Decision_Tree

October 14, 2021

[54]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import tree
import seaborn as sns
sns.set(style="white")
sns.set(style="whitegrid", color_codes=True)
from sklearn import metrics
from sklearn.metrics import classification_report, confusion_matrix
```

[55]:
```
# load the SANK dataset:
sank_data=pd.read_csv("Sank.csv",header=0)
```

[56]:
```
print(sank_data.shape)
print(list(sank_data.columns))
```

```
(891, 5)
['Alive', 'Class', 'Sex', 'Age', 'Fare']
```

[57]:
```
sank_data
```

[57]:

	Alive	Class	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
..	…	…	…	…	…
886	0	2	male	27.0	13.0000
887	1	1	female	19.0	30.0000
888	0	3	female	NaN	23.4500
889	1	1	male	26.0	30.0000
890	0	3	male	32.0	7.7500

[891 rows x 5 columns]

[58]:
```
sank_data = sank_data.dropna()
```

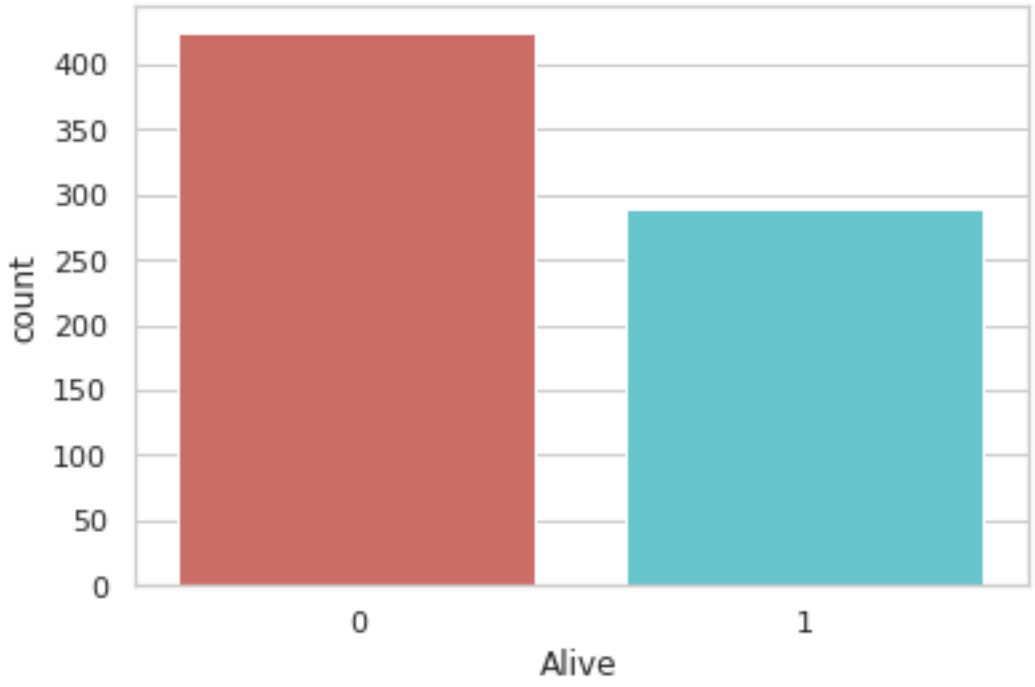
[59]:
```
sank_data
```

[59]:

	Alive	Class	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
..	…	…	…	…	…
885	0	3	female	39.0	29.1250
886	0	2	male	27.0	13.0000
887	1	1	female	19.0	30.0000
889	1	1	male	26.0	30.0000
890	0	3	male	32.0	7.7500

[714 rows x 5 columns]

[60]:
```
sns.countplot(x="Alive",data=sank_data,palette='hls')
plt.show()
```



[61]:
```
X = sank_data.loc[:, sank_data.columns != 'Alive']
y = sank_data.loc[:, sank_data.columns == 'Alive']

X
```

[61]:
```
     Class     Sex   Age     Fare
0        3    male  22.0   7.2500
1        1  female  38.0  71.2833
2        3  female  26.0   7.9250
3        1  female  35.0  53.1000
4        3    male  35.0   8.0500
..     …     …   …       …
885      3  female  39.0  29.1250
886      2    male  27.0  13.0000
887      1  female  19.0  30.0000
889      1    male  26.0  30.0000
890      3    male  32.0   7.7500

[714 rows x 4 columns]
```

[62]:
```
d = {'male': 1, 'female': 0}
X = X.replace({"Sex": d})
```

[63]:
```
X
```

[63]:
```
     Class  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
..     …  …   …       …
885      3    0  39.0  29.1250
886      2    1  27.0  13.0000
887      1    0  19.0  30.0000
889      1    1  26.0  30.0000
890      3    1  32.0   7.7500

[714 rows x 4 columns]
```

[64]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

[65]:
```
dtree = tree.DecisionTreeClassifier()
dtree = dtree.fit(X_train,y_train)

y_pred=dtree.predict(X_test)
```

[66]:
```
print("Model Score:",dtree.score(X_test,y_test))
```

```
Model Score: 0.7552447552447552
```

[67]:
```
cnfusion_matrix = metrics.confusion_matrix(y_test, y_pred)
cnfusion_matrix
```

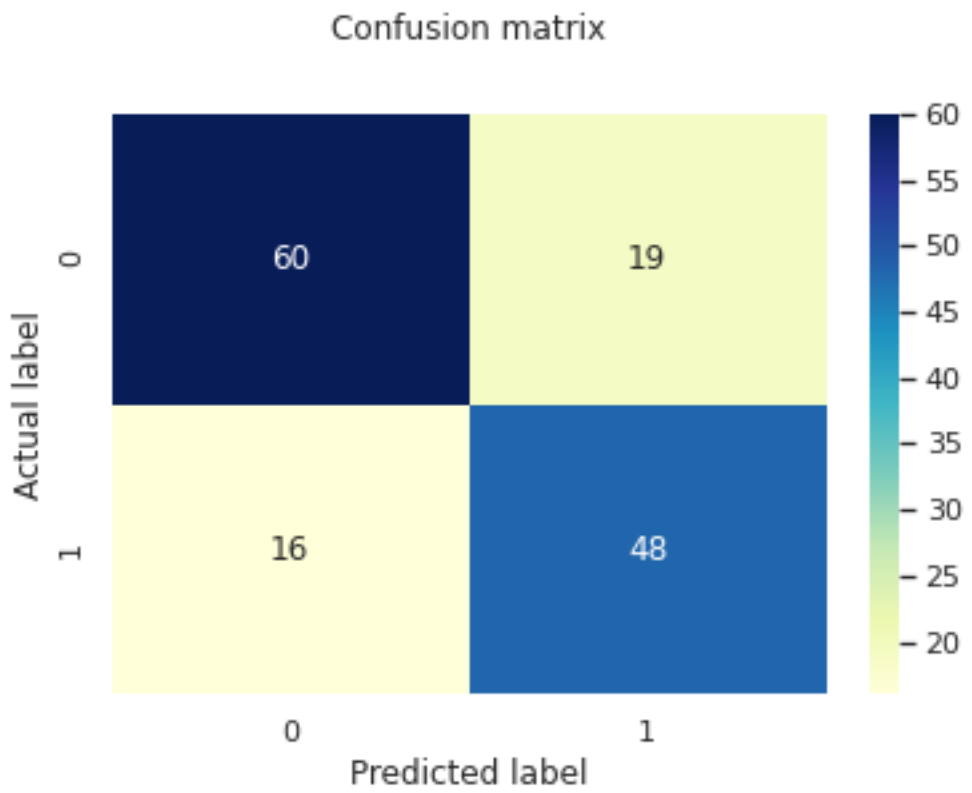
[67]:
```
array([[60, 19],
       [16, 48]])
```

[68]:
```
# Heat Map Visualization

# fig, ax = plt.subplots()
tick_marks = np.arange(len(sank_data.Alive))
plt.xticks(tick_marks, sank_data.Alive)
plt.yticks(tick_marks, sank_data.Alive)

sns.heatmap(pd.DataFrame(cnfusion_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

[68]:
```
Text(0.5, 12.5, 'Predicted label')
```



[69]:
```
print(classification_report(y_test, dtree.predict(X_test)))
```

	precision	recall	f1-score	support
0	0.79	0.76	0.77	79
1	0.72	0.75	0.73	64
accuracy			0.76	143
macro avg	0.75	0.75	0.75	143
weighted avg	0.76	0.76	0.76	143

[70]:
```
# Male = 1, Female = 0 ; As per our defined Dictionary d = {'male': 1, 'female':
↪ 0}

# Class = 1, Male, Age: 28,  Fare:  20.5
dtree.predict((np.array([1, 1, 28, 20.5]).reshape(1, -1)))
```

[70]: array([1])

[71]:
```
# Class = 2, Male, Age: 70,  Fare:  7.5
dtree.predict((np.array([2, 1, 70, 7.5]).reshape(1, -1)))
```

[71]: array([0])

[72]:
```
# Class = 3, Female, Age: 25,  Fare:  6.76
dtree.predict((np.array([3, 0, 25, 6.76]).reshape(1, -1)))
```

[72]: array([0])

[73]:
```
# Class = 2, Female, Age: 43,  Fare:  12.88
dtree.predict((np.array([2, 0, 43, 12.88]).reshape(1, -1)))
```

[73]: array([1])

[74]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
print("Precision:",metrics.precision_score(y_test, y_pred))
```

```
Accuracy: 0.7552447552447552
Precision: 0.7164179104477612
```