478,
        0.25126512,  0.75997109,  0.33796183,  1.19009127,  0.33666185,
        0.01223448,  0.        ])
```

In [55]:
```
x_test[0]
```

Out[55]:
```
array([-1.25724849,  0.66254235, -0.37144669, -0.11823068, -0.10696181,
       -0.20208625, -0.11668415, -0.11353134, -0.12569377, -0.25912251,
       -0.26298479, -0.07052203, -0.09053575, -0.16275031, -0.01880444,
       -0.22472494,  0.32615128, -0.12908581,  0.49790031,  0.25156559,
        0.25126512, -0.38131003,  0.33796183,  0.03573321,  0.33666185,
       -0.49086425,  0.        ])
```

In [56]:
```
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.model_selection import cross_val_score, KFold
```

In [57]:
```
model = LogisticRegression()
model.fit(x_train, y_train)
```

Out[57]:
```
▾ LogisticRegression

LogisticRegression()
```

In [58]:
```
predicted = model.predict(x_test)
```

In [59]:
```
metrics.accuracy_score(y_test,predicted)
```

Out[59]:
```
0.9692470837751855
```

In [60]:
```
metrics.confusion_matrix(y_test,predicted)
```

Out[60]:
```
array([[877,   1],
       [ 28,  37]])
```

In [61]:
```
metrics.classification_report(y_test, predicted)
```

Out[61]:
```
'              precision    recall  f1-score   support\n\n           0       0.97
1.00      0.98       878\n           1       0.97      0.57      0.72        65\n\n
accuracy                           0.97       943\n   macro avg       0.97      0.7
8      0.85       943\nweighted avg       0.97      0.97      0.97       943\n'
```

In [62]:
```
print(metrics.classification_report(y_test, predicted))
```

	precision	recall	f1-score	support
0	0.97	1.00	0.98	878
1	0.97	0.57	0.72	65
accuracy			0.97	943
macro avg	0.97	0.78	0.85	943
weighted avg	0.97	0.97	0.97	943

In [63]:
```
scores = cross_val_score(LogisticRegression(), x_train, y_train, scoring='accuracy'
scores, scores.mean()
```

Out[63]:
```
(array([0.96466431, 0.9540636 , 0.96113074, 0.96113074, 0.96466431,
        0.96819788, 0.95759717, 0.97173145, 0.95053004, 0.96453901]),
 0.9618249254442022)
```

In [64]:
```
from sklearn.model_selection import GridSearchCV
```

In [65]:
```
parameters = [{'penalty':['l1','l2']},
              {'C':[1, 10, 100, 1000]}]
grid_search = GridSearchCV(LogisticRegression(),
                           param_grid = parameters,
                           scoring = 'accuracy',
                           cv = 5,
                           verbose=0)

best_model = grid_search.fit(x_train, y_train)
```

In [66]:
```
best_model.best_params_
```

Out[66]:
```
{'C': 100}
```

In [67]:
```
model2 = LogisticRegression(C=100)
model2.fit(x_train, y_train)
```

Out[67]:
```
▾  LogisticRegression

LogisticRegression(C=100)
```

In [68]:
```
predicted = model.predict(x_test)
```

In [69]:
```
metrics.accuracy_score(y_test,predicted)
```

Out[69]:
```
0.9692470837751855
```

In [70]:
```
from imblearn.over_sampling import SMOTE

sm = SMOTE(random_state = 2)
x_train_res, y_train_res = sm.fit_resample(x_train, y_train.ravel())
```

In [71]:
```
sm = SMOTE(random_state = 2)
x_test_res, y_test_res = sm.fit_resample(x_test, y_test.ravel())
```

In [72]:
```
x_train_res.shape, x_train.shape
```

Out[72]:
```
((5206, 27), (2829, 27))
```

In [73]:
```
y_train_res.shape, y_train.shape
```

Out[73]:
```
((5206,), (2829,))
```

In [74]:
```
zeros = 0
ones = 0
for item in y_train_res:
    if item == 1:
        ones += 1
    else:
        zeros += 1

print('zeros:',zeros)
print('ones:',ones)
```

```
zeros: 2603
ones: 2603
```

In [75]:
```
lr = LogisticRegression()
model_res = lr.fit(x_train_res, y_train_res)
```

In [76]:
```
scores = cross_val_score(lr, x_train_res, y_train_res, scoring='accuracy', cv=10)
scores, scores.mean()
```

Out[76]:
```
(array([0.97504798, 0.99232246, 0.98848369, 0.98272553, 0.99424184,
        0.9865643 , 0.98653846, 0.98846154, 0.98076923, 0.98269231]),
 0.9857847335006642)
```

In [77]:
```
predicted_res = model.predict(x_test_res)
```

In [78]:
```
metrics.accuracy_score(y_test_res,predicted_res)
```

Out[78]:
```
0.8080865603644647
```

In [79]:
```
params = [{'learning_rate':[0.01,0.001],
           'max_depth': [3,5,10],
           'n_estimators':[10,50,100,200]}]
```

In [80]:
```
from sklearn.model_selection import GridSearchCV
from xgboost import XGBClassifier
Xbc = XGBClassifier()
Gcv = GridSearchCV(Xbc,params,scoring='accuracy',cv=5,n_jobs=3,verbose=3)
Gcv.fit(x_train_res,y_train_res)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
[CV 1/5] END learning_rate=0.01, max_depth=3, n_estimators=10;, score=0.991 total ti
me=   0.0s
[CV 2/5] END learning_rate=0.01, max_depth=3, n_estimators=10;, score=0.994 total ti
me=   0.0s
[CV 4/5] END learning_rate=0.01, max_depth=3, n_estimators=10;, score=0.994 total ti
me=   0.0s
[CV 3/5] END learning_rate=0.01, max_depth=3, n_estimators=10;, score=0.996 total ti
me=   0.0s
[CV 5/5] END learning_rate=0.01, max_depth=3, n_estimators=10;, score=0.995 total ti
me=   0.0s
[CV 1/5] END learning_rate=0.01, max_depth=3, n_estimators=50;, score=0.993 total ti
me=   0.1s
[CV 2/5] END learning_rate=0.01, max_depth=3, n_estimators=50;, score=0.996 total ti
me=   0.1s
[CV 3/5] END learning_rate=0.01, max_depth=3, n_estimators=50;, score=0.996 total ti
me=   0.1s
[CV 4/5] END learning_rate=0.01, max_depth=3, n_estimators=50;, score=0.994 total ti
me=   0.1s
[CV 5/5] END learning_rate=0.01, max_depth=3, n_estimators=50;, score=0.995 total ti
me=   0.1s
[CV 1/5] END learning_rate=0.01, max_depth=3, n_estimators=100;, score=0.993 total t
ime=   0.2s
[CV 2/5] END learning_rate=0.01, max_depth=3, n_estimators=100;, score=0.996 total t
ime=   0.2s
[CV 3/5] END learning_rate=0.01, max_depth=3, n_estimators=100;, score=0.996 total t
ime=   0.2s
[CV 4/5] END learning_rate=0.01, max_depth=3, n_estimators=100;, score=0.995 total t
ime=   0.2s
[CV 5/5] END learning_rate=0.01, max_depth=3, n_estimators=100;, score=0.995 total t
ime=   0.2s
[CV 1/5] END learning_rate=0.01, max_depth=3, n_estimators=200;, score=0.995 total t
ime=   0.3s
[CV 2/5] END learning_rate=0.01, max_depth=3, n_estimators=200;, score=0.996 total t
ime=   0.3s
[CV 3/5] END learning_rate=0.01, max_depth=3, n_estimators=200;, score=0.997 total t
ime=   0.3s
[CV 1/5] END learning_rate=0.01, max_depth=5, n_estimators=10;, score=0.993 total ti
me=   0.0s
[CV 2/5] END learning_rate=0.01, max_depth=5, n_estimators=10;, score=0.997 total ti
me=   0.0s
[CV 3/5] END learning_rate=0.01, max_depth=5, n_estimators=10;, score=0.997 total ti
me=   0.1s
[CV 4/5] END learning_rate=0.01, max_depth=3, n_estimators=200;, score=0.997 total t
ime=   0.4s
[CV 4/5] END learning_rate=0.01, max_depth=5, n_estimators=10;, score=0.997 total ti
me=   0.0s
[CV 5/5] END learning_rate=0.01, max_depth=5, n_estimators=10;, score=0.997 total ti
me=   0.0s
[CV 5/5] END learning_rate=0.01, max_depth=3, n_estimators=200;, score=0.996 total t
ime=   0.3s
[CV 1/5] END learning_rate=0.01, max_depth=5, n_estimators=50;, score=0.995 total ti
me=   0.1s
[CV 2/5] END learning_rate=0.01, max_depth=5, n_estimators=50;, score=0.997 total ti
me=   0.1s
[CV 3/5] END learning_rate=0.01, max_depth=5, n_estimators=50;, score=0.997 total ti
```

```
me=   0.1s
[CV 4/5] END learning_rate=0.01, max_depth=5, n_estimators=50;, score=0.997 total ti
me=   0.1s
[CV 5/5] END learning_rate=0.01, max_depth=5, n_estimators=50;, score=0.993 total ti
me=   0.1s
[CV 2/5] END learning_rate=0.01, max_depth=5, n_estimators=100;, score=0.998 total t
ime=   0.2s
[CV 3/5] END learning_rate=0.01, max_depth=5, n_estimators=100;, score=0.997 total t
ime=   0.2s
[CV 1/5] END learning_rate=0.01, max_depth=5, n_estimators=100;, score=0.995 total t
ime=   0.4s
[CV 4/5] END learning_rate=0.01, max_depth=5, n_estimators=100;, score=0.997 total t
ime=   0.2s
[CV 5/5] END learning_rate=0.01, max_depth=5, n_estimators=100;, score=0.993 total t
ime=   0.2s
[CV 2/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.999 total t
ime=   0.4s
[CV 1/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.998 total t
ime=   0.6s
[CV 3/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.997 total t
ime=   0.6s
[CV 1/5] END learning_rate=0.01, max_depth=10, n_estimators=10;, score=0.993 total t
ime=   0.0s
[CV 2/5] END learning_rate=0.01, max_depth=10, n_estimators=10;, score=0.997 total t
ime=   0.0s
[CV 4/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.999 total t
ime=   0.5s
[CV 3/5] END learning_rate=0.01, max_depth=10, n_estimators=10;, score=0.997 total t
ime=   0.2s
[CV 5/5] END learning_rate=0.01, max_depth=10, n_estimators=10;, score=0.997 total t
ime=   0.0s
[CV 4/5] END learning_rate=0.01, max_depth=10, n_estimators=10;, score=0.997 total t
ime=   0.1s
[CV 5/5] END learning_rate=0.01, max_depth=5, n_estimators=200;, score=0.993 total t
ime=   0.5s
[CV 1/5] END learning_rate=0.01, max_depth=10, n_estimators=50;, score=0.995 total t
ime=   0.1s
[CV 2/5] END learning_rate=0.01, max_depth=10, n_estimators=50;, score=0.997 total t
ime=   0.2s
[CV 3/5] END learning_rate=0.01, max_depth=10, n_estimators=50;, score=0.997 total t
ime=   0.2s
[CV 4/5] END learning_rate=0.01, max_depth=10, n_estimators=50;, score=0.997 total t
ime=   0.2s
[CV 5/5] END learning_rate=0.01, max_depth=10, n_estimators=50;, score=0.993 total t
ime=   0.2s
[CV 1/5] END learning_rate=0.01, max_depth=10, n_estimators=100;, score=0.995 total
time=   0.3s
[CV 2/5] END learning_rate=0.01, max_depth=10, n_estimators=100;, score=0.998 total
time=   0.3s
[CV 3/5] END learning_rate=0.01, max_depth=10, n_estimators=100;, score=0.997 total
time=   0.3s
[CV 4/5] END learning_rate=0.01, max_depth=10, n_estimators=100;, score=0.997 total
time=   0.2s
[CV 5/5] END learning_rate=0.01, max_depth=10, n_estimators=100;, score=0.993 total
time=   0.2s
[CV 1/5] END learning_rate=0.01, max_depth=10, n_estimators=200;, score=0.998 total
```

```
time=   0.5s
[CV 2/5] END learning_rate=0.01, max_depth=10, n_estimators=200;, score=0.999 total
time=   0.6s
[CV 3/5] END learning_rate=0.01, max_depth=10, n_estimators=200;, score=0.997 total
time=   0.6s
[CV 1/5] END learning_rate=0.001, max_depth=3, n_estimators=10;, score=0.991 total t
ime=   0.0s
[CV 2/5] END learning_rate=0.001, max_depth=3, n_estimators=10;, score=0.994 total t
ime=   0.1s
[CV 3/5] END learning_rate=0.001, max_depth=3, n_estimators=10;, score=0.996 total t
ime=   0.0s
[CV 4/5] END learning_rate=0.001, max_depth=3, n_estimators=10;, score=0.994 total t
ime=   0.0s
[CV 5/5] END learning_rate=0.001, max_depth=3, n_estimators=10;, score=0.995 total t
ime=   0.0s
[CV 1/5] END learning_rate=0.001, max_depth=3, n_estimators=50;, score=0.991 total t
ime=   0.1s
[CV 4/5] END learning_rate=0.01, max_depth=10, n_estimators=200;, score=0.999 total
time=   0.5s
[CV 2/5] END learning_rate=0.001, max_depth=3, n_estimators=50;, score=0.994 total t
ime=   0.1s
[CV 3/5] END learning_rate=0.001, max_depth=3, n_estimators=50;, score=0.996 total t
ime=   0.1s
[CV 4/5] END learning_rate=0.001, max_depth=3, n_estimators=50;, score=0.994 total t
ime=   0.1s
[CV 5/5] END learning_rate=0.01, max_depth=10, n_estimators=200;, score=0.993 total
time=   0.5s
[CV 5/5] END learning_rate=0.001, max_depth=3, n_estimators=50;, score=0.995 total t
ime=   0.1s
[CV 1/5] END learning_rate=0.001, max_depth=3, n_estimators=100;, score=0.991 total
time=   0.2s
[CV 2/5] END learning_rate=0.001, max_depth=3, n_estimators=100;, score=0.994 total
time=   0.2s
[CV 3/5] END learning_rate=0.001, max_depth=3, n_estimators=100;, score=0.996 total
time=   0.2s
[CV 5/5] END learning_rate=0.001, max_depth=3, n_estimators=100;, score=0.995 total
time=   0.2s
[CV 4/5] END learning_rate=0.001, max_depth=3, n_estimators=100;, score=0.994 total
time=   0.2s
[CV 1/5] END learning_rate=0.001, max_depth=3, n_estimators=200;, score=0.991 total
time=   0.3s
[CV 2/5] END learning_rate=0.001, max_depth=3, n_estimators=200;, score=0.994 total
time=   0.3s
[CV 3/5] END learning_rate=0.001, max_depth=3, n_estimators=200;, score=0.996 total
time=   0.3s
[CV 1/5] END learning_rate=0.001, max_depth=5, n_estimators=10;, score=0.993 total t
ime=   0.0s
[CV 2/5] END learning_rate=0.001, max_depth=5, n_estimators=10;, score=0.997 total t
ime=   0.0s
[CV 3/5] END learning_rate=0.001, max_depth=5, n_estimators=10;, score=0.997 total t
ime=   0.0s
[CV 4/5] END learning_rate=0.001, max_depth=3, n_estimators=200;, score=0.994 total
time=   0.3s
[CV 4/5] END learning_rate=0.001, max_depth=5, n_estimators=10;, score=0.997 total t
ime=   0.0s
[CV 5/5] END learning_rate=0.001, max_depth=5, n_estimators=10;, score=0.997 total t
```

```
ime=   0.0s
[CV 5/5] END learning_rate=0.001, max_depth=3, n_estimators=200;, score=0.995 total
time=   0.3s
[CV 1/5] END learning_rate=0.001, max_depth=5, n_estimators=50;, score=0.993 total t
ime=   0.1s
[CV 2/5] END learning_rate=0.001, max_depth=5, n_estimators=50;, score=0.997 total t
ime=   0.1s
[CV 3/5] END learning_rate=0.001, max_depth=5, n_estimators=50;, score=0.997 total t
ime=   0.1s
[CV 4/5] END learning_rate=0.001, max_depth=5, n_estimators=50;, score=0.997 total t
ime=   0.1s
[CV 5/5] END learning_rate=0.001, max_depth=5, n_estimators=50;, score=0.997 total t
ime=   0.2s
[CV 1/5] END learning_rate=0.001, max_depth=5, n_estimators=100;, score=0.993 total
time=   0.2s
[CV 4/5] END learning_rate=0.001, max_depth=5, n_estimators=100;, score=0.997 total
time=   0.2s
[CV 3/5] END learning_rate=0.001, max_depth=5, n_estimators=100;, score=0.997 total
time=   0.4s
[CV 2/5] END learning_rate=0.001, max_depth=5, n_estimators=100;, score=0.997 total
time=   0.4s
[CV 5/5] END learning_rate=0.001, max_depth=5, n_estimators=100;, score=0.996 total
time=   0.2s
[CV 1/5] END learning_rate=0.001, max_depth=5, n_estimators=200;, score=0.993 total
time=   0.5s
[CV 2/5] END learning_rate=0.001, max_depth=5, n_estimators=200;, score=0.997 total
time=   0.5s
[CV 3/5] END learning_rate=0.001, max_depth=5, n_estimators=200;, score=0.997 total
time=   0.5s
[CV 1/5] END learning_rate=0.001, max_depth=10, n_estimators=10;, score=0.993 total
time=   0.1s
[CV 4/5] END learning_rate=0.001, max_depth=5, n_estimators=200;, score=0.997 total
time=   0.4s
[CV 2/5] END learning_rate=0.001, max_depth=10, n_estimators=10;, score=0.997 total
time=   0.2s
[CV 3/5] END learning_rate=0.001, max_depth=10, n_estimators=10;, score=0.997 total
time=   0.0s
[CV 4/5] END learning_rate=0.001, max_depth=10, n_estimators=10;, score=0.997 total
time=   0.0s
[CV 5/5] END learning_rate=0.001, max_depth=10, n_estimators=10;, score=0.997 total
time=   0.0s
[CV 5/5] END learning_rate=0.001, max_depth=5, n_estimators=200;, score=0.993 total
time=   0.5s
[CV 1/5] END learning_rate=0.001, max_depth=10, n_estimators=50;, score=0.993 total
time=   0.1s
[CV 2/5] END learning_rate=0.001, max_depth=10, n_estimators=50;, score=0.997 total
time=   0.1s
[CV 3/5] END learning_rate=0.001, max_depth=10, n_estimators=50;, score=0.997 total
time=   0.1s
[CV 4/5] END learning_rate=0.001, max_depth=10, n_estimators=50;, score=0.997 total
time=   0.1s
[CV 5/5] END learning_rate=0.001, max_depth=10, n_estimators=50;, score=0.997 total
time=   0.1s
[CV 1/5] END learning_rate=0.001, max_depth=10, n_estimators=100;, score=0.993 total
time=   0.2s
[CV 2/5] END learning_rate=0.001, max_depth=10, n_estimators=100;, score=0.997 total
```

```
time=   0.2s
[CV 3/5] END learning_rate=0.001, max_depth=10, n_estimators=100;, score=0.997 total
time=   0.3s
[CV 4/5] END learning_rate=0.001, max_depth=10, n_estimators=100;, score=0.997 total
time=   0.2s
[CV 5/5] END learning_rate=0.001, max_depth=10, n_estimators=100;, score=0.996 total
time=   0.2s
[CV 1/5] END learning_rate=0.001, max_depth=10, n_estimators=200;, score=0.993 total
time=   0.4s
[CV 3/5] END learning_rate=0.001, max_depth=10, n_estimators=200;, score=0.997 total
time=   0.5s
[CV 2/5] END learning_rate=0.001, max_depth=10, n_estimators=200;, score=0.997 total
time=   0.6s
[CV 4/5] END learning_rate=0.001, max_depth=10, n_estimators=200;, score=0.997 total
time=   0.5s
[CV 5/5] END learning_rate=0.001, max_depth=10, n_estimators=200;, score=0.993 total
time=   0.4s
```

Out[80]:

▸ **GridSearchCV**

▸ **estimator: XGBClassifier**

▸ XGBClassifier

In [81]:
```
Gcv.best_params_
```

Out[81]:
```
{'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 200}
```

In [82]:
```
XBC = XGBClassifier(learning_rate=0.01,max_depth=5,n_estimators=200)
XBC.fit(x_train_res,y_train_res)
```

Out[82]:

▾ XGBClassifier

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=No
ne,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=N
one,
              interaction_constraints=None, learning_rate=0.01, max_bin=No
ne,
```

In [83]:
```
XBC.score(x_test_res,y_test_res)
```

Out[83]: 0.9988610478359908

In [84]:
```
y_pred = XBC.predict(x_test_res)
```

In [85]:
```
metrics.accuracy_score(y_test_res,y_pred)
```

Out[85]: 0.9988610478359908

In [86]:
```
from sklearn.ensemble import RandomForestRegressor
rfr = RandomForestRegressor()
rfr.fit(x_train,y_train)
```

Out[86]:
```
▾ RandomForestRegressor

RandomForestRegressor()
```

In [87]:
```
print(f"Train_model_score: {rfr.score(x_train,y_train)}")
print(f"Test_model_score: {rfr.score(x_test,y_test)}")
```

```
Train_model_score: 0.9928308160427552
Test_model_score: 0.984167117574908
```

In [88]:
```
grid_params = {"n_estimators" : [10,100],
               "max_depth" : range(2,10,1),
               "max_features" : ['log2']
               }
```

In [89]:
```
from sklearn.model_selection import GridSearchCV
grid_search = GridSearchCV(estimator=rfr,param_grid=grid_params,cv=5,n_jobs=-1,verb
```

In [90]:
```
grid_search.fit(x_train,y_train)
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
```

In [ ]:
```
grid_search.best_params_
```

Out[ ]: {'max_depth': 9, 'max_features': 'log2', 'n_estimators': 10}

In [ ]:
```
rfr2 = RandomForestRegressor(n_estimators=100,max_depth=10,max_features='log2')
```

In [ ]:
```
rfr2.fit(x_train,y_train)
```

Out[ ]:
```
▾                    RandomForestRegressor

RandomForestRegressor(max_depth=10, max_features='log2')
```

In [ ]:
```
rfr2.score(x_test,y_test)
```

Out[ ]: 0.8837919470333327

In [ ]:
```
y_pred = rfr2.predict(x_test)
```

In [ ]:
```
#### DecisionTreeClassifier
```

In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier()
dtc.fit(x_train,y_train)
dtc.score(x_test,y_test)
```

Out[ ]:
```
0.9968186638388123
```

In [ ]:
```
y_pred = dtc.predict(x_test)
```

In [ ]:
```
Accuracy_score = metrics.accuracy_score(y_test,y_pred)
Accuracy_score
```

Out[ ]:
```
0.9968186638388123
```

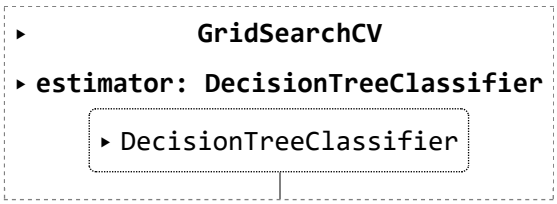
In [ ]:
```
con_matrix = metrics.confusion_matrix(y_test,y_pred)
con_matrix
```

Out[ ]:
```
array([[877,   1],
       [  2,  63]], dtype=int64)
```

In [ ]:
```
grid_param = {
    'criterion': ['gini', 'entropy'],
    'max_depth' : range(2,32,1),
    'splitter' : ['best', 'random']
}
```

In [ ]:
```
from sklearn.model_selection import GridSearchCV
grid_search = GridSearchCV(estimator=dtc,
                     param_grid=grid_param,
                     cv=5,
                    n_jobs =-1)
```

In [ ]:
```
grid_search.fit(x_train,y_train)
```

Out[ ]:


In [ ]:
```
best_parameters=grid_search.best_params_
print(best_parameters)

grid_search.best_score_
```

```
{'criterion': 'gini', 'max_depth': 4, 'splitter': 'best'}
```

Out[ ]:
```
0.9964651802745552
```

In [ ]:
```
dtc2 = DecisionTreeClassifier(criterion = 'gini', max_depth =4, splitter ='best')
dtc2.fit(x_train,y_train)
```

Out[ ]:
```
▾      DecisionTreeClassifier

DecisionTreeClassifier(max_depth=4)
```

In [ ]:
```
dtc2.score(x_test,y_test)
```

Out[ ]:
```
0.9989395546129375
```

In [ ]:

In [ ]: