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

In [2]:
```
path = 'https://raw.githubusercontent.com/ovibaridar/Data_sets/main/hepatitis_2.csv'
```

In [3]:
```
data = pd.read_csv(path)
```

In [4]:
```
data.head()
```

Out[4]:

	Class	AGE	SEX	STEROID	ANTIVIRALS	FATIGUE	MALAISE	ANOREXIA	LIVER BIG	LIVER FIRM	SPLEEN PALPABLE	SPIDERS	ASCITES	VARICES	BILIRUBIN
0	0	30	2	1.0	2	2	2	2	1.0	2.0	2.0	2.0	2.0	2.0	1.0
1	0	50	1	1.0	2	1	2	2	1.0	2.0	2.0	2.0	2.0	2.0	0.9
2	0	78	1	2.0	2	1	2	2	2.0	2.0	2.0	2.0	2.0	2.0	0.7
3	0	31	1	NaN	1	2	2	2	2.0	2.0	2.0	2.0	2.0	2.0	0.7
4	0	34	1	2.0	2	2	2	2	2.0	2.0	2.0	2.0	2.0	2.0	1.0

In [5]:
```
data.isnull().sum()
```

Out[5]:
```
Class               0
AGE                 0
SEX                 0
STEROID             1
ANTIVIRALS          0
FATIGUE             0
MALAISE             0
ANOREXIA            0
LIVER BIG           9
LIVER FIRM         10
SPLEEN PALPABLE     4
SPIDERS             4
ASCITES             4
VARICES             4
BILIRUBIN           5
ALK PHOSPHATE      28
SGOT                3
ALBUMIN            15
PROTIME            66
HISTOLOGY           0
dtype: int64
```

Remove Null Values from Data

In [6]:
```
for col in data.columns:
        null_count = data[col].isnull().sum()
        if null_count > 0:
            data[col].fillna(data[col].mean(), inplace=True)
```

In [7]:
```
data.isnull().sum()
```

Out[7]:
```
Class              0
AGE                0
SEX                0
STEROID            0
ANTIVIRALS         0
FATIGUE            0
MALAISE            0
ANOREXIA           0
LIVER BIG          0
LIVER FIRM         0
SPLEEN PALPABLE    0
SPIDERS            0
ASCITES            0
VARICES            0
BILIRUBIN          0
ALK PHOSPHATE      0
SGOT               0
ALBUMIN            0
PROTIME            0
HISTOLOGY          0
dtype: int64
```

In [8]:
```
data.tail(10)
```

Out[8]:

	Class	AGE	SEX	STEROID	ANTIVIRALS	FATIGUE	MALAISE	ANOREXIA	LIVER BIG	LIVER FIRM	SPLEEN PALPABLE	SPIDERS	ASCITES	VARICES	BIL
144	0	31	1	1.0	2	1	2	2	2.000000	2.000000	2.0	2.00	2.000000	2.00	
145	1	41	1	2.0	2	1	2	2	2.000000	1.000000	1.0	1.00	2.000000	1.00	
146	1	70	1	1.0	2	1	1	1	1.827586	1.583333	1.8	1.66	1.866667	1.88	
147	0	20	1	1.0	2	2	2	2	2.000000	1.583333	2.0	2.00	2.000000	2.00	
148	0	36	1	2.0	2	2	2	2	2.000000	2.000000	2.0	2.00	2.000000	2.00	
149	1	46	1	2.0	2	1	1	1	2.000000	2.000000	2.0	1.00	1.000000	1.00	
150	0	44	1	2.0	2	1	2	2	2.000000	1.000000	2.0	2.00	2.000000	2.00	
151	0	61	1	1.0	2	1	1	2	1.000000	1.000000	2.0	1.00	2.000000	2.00	
152	0	53	2	1.0	2	1	2	2	2.000000	2.000000	1.0	1.00	2.000000	1.00	
153	1	43	1	2.0	2	1	2	2	2.000000	2.000000	1.0	1.00	1.000000	2.00	

In [9]:
```
data.head()
```

Out[9]:

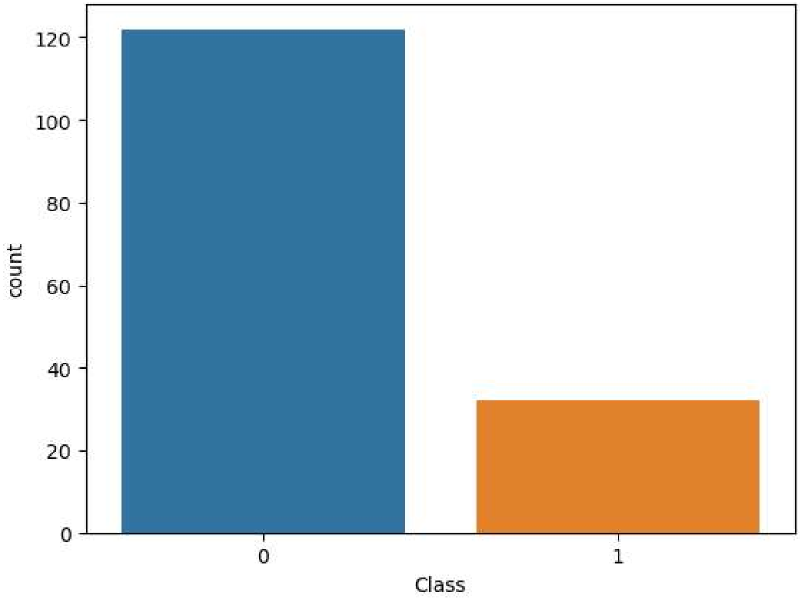
	Class	AGE	SEX	STEROID	ANTIVIRALS	FATIGUE	MALAISE	ANOREXIA	LIVER BIG	LIVER FIRM	SPLEEN PALPABLE	SPIDERS	ASCITES	VARICES	BILIRUBIN
0	0	30	2	1.000000	2	2	2	2	1.0	2.0	2.0	2.0	2.0	2.0	1.0
1	0	50	1	1.000000	2	1	2	2	1.0	2.0	2.0	2.0	2.0	2.0	0.9
2	0	78	1	2.000000	2	1	2	2	2.0	2.0	2.0	2.0	2.0	2.0	0.7
3	0	31	1	1.509804	1	2	2	2	2.0	2.0	2.0	2.0	2.0	2.0	0.7
4	0	34	1	2.000000	2	2	2	2	2.0	2.0	2.0	2.0	2.0	2.0	1.0

Graphical visualization

In [57]:
```
import seaborn as sns
```

In [58]:
```
sns.countplot(x= 'Class', data = data )
```

Out[58]:
```
<Axes: xlabel='Class', ylabel='count'>
```



Split the dataset into features (X) and target variable (y), where 1 indicates a positive case and 0 indicates a negative case.

In [59]:
```
x = data.drop('Class' , axis=1)
y = data[['Class']]
```

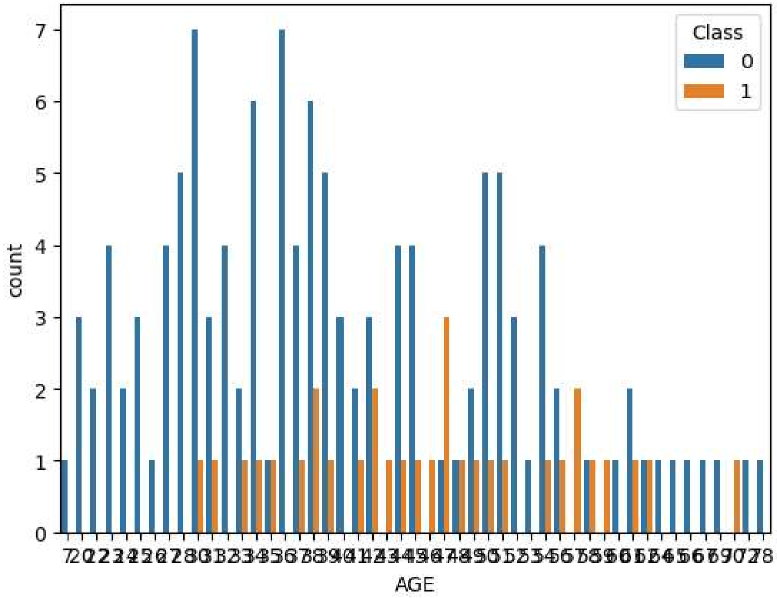
In [60]:
```
x.head()
```

Out[60]:

	AGE	SEX	STEROID	ANTIVIRALS	FATIGUE	MALAISE	ANOREXIA	LIVER BIG	LIVER FIRM	SPLEEN PALPABLE	SPIDERS	ASCITES	VARICES	BILIRUBIN	PHOS
0	30	2	1.000000	2	2	2	2	1.0	2.0	2.0	2.0	2.0	2.0	1.0	85
1	50	1	1.000000	2	1	2	2	1.0	2.0	2.0	2.0	2.0	2.0	0.9	135
2	78	1	2.000000	2	1	2	2	2.0	2.0	2.0	2.0	2.0	2.0	0.7	96
3	31	1	1.509804	1	2	2	2	2.0	2.0	2.0	2.0	2.0	2.0	0.7	46
4	34	1	2.000000	2	2	2	2	2.0	2.0	2.0	2.0	2.0	2.0	1.0	105

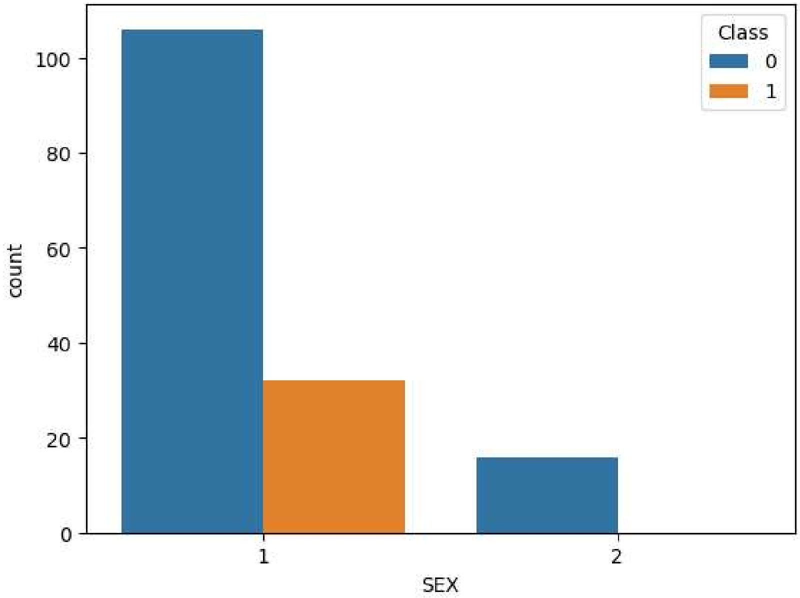
```
In [61]: sns.countplot(x='AGE' , data=data, hue='Class')
```

```
Out[61]: <Axes: xlabel='AGE', ylabel='count'>
```



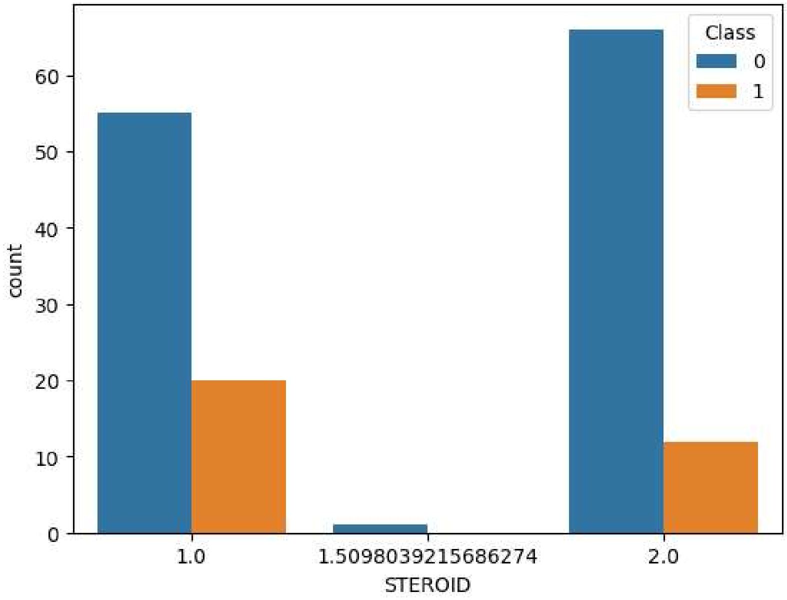
```
In [62]: sns.countplot(x='SEX' , data=data, hue='Class')
```

```
Out[62]: <Axes: xlabel='SEX', ylabel='count'>
```



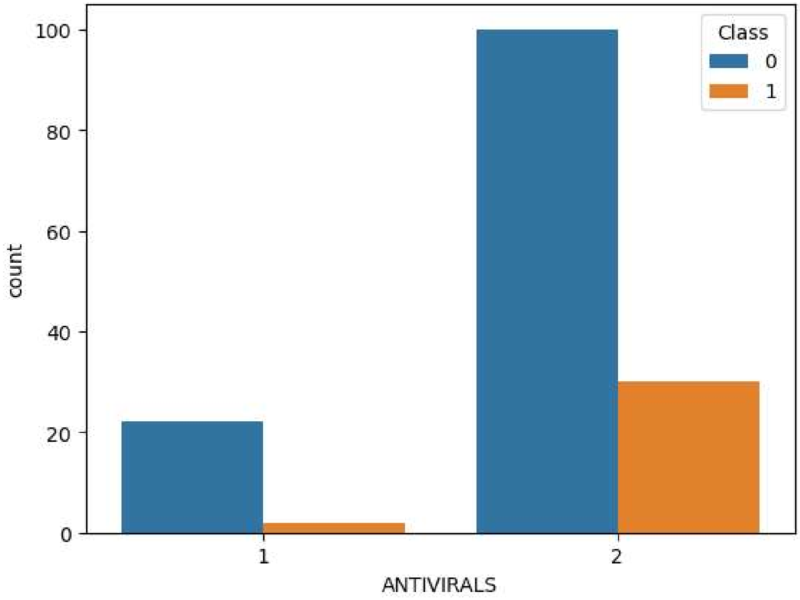
```
In [63]: sns.countplot(x='STEROID' , data=data, hue='Class')
```

```
Out[63]: <Axes: xlabel='STEROID', ylabel='count'>
```



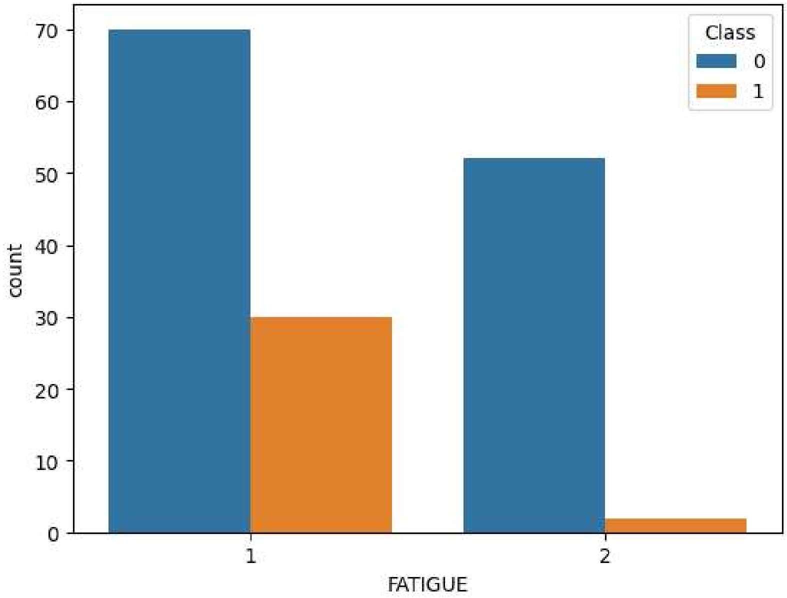
```
In [64]: sns.countplot(x='ANTIVIRALS' , data=data, hue='Class')
```

```
Out[64]: <Axes: xlabel='ANTIVIRALS', ylabel='count'>
```



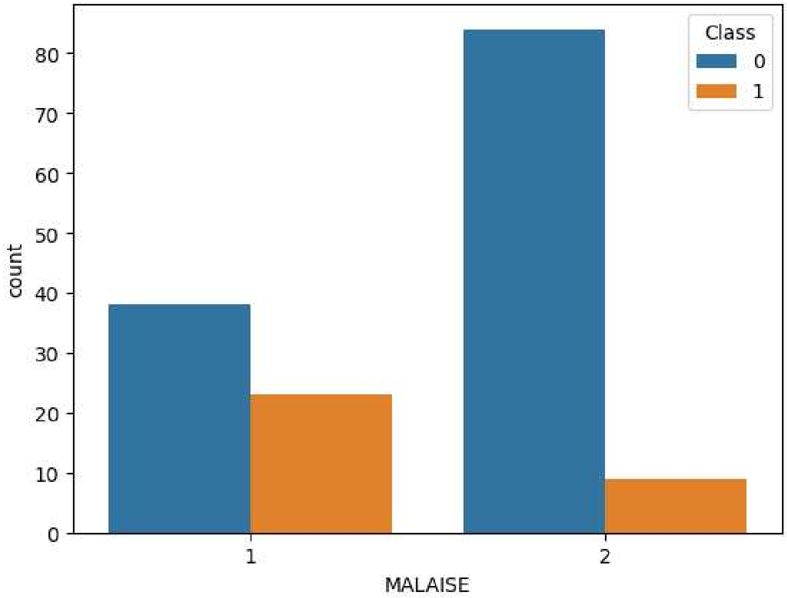
```
In [65]: sns.countplot(x='FATIGUE' , data=data, hue='Class')
```

```
Out[65]: <Axes: xlabel='FATIGUE', ylabel='count'>
```



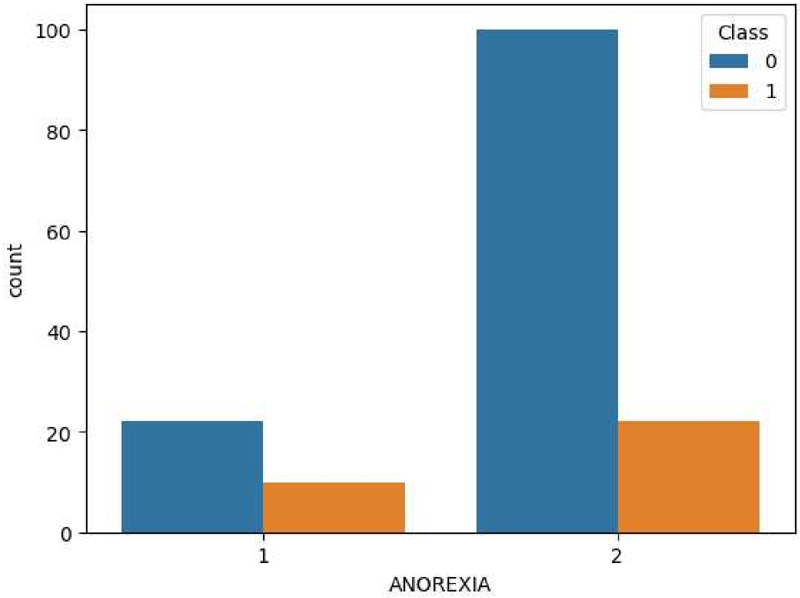
```
In [66]: sns.countplot(x='MALAISE' , data=data, hue='Class')
```

```
Out[66]: <Axes: xlabel='MALAISE', ylabel='count'>
```



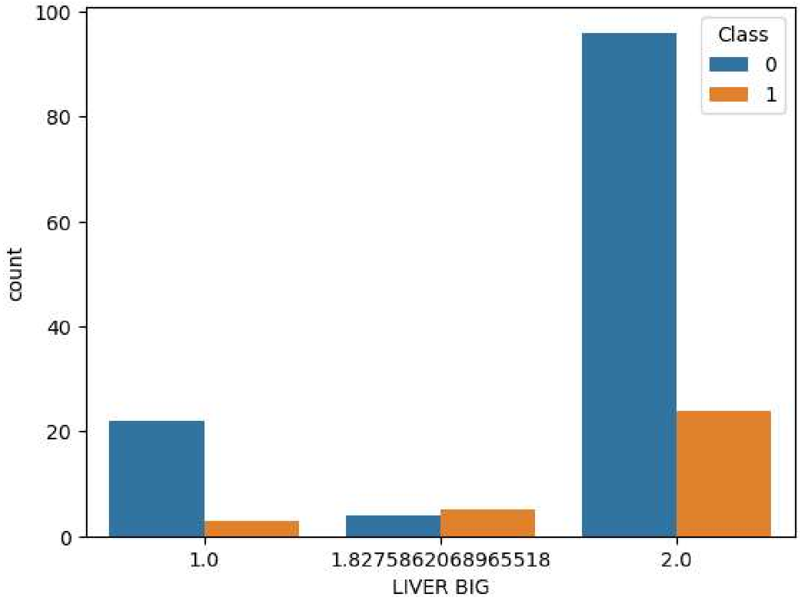
In [67]:
```
sns.countplot(x='ANOREXIA' , data=data, hue='Class')
```

Out[67]:
```
<Axes: xlabel='ANOREXIA', ylabel='count'>
```



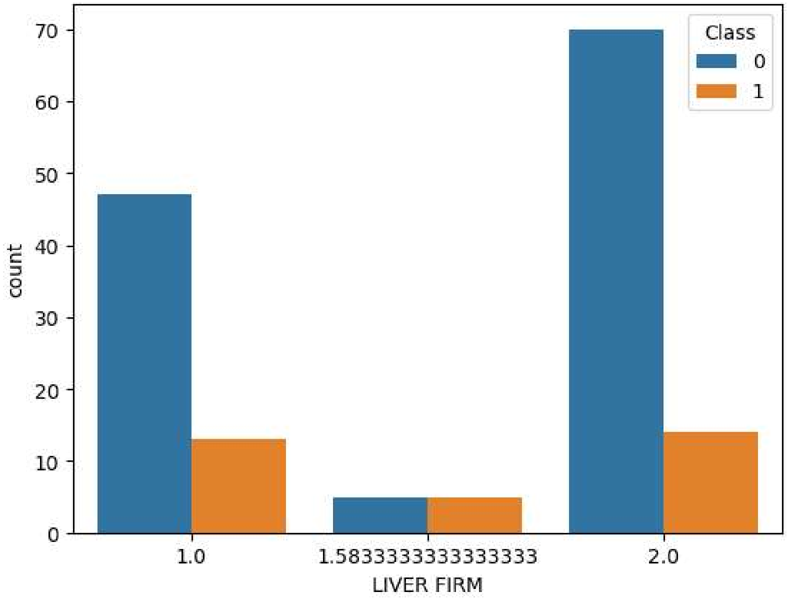
In [68]:
```
sns.countplot(x='LIVER BIG' , data=data, hue='Class')
```

Out[68]:
```
<Axes: xlabel='LIVER BIG', ylabel='count'>
```



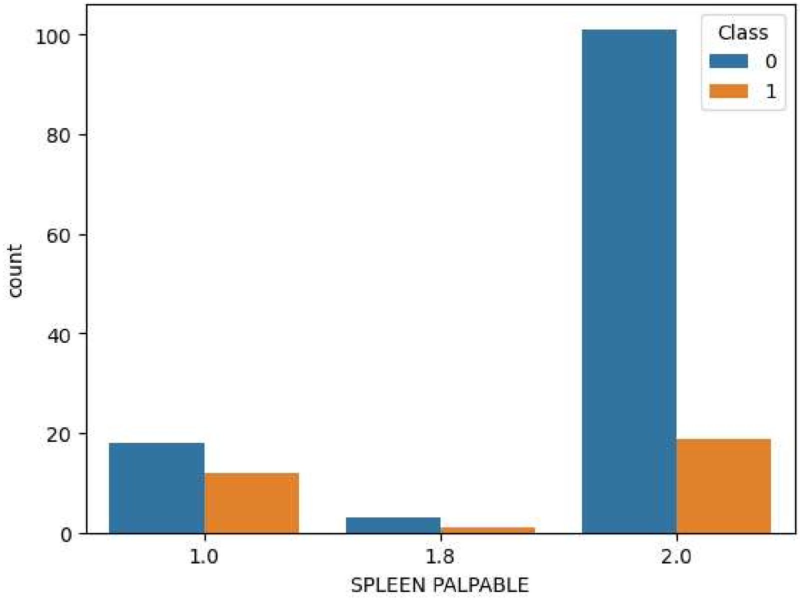
```
In [69]: sns.countplot(x='LIVER FIRM' , data=data, hue='Class')
```

```
Out[69]: <Axes: xlabel='LIVER FIRM', ylabel='count'>
```



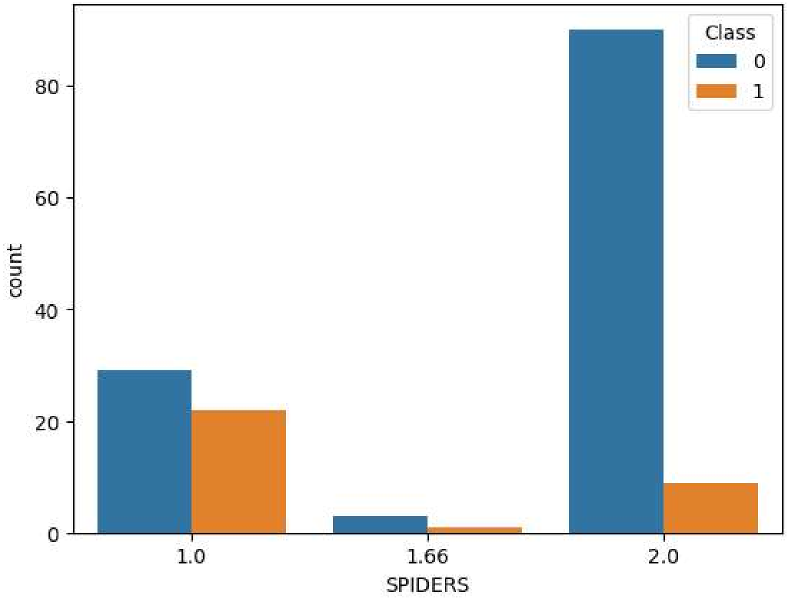
```
In [70]: sns.countplot(x='SPLEEN PALPABLE' , data=data, hue='Class')
```

```
Out[70]: <Axes: xlabel='SPLEEN PALPABLE', ylabel='count'>
```



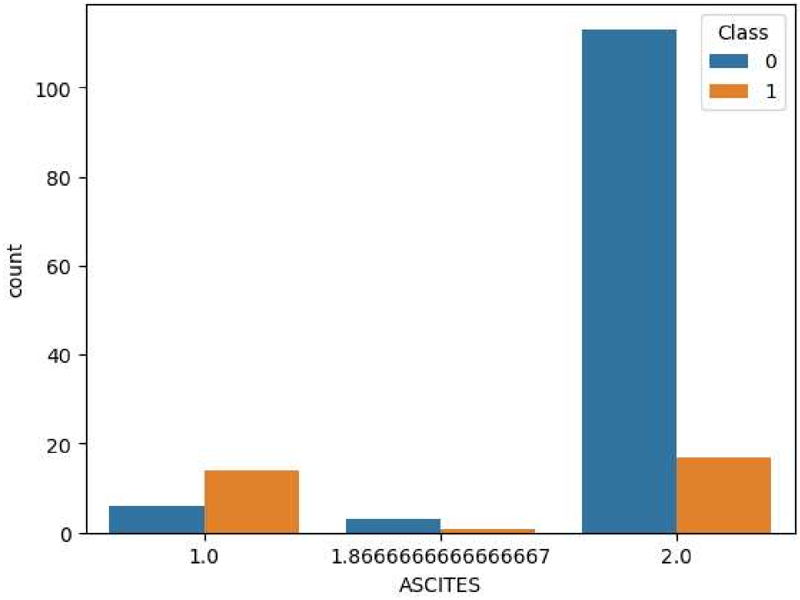
```
In [71]: sns.countplot(x='SPIDERS' , data=data, hue='Class')
```

```
Out[71]: <Axes: xlabel='SPIDERS', ylabel='count'>
```



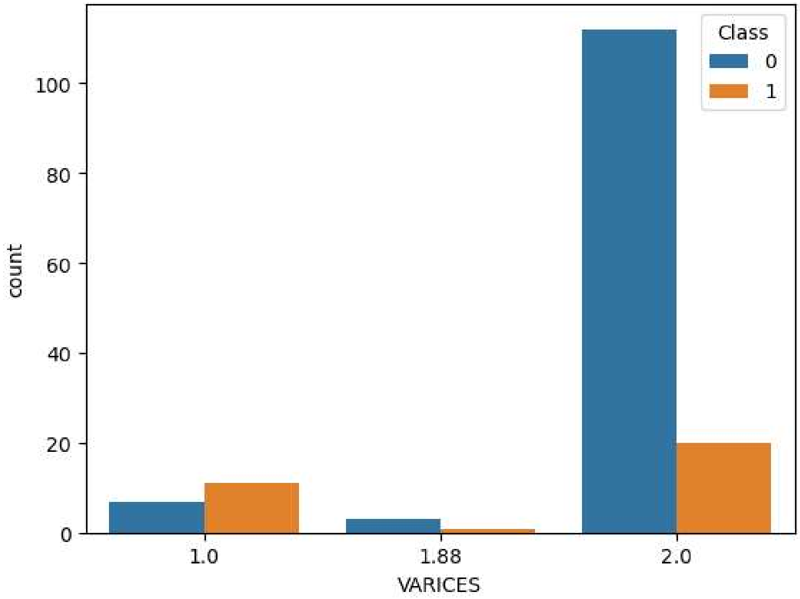
```
In [72]: sns.countplot(x='ASCITES' , data=data, hue='Class')
```

```
Out[72]: <Axes: xlabel='ASCITES', ylabel='count'>
```



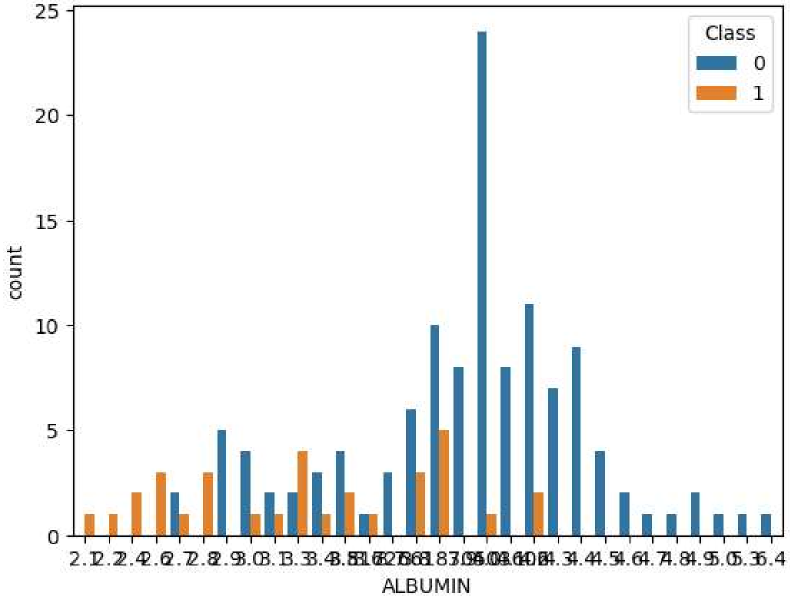
```
In [73]: 
sns.countplot(x='VARICES' , data=data, hue='Class')
```

```
Out[73]: <Axes: xlabel='VARICES', ylabel='count'>
```



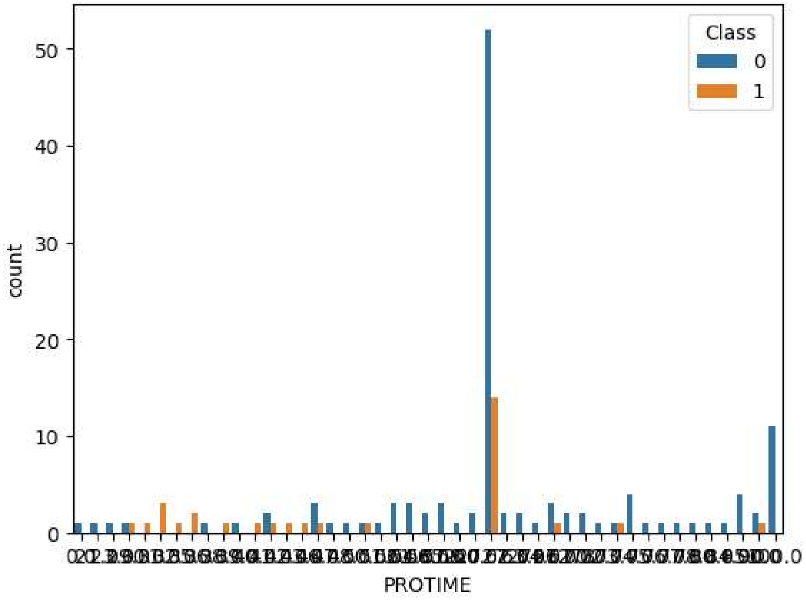
```
In [74]: sns.countplot(x='ALBUMIN' , data=data, hue='Class')
```

```
Out[74]: <Axes: xlabel='ALBUMIN', ylabel='count'>
```



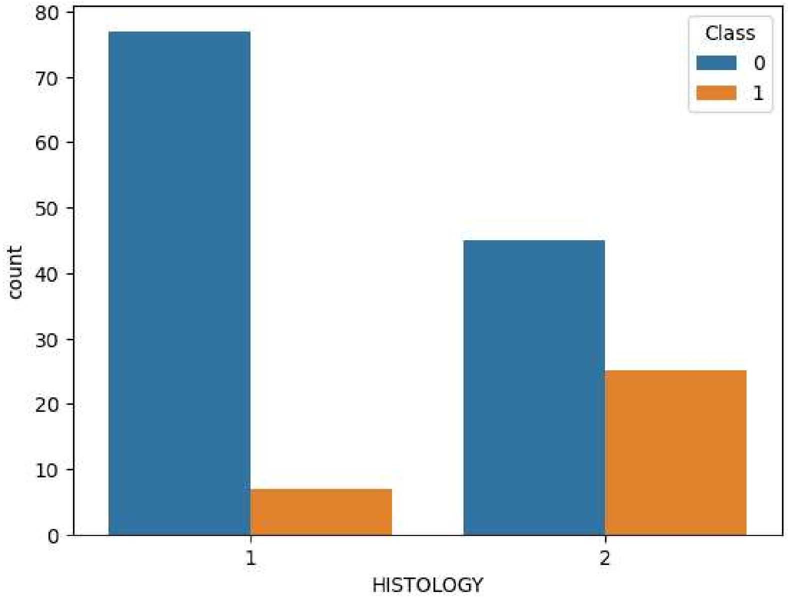
In [100]:
```
sns.countplot(x='PROTIME' , data=data, hue='Class')
```

Out[100]:
```
<Axes: xlabel='PROTIME', ylabel='count'>
```



In [101]:
```
sns.countplot(x='HISTOLOGY' , data=data, hue='Class')
```

Out[101]:
```
<Axes: xlabel='HISTOLOGY', ylabel='count'>
```



Split the dataset into training (70%) and testing sets.

In [ ]:
```
from sklearn.model_selection import train_test_split
```

In [174]:
```
xtrain,xtest,ytrain,ytest = train_test_split(x,y,random_state=40 , test_size=.25)
```

In [175]:
```
xtrain.shape
```

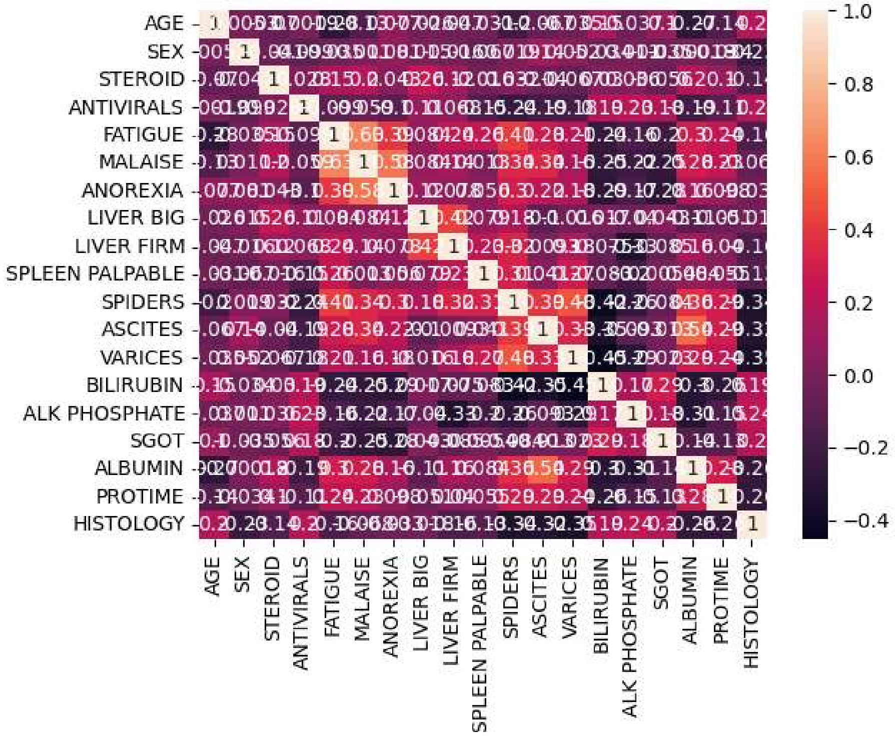
Out[175]:
```
(115, 19)
```

In [176]:
```
xtest.shape
```

Out[176]:
```
(39, 19)
```

In [177]:
```
sns.heatmap(xtrain.corr(), annot=True)
```

Out[177]:
```
<Axes: >
```



- Build a Decision Tree classifier using a suitable library (scikit-learn).

In [178]:
```
from sklearn.tree import DecisionTreeClassifier
import warnings
warnings.filterwarnings("ignore")
```

In [179]:
```
dtc = DecisionTreeClassifier()
```

- Train the model on the training set.

In [180]:
```
dtc.fit(xtrain,ytrain)
```

Out[180]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [181]:
```
dtc.score(xtest,ytest)
```

Out[181]:
```
0.7948717948717948
```

In [182]:
```
dtc.score(xtrain,ytrain)
```

Out[182]:
```
1.0
```

- Make predictions on the testing set.

In [183]:
```
perd= dtc.predict(xtest)
perd
```

Out[183]:
```
array([1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

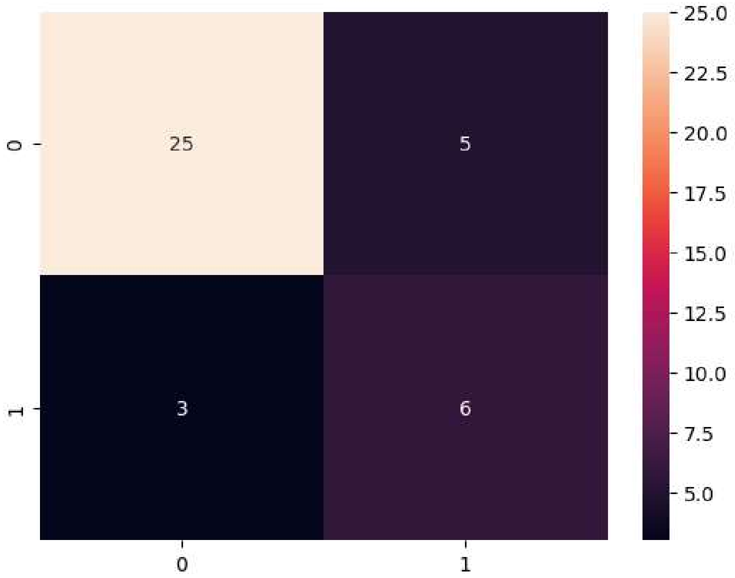
In [184]:
```
from sklearn.metrics import confusion_matrix ,precision_score ,recall_score ,accuracy_score , f1_score , RocCurveDispla
```

In [185]:
```
cm = confusion_matrix(ytest, perd)
cm
```

Out[185]:
```
array([[25,  5],
       [ 3,  6]], dtype=int64)
```

In [186]:
```
sns.heatmap(cm , annot=True)
```

Out[186]:
```
<Axes: >
```



In [187]:
```
precision_score(ytest,perd)
```

Out[187]:
```
0.5454545454545454
```

In [188]:
```
recall_score(ytest,perd)
```

Out[188]:
```
0.6666666666666666
```

In [189]:
```
accuracy_score(ytest,perd)
```

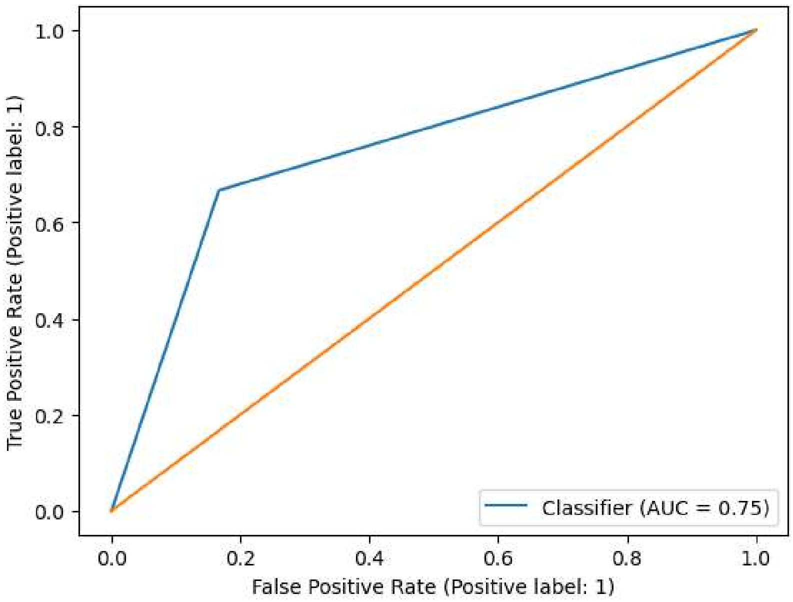
Out[189]:
```
0.7948717948717948
```

In [190]:
```
f1_score(ytest,perd)
```

Out[190]:
```
0.6
```

In [191]:
```
from matplotlib import pyplot as plt
RocCurveDisplay.from_predictions(ytest,perd)
plt.plot([0,1],[0,1])
```

Out[191]:
```
[<matplotlib.lines.Line2D at 0x251cd06ad90>]
```


True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
Classifier (AUC = 0.75)