Batch: B1 **Experiment Number: 8**

Roll Number: 16010420133 **Name Soumen samanta**

Aim of the Experiment:

To implement Decision Tree Algorithm (ID3 using library functions)

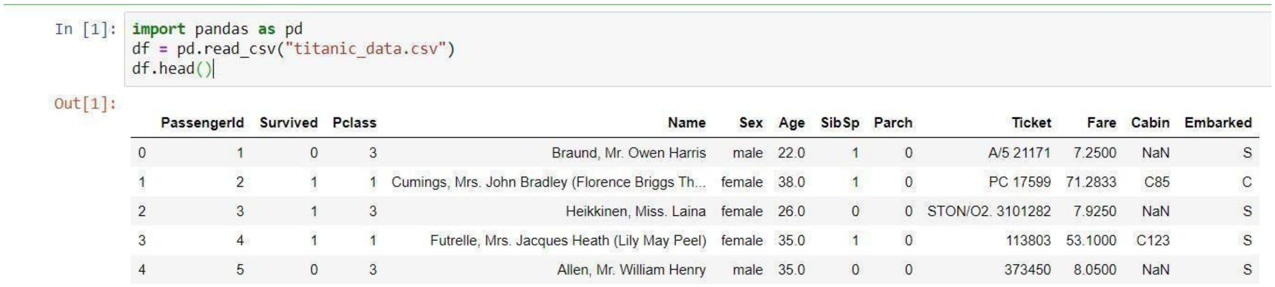
Program/ Steps:

Set up and train a decision tree classifier on the Titanic dataset and see how well the classifier performs on a validation set (80-20 train-test dataset). Find out accuracy and confusion matrix and plot created decision tree with following variations

1. Target Variable: Survived, remaining all input features
2. Target Variable: Survived, selecting subset of features as input
3. Target Variable: Survived, using transformed input feature (e.g. create new feature family = sibsp + parch, weighted_class = pclass*2 if pclass =1; pclass*3 if pclass =2; pclass*4 if pclass =3 etc)

Output/Result:

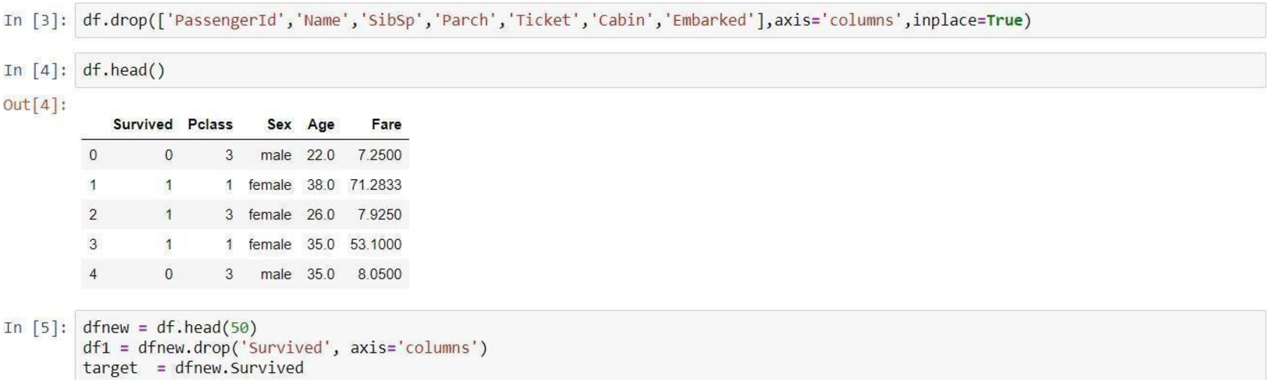
Code:



```
In [1]: import pandas as pd
        df = pd.read_csv("titanic_data.csv")
        df.head()
```

Out[1]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S



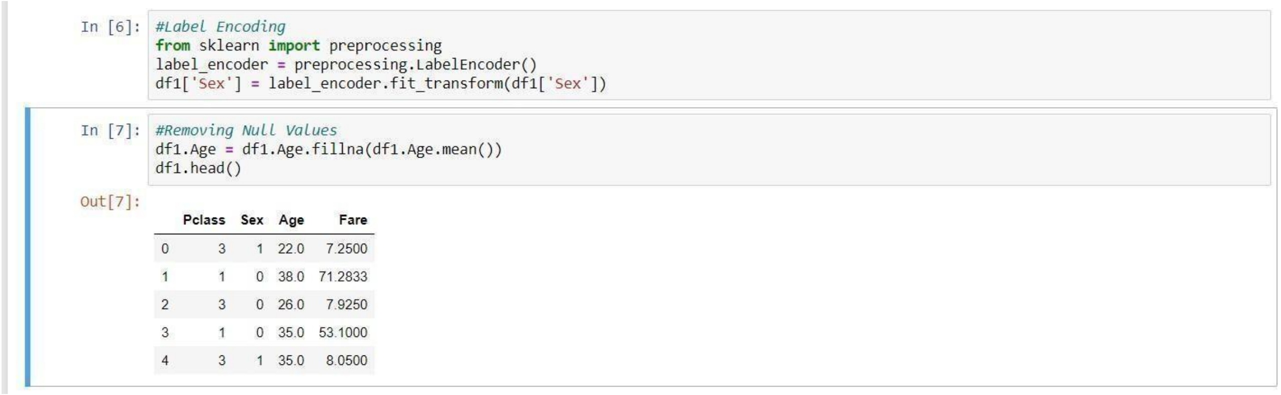
```
In [3]: df.drop(['PassengerId','Name','SibSp','Parch','Ticket','Cabin','Embarked'],axis='columns',inplace=True)
```

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

Out[4]:

	Survived	Pclass	Sex	Age	Fare
0	0	3	male	22.0	7.2500
1	1	1	female	38.0	71.2833
2	1	3	female	26.0	7.9250
3	1	1	female	35.0	53.1000
4	0	3	male	35.0	8.0500

```
In [5]: dfnew = df.head(50)
        df1 = dfnew.drop('Survived', axis='columns')
        target  = dfnew.Survived
```

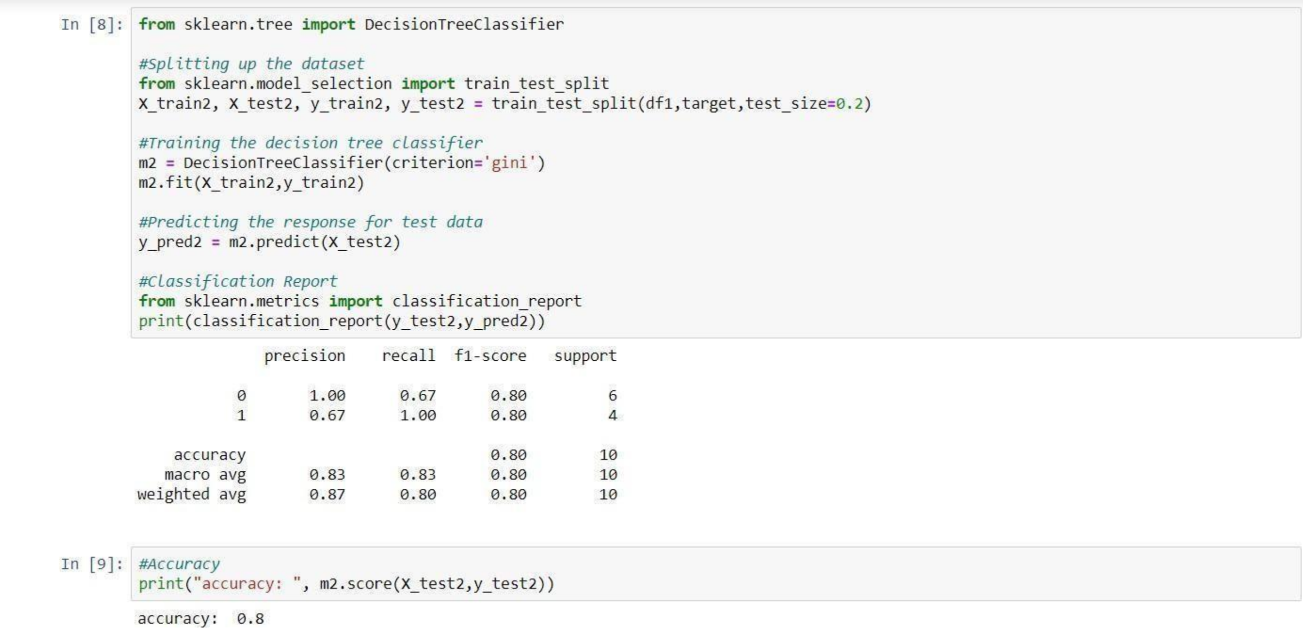


```
In [6]: #Label Encoding
        from sklearn import preprocessing
        label_encoder = preprocessing.LabelEncoder()
        df1['Sex'] = label_encoder.fit_transform(df1['Sex'])
```

```
In [7]: #Removing Null Values
        df1.Age = df1.Age.fillna(df1.Age.mean())
        df1.head()
```

Out[7]:

	Pclass	Sex	Age	Fare
0	3	1	22.0	7.2500
1	1	0	38.0	71.2833
2	3	0	26.0	7.9250
3	1	0	35.0	53.1000
4	3	1	35.0	8.0500

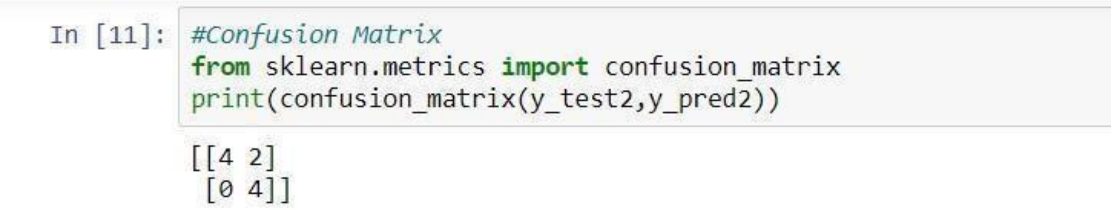


```
In [8]: from sklearn.tree import DecisionTreeClassifier

        #Splitting up the dataset
        from sklearn.model_selection import train_test_split
        X_train2, X_test2, y_train2, y_test2 = train_test_split(df1,target,test_size=0.2)

        #Training the decision tree classifier
        m2 = DecisionTreeClassifier(criterion='gini')
        m2.fit(X_train2,y_train2)

        #Predicting the response for test data
        y_pred2 = m2.predict(X_test2)

        #Classification Report
        from sklearn.metrics import classification_report
        print(classification_report(y_test2,y_pred2))
```

```
              precision    recall  f1-score   support

           0       1.00      0.67      0.80         6
           1       0.67      1.00      0.80         4

    accuracy                           0.80        10
   macro avg       0.83      0.83      0.80        10
weighted avg       0.87      0.80      0.80        10
```

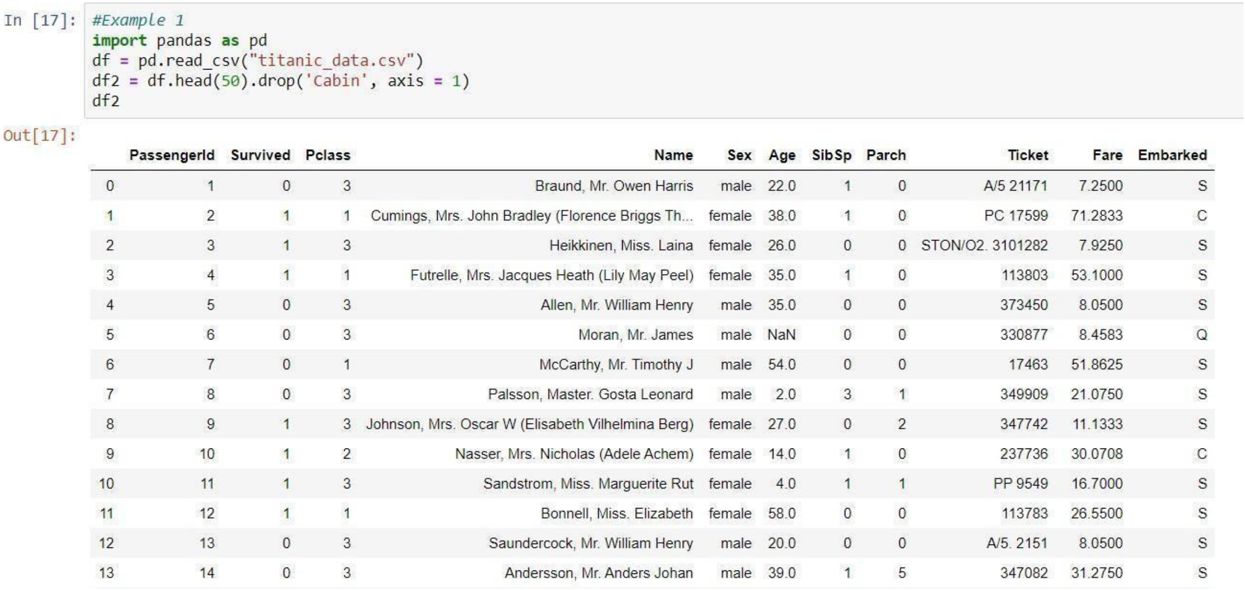
```
In [9]: #Accuracy
        print("accuracy: ", m2.score(X_test2,y_test2))
```

```
accuracy:  0.8
```



```
In [11]: #Confusion Matrix
         from sklearn.metrics import confusion_matrix
         print(confusion_matrix(y_test2,y_pred2))
```

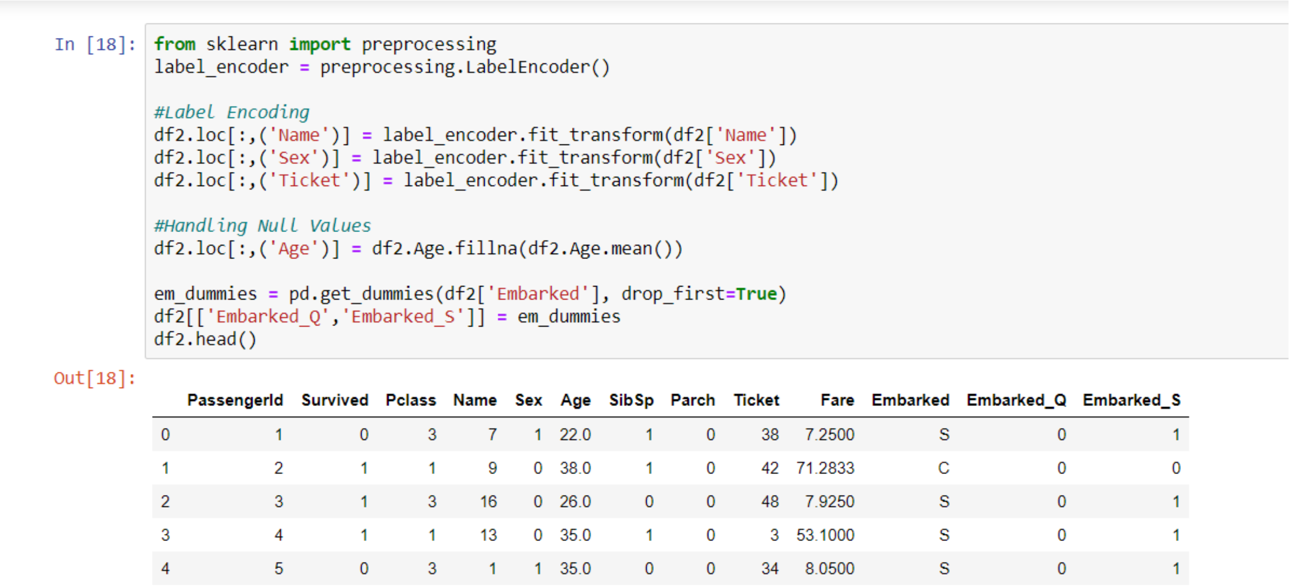
```
[[4 2]
 [0 4]]
```



```
In [17]: #Example 1
         import pandas as pd
         df = pd.read_csv("titanic_data.csv")
         df2 = df.head(50).drop('Cabin', axis = 1)
         df2
```

Out[17]:

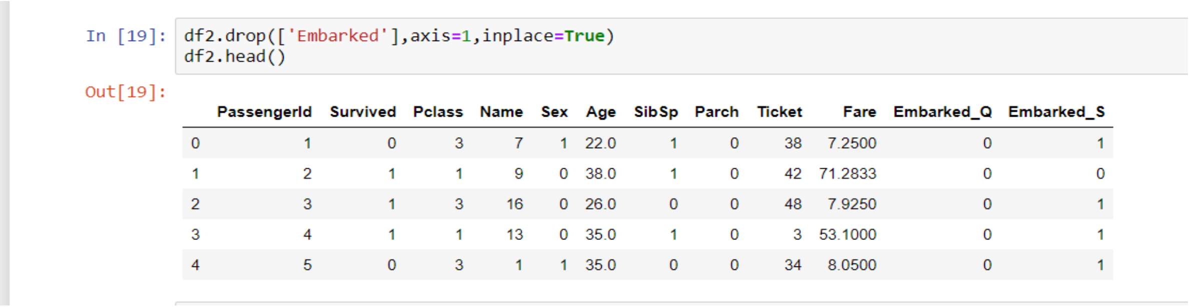
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	C
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	S
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	S
12	13	0	3	Saundercock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.0500	S
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.2750	S



```
In [18]: from sklearn import preprocessing
         label_encoder = preprocessing.LabelEncoder()

         #Label Encoding
         df2.loc[:,('Name')] = label_encoder.fit_transform(df2['Name'])
         df2.loc[:,('Sex')] = label_encoder.fit_transform(df2['Sex'])
         df2.loc[:,('Ticket')] = label_encoder.fit_transform(df2['Ticket'])

         #Handling Null Values
         df2.loc[:,('Age')] = df2.Age.fillna(df2.Age.mean())

         em_dummies = pd.get_dummies(df2['Embarked'], drop_first=True)
         df2[['Embarked_Q','Embarked_S']] = em_dummies
         df2.head()
```

Out[18]:

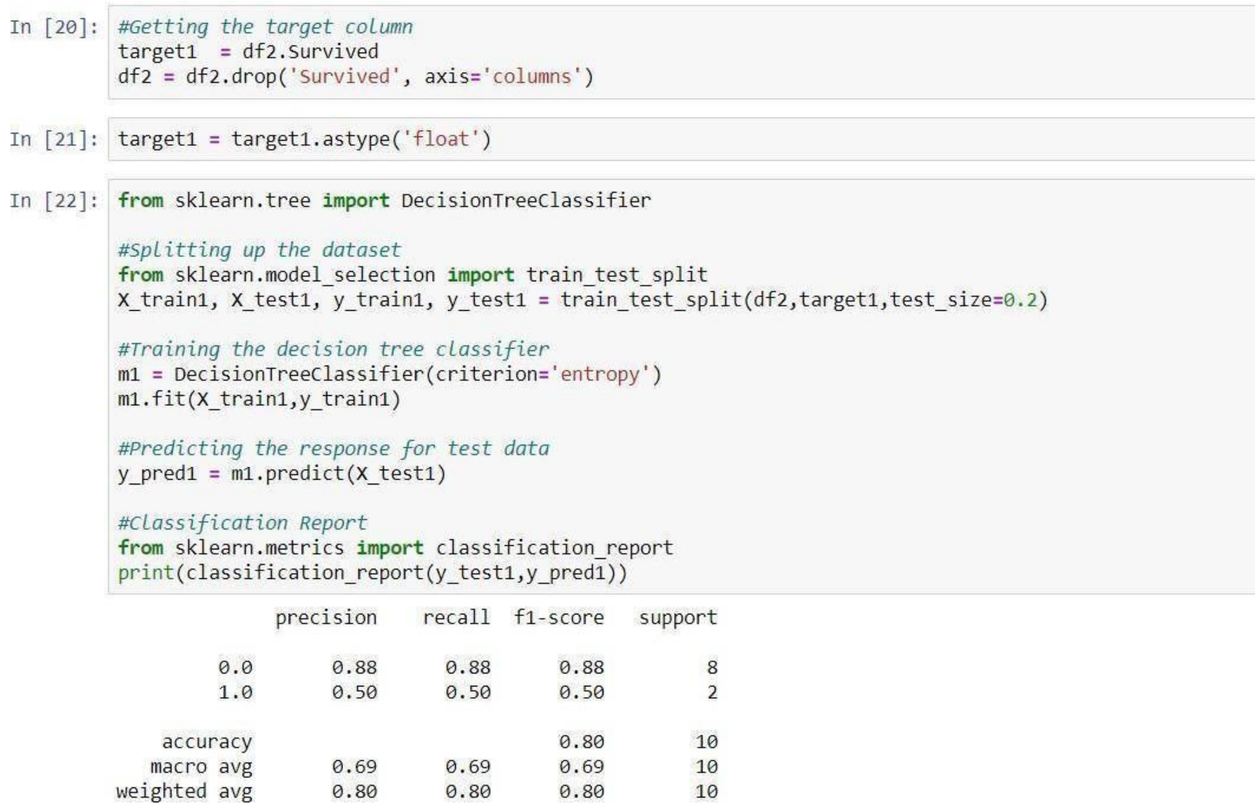
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Embarked_Q	Embarked_S
0	1	0	3	7	1	22.0	1	0	38	7.2500	S	0	1
1	2	1	1	9	0	38.0	1	0	42	71.2833	C	0	0
2	3	1	3	16	0	26.0	0	0	48	7.9250	S	0	1
3	4	1	1	13	0	35.0	1	0	3	53.1000	S	0	1
4	5	0	3	1	1	35.0	0	0	34	8.0500	S	0	1



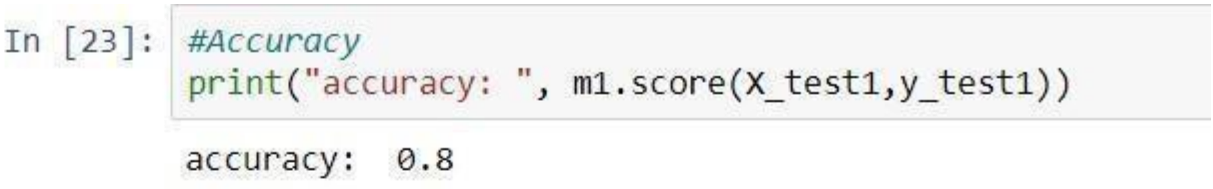
```
In [19]: df2.drop(['Embarked'],axis=1,inplace=True)
         df2.head()
```

Out[19]:

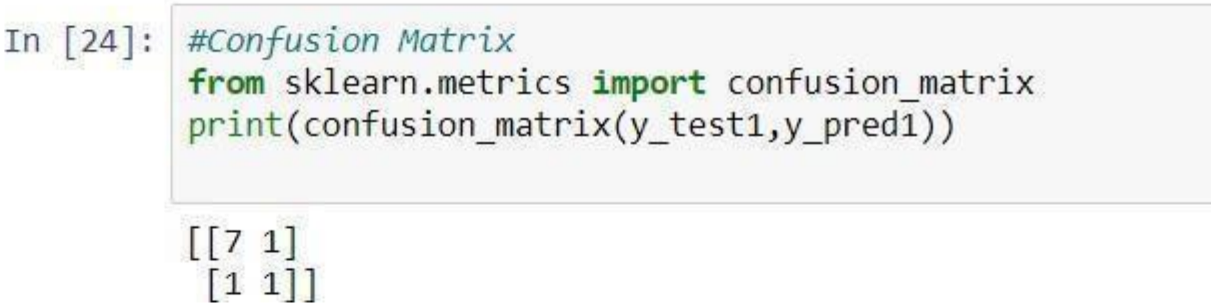
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked_Q	Embarked_S
0	1	0	3	7	1	22.0	1	0	38	7.2500	0	1
1	2	1	1	9	0	38.0	1	0	42	71.2833	0	0
2	3	1	3	16	0	26.0	0	0	48	7.9250	0	1
3	4	1	1	13	0	35.0	1	0	3	53.1000	0	1
4	5	0	3	1	1	35.0	0	0	34	8.0500	0	1



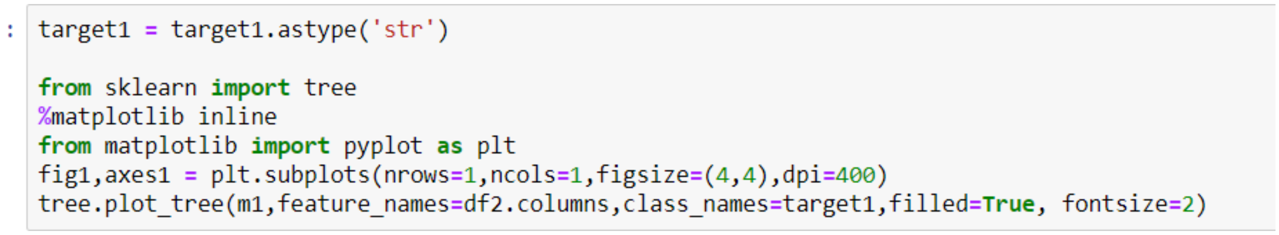
```
In [20]: #Getting the target column
         target1  = df2.Survived
         df2 = df2.drop('Survived', axis='columns')

In [21]: target1 = target1.astype('float')

In [22]: from sklearn.tree import DecisionTreeClassifier

         #Splitting up the dataset
         from sklearn.model_selection import train_test_split
         X_train1, X_test1, y_train1, y_test1 = train_test_split(df2,target1,test_size=0.2)

         #Training the decision tree classifier
         m1 = DecisionTreeClassifier(criterion='entropy')
         m1.fit(X_train1,y_train1)

         #Predicting the response for test data
         y_pred1 = m1.predict(X_test1)

         #Classification Report
         from sklearn.metrics import classification_report
         print(classification_report(y_test1,y_pred1))

                       precision    recall  f1-score   support

                  0.0       0.88      0.88      0.88         8
                  1.0       0.50      0.50      0.50         2

             accuracy                           0.80        10
            macro avg       0.69      0.69      0.69        10
         weighted avg       0.80      0.80      0.80        10
```



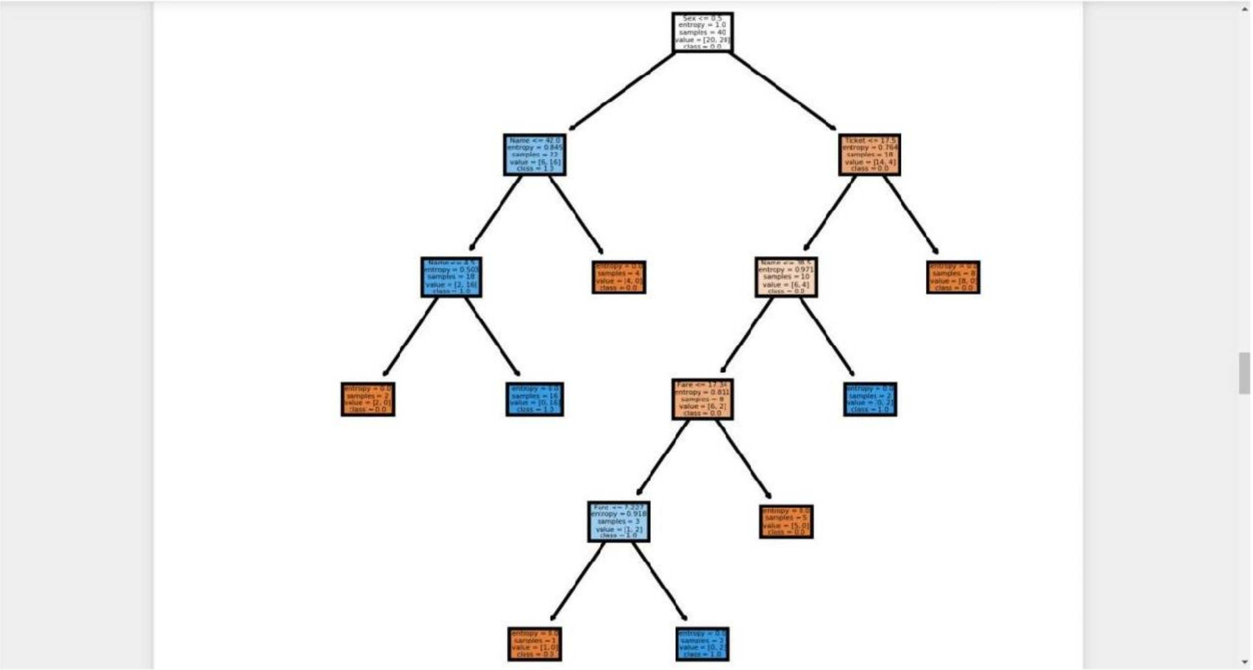
```
In [23]: #Accuracy
         print("accuracy: ", m1.score(X_test1,y_test1))

         accuracy:  0.8
```

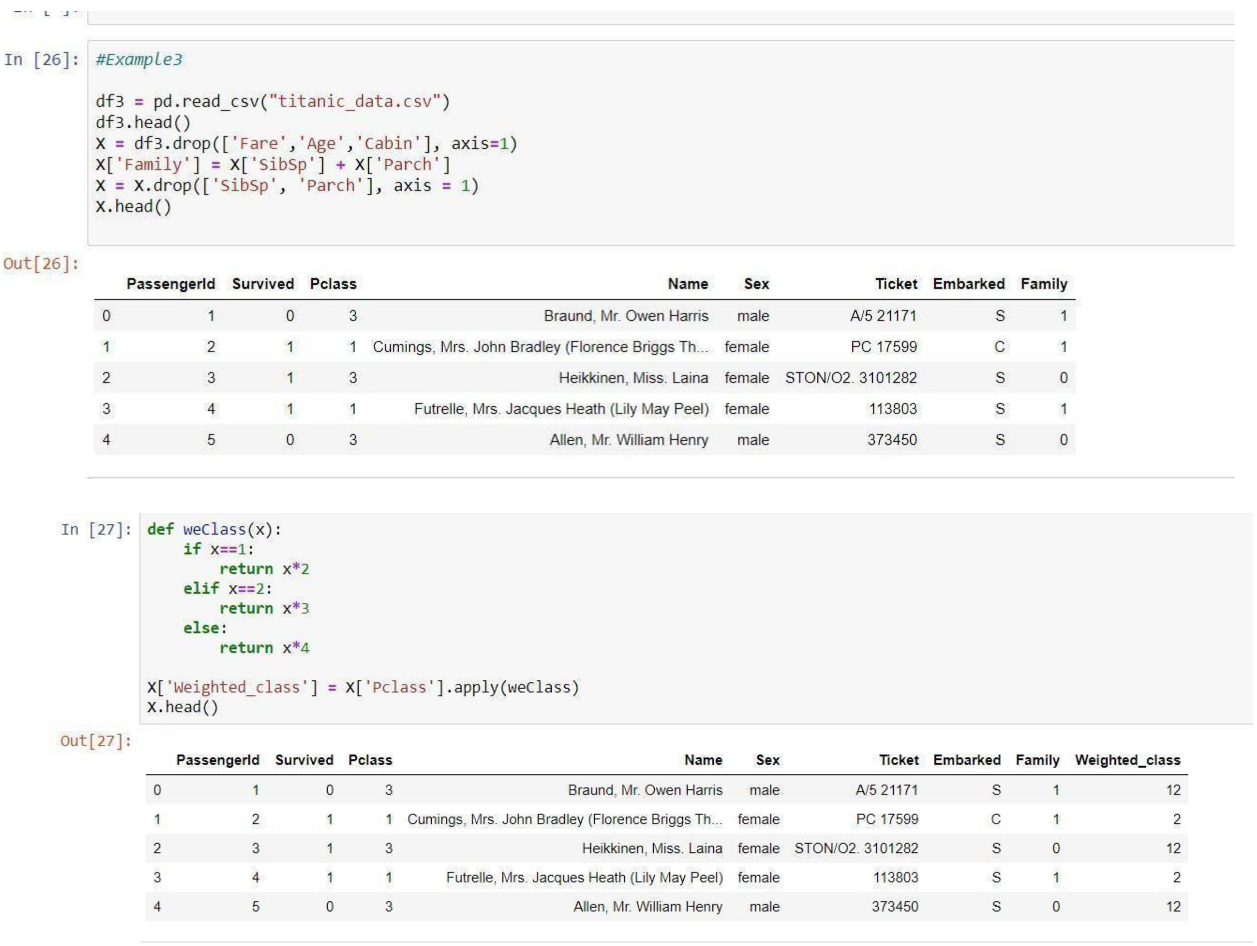


```
In [24]: #Confusion Matrix
         from sklearn.metrics import confusion_matrix
         print(confusion_matrix(y_test1,y_pred1))

         [[7 1]
          [1 1]]
```



```
: target1 = target1.astype('str')

  from sklearn import tree
  %matplotlib inline
  from matplotlib import pyplot as plt
  fig1,axes1 = plt.subplots(nrows=1,ncols=1,figsize=(4,4),dpi=400)
  tree.plot_tree(m1,feature_names=df2.columns,class_names=target1,filled=True, fontsize=2)
```





In [26]:
```
#Example3

df3 = pd.read_csv("titanic_data.csv")
df3.head()
X = df3.drop(['Fare','Age','Cabin'], axis=1)
X['Family'] = X['SibSp'] + X['Parch']
X = X.drop(['SibSp', 'Parch'], axis = 1)
X.head()
```

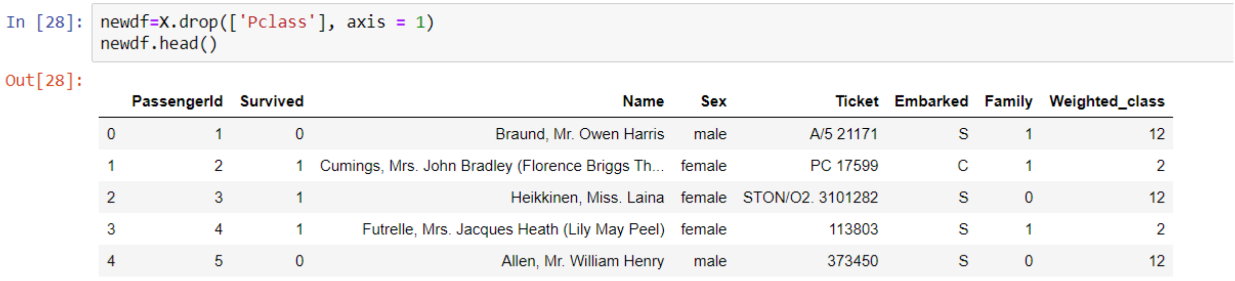
Out[26]:

	PassengerId	Survived	Pclass	Name	Sex	Ticket	Embarked	Family
0	1	0	3	Braund, Mr. Owen Harris	male	A/5 21171	S	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	PC 17599	C	1
2	3	1	3	Heikkinen, Miss. Laina	female	STON/O2. 3101282	S	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	113803	S	1
4	5	0	3	Allen, Mr. William Henry	male	373450	S	0

In [27]:
```
def weClass(x):
    if x==1:
        return x*2
    elif x==2:
        return x*3
    else:
        return x*4

X['Weighted_class'] = X['Pclass'].apply(weClass)
X.head()
```

Out[27]:

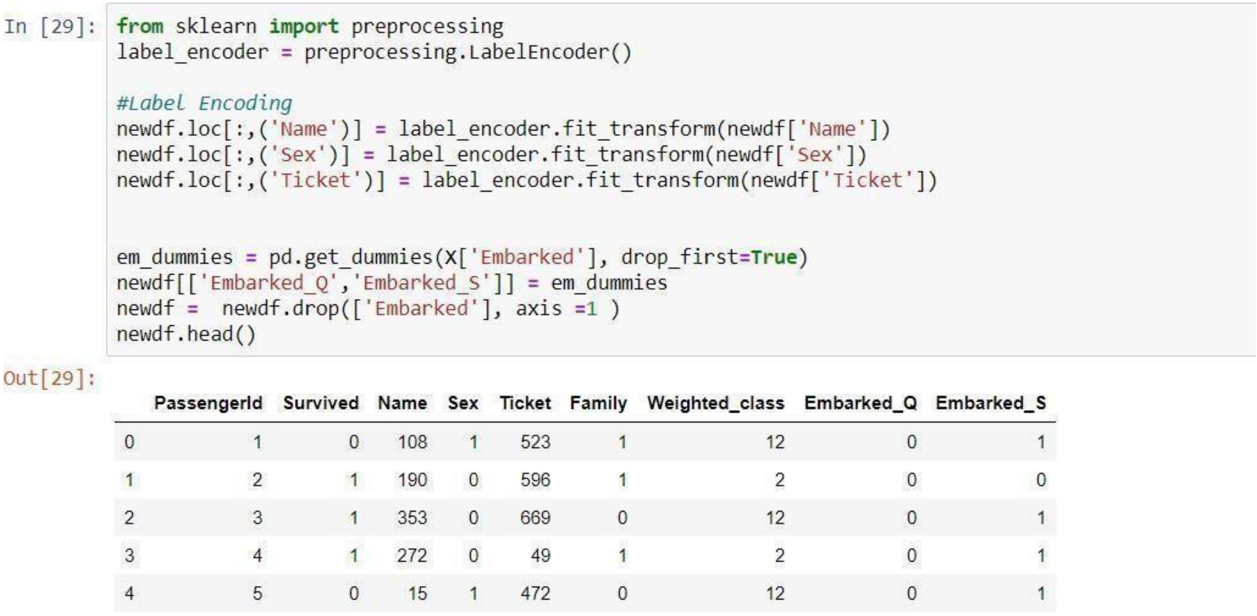
	PassengerId	Survived	Pclass	Name	Sex	Ticket	Embarked	Family	Weighted_class
0	1	0	3	Braund, Mr. Owen Harris	male	A/5 21171	S	1	12
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	PC 17599	C	1	2
2	3	1	3	Heikkinen, Miss. Laina	female	STON/O2. 3101282	S	0	12
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	113803	S	1	2
4	5	0	3	Allen, Mr. William Henry	male	373450	S	0	12



In [28]:
```
newdf=X.drop(['Pclass'], axis = 1)
newdf.head()
```

Out[28]:

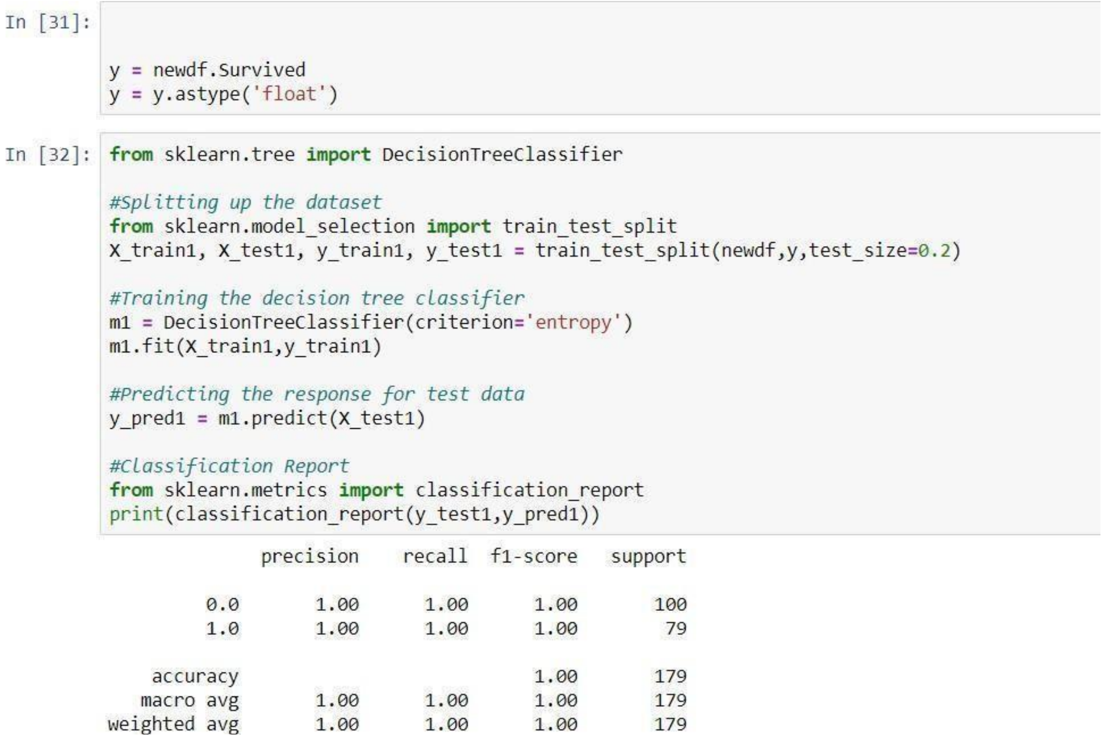
	PassengerId	Survived	Name	Sex	Ticket	Embarked	Family	Weighted_class
0	1	0	Braund, Mr. Owen Harris	male	A/5 21171	S	1	12
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	PC 17599	C	1	2
2	3	1	Heikkinen, Miss. Laina	female	STON/O2. 3101282	S	0	12
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	113803	S	1	2
4	5	0	Allen, Mr. William Henry	male	373450	S	0	12



In [29]:
```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()

#Label Encoding
newdf.loc[:,('Name')] = label_encoder.fit_transform(newdf['Name'])
newdf.loc[:,('Sex')] = label_encoder.fit_transform(newdf['Sex'])
newdf.loc[:,('Ticket')] = label_encoder.fit_transform(newdf['Ticket'])


em_dummies = pd.get_dummies(X['Embarked'], drop_first=True)
newdf[['Embarked_Q','Embarked_S']] = em_dummies
newdf =  newdf.drop(['Embarked'], axis =1 )
newdf.head()
```

Out[29]:

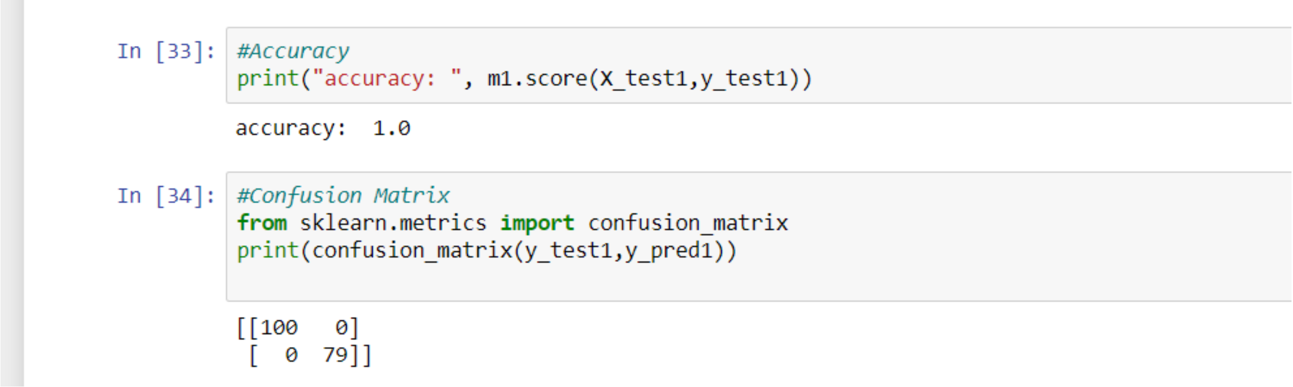
	PassengerId	Survived	Name	Sex	Ticket	Family	Weighted_class	Embarked_Q	Embarked_S
0	1	0	108	1	523	1	12	0	1
1	2	1	190	0	596	1	2	0	0
2	3	1	353	0	669	0	12	0	1
3	4	1	272	0	49	1	2	0	1
4	5	0	15	1	472	0	12	0	1



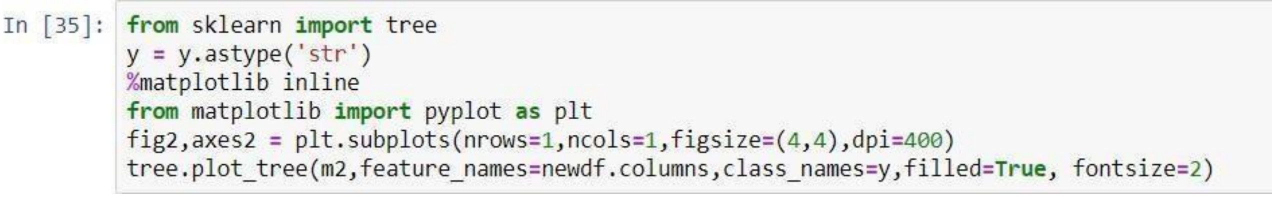
In [31]:
```


y = newdf.Survived
y = y.astype('float')
```

In [32]:
```
from sklearn.tree import DecisionTreeClassifier

#Splitting up the dataset
from sklearn.model_selection import train_test_split
X_train1, X_test1, y_train1, y_test1 = train_test_split(newdf,y,test_size=0.2)

#Training the decision tree classifier
m1 = DecisionTreeClassifier(criterion='entropy')
m1.fit(X_train1,y_train1)

#Predicting the response for test data
y_pred1 = m1.predict(X_test1)

#Classification Report
from sklearn.metrics import classification_report
print(classification_report(y_test1,y_pred1))
```

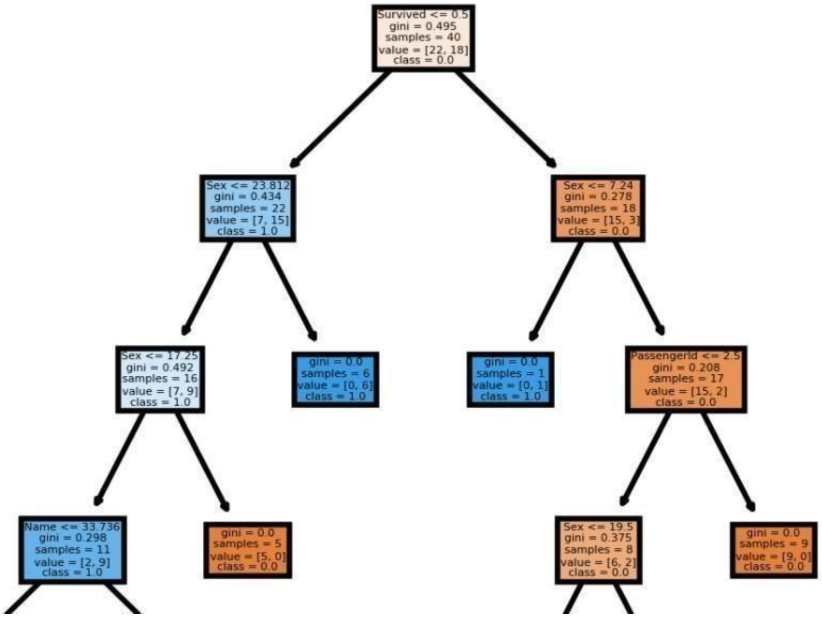
```
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00       100
         1.0       1.00      1.00      1.00        79

    accuracy                           1.00       179
   macro avg       1.00      1.00      1.00       179
weighted avg       1.00      1.00      1.00       179
```

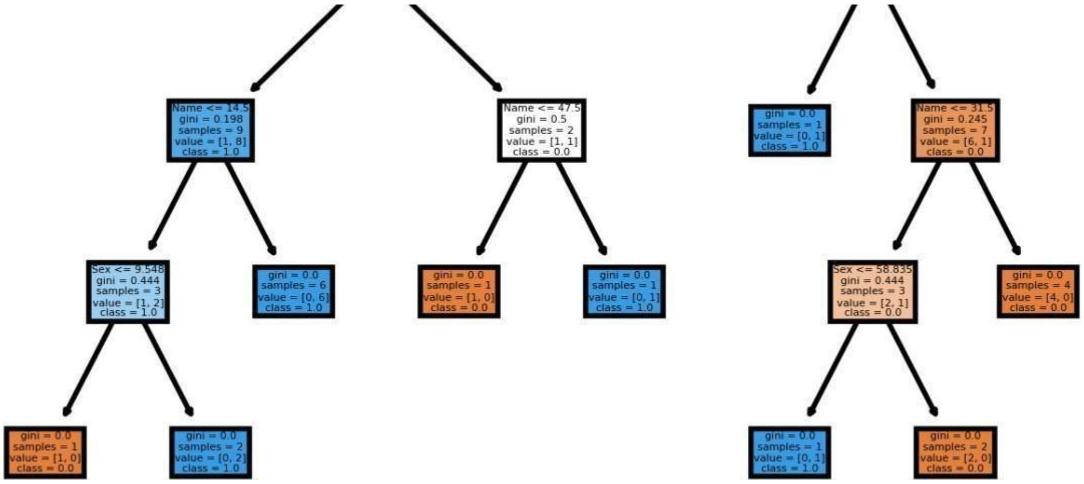


```
In [33]: #Accuracy
         print("accuracy: ", m1.score(X_test1,y_test1))

         accuracy:  1.0

In [34]: #Confusion Matrix
         from sklearn.metrics import confusion_matrix
         print(confusion_matrix(y_test1,y_pred1))

         [[100   0]
          [  0  79]]
```



```
In [35]: from sklearn import tree
         y = y.astype('str')
         %matplotlib inline
         from matplotlib import pyplot as plt
         fig2,axes2 = plt.subplots(nrows=1,ncols=1,figsize=(4,4),dpi=400)
         tree.plot_tree(m2,feature_names=newdf.columns,class_names=y,filled=True, fontsize=2)
```





Outcomes:

CO5 Understand fundamentals of learning in AI.

Conclusion: (based on the Results and outcomes achieved)

Decision Tree Algorithms (ID3 using library functions) were implemented .

References:

- Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Second Edition, Pearson Publication
- Luger, George F. Artificial Intelligence: Structures and strategies for complex problem solving, 2009 ,6th Edition, Pearson Education