In [1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import metrics
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np
```

In [2]:
```
parameters = pd.read_csv(r"D:\Detecting parkinsons disease\cleaned-data.csv")
parameters.head()
```

Out[2]:

	Unnamed: 0	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DI
0	0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.000070	0.00370	0.00554	0.011
1	1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.000080	0.00465	0.00576	0.013
2	2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.000090	0.00544	0.00576	0.016
3	3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.000090	0.00502	0.00576	0.015
4	4	phon_R01_S01_5	116.014	141.781	110.655	0.01101	0.000037	0.00593	0.00576	0.017

5 rows × 25 columns

In [3]:
```
columns = list(parameters.columns)
columns.remove('name')
columns
```

Out[3]:
```
['Unnamed: 0',
 'MDVP:Fo(Hz)',
 'MDVP:Fhi(Hz)',
 'MDVP:Flo(Hz)',
 'MDVP:Jitter(%)',
 'MDVP:Jitter(Abs)',
 'MDVP:RAP',
 'MDVP:PPQ',
 'Jitter:DDP',
 'MDVP:Shimmer',
 'MDVP:Shimmer(dB)',
 'Shimmer:APQ3',
 'Shimmer:APQ5',
 'MDVP:APQ',
 'Shimmer:DDA',
 'NHR',
 'HNR',
 'status',
 'RPDE',
 'DFA',
 'spread1',
 'spread2',
 'D2',
 'PPE']
```

In [4]:
```
feature_columns = columns
feature_columns.remove('status')
feature_columns
```

Out[4]:
```
['Unnamed: 0',
 'MDVP:Fo(Hz)',
 'MDVP:Fhi(Hz)',
 'MDVP:Flo(Hz)',
 'MDVP:Jitter(%)',
 'MDVP:Jitter(Abs)',
 'MDVP:RAP',
 'MDVP:PPQ',
 'Jitter:DDP',
 'MDVP:Shimmer',
 'MDVP:Shimmer(dB)',
 'Shimmer:APQ3',
 'Shimmer:APQ5',
 'MDVP:APQ',
 'Shimmer:DDA',
 'NHR',
 'HNR',
 'RPDE',
 'DFA',
 'spread1',
 'spread2',
 'D2',
 'PPE']
```

In [5]:
```
x = parameters[feature_columns]
y = parameters.status
x_train,x_test,y_train,y_test = train_test_split(x, y, test_size=0.3, random_state=1)
```

In [6]:
```
clf = svm.SVC(kernel='linear',probability=True)
clf.fit(x_train, y_train)
y_pred = clf.predict(x_test)
```

In [7]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.8305084745762712
```

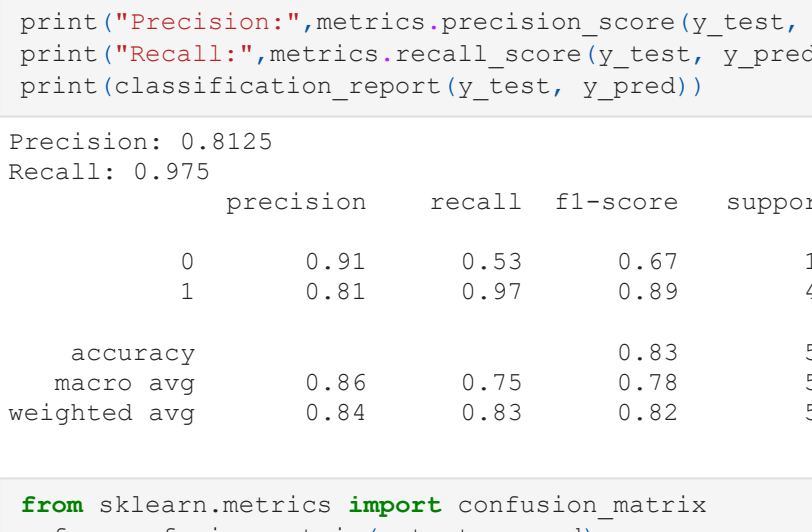
In [8]:
```
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Precision: 0.8125
Recall: 0.975
              precision    recall  f1-score   support

           0       0.91      0.53      0.67        19
           1       0.81      0.97      0.89        40

    accuracy                           0.83        59
   macro avg       0.86      0.75      0.78        59
weighted avg       0.84      0.83      0.82        59
```

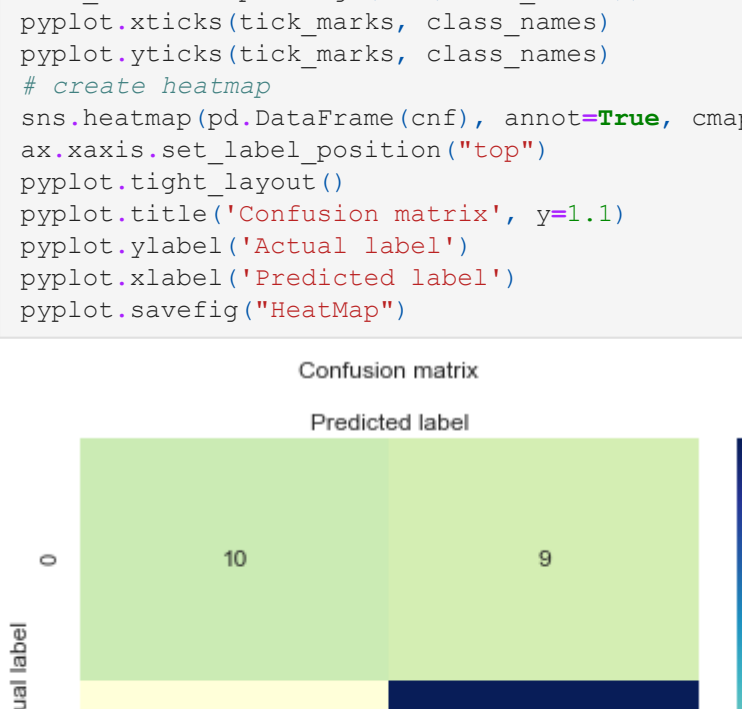
In [9]:
```
from sklearn.metrics import confusion_matrix
cnf = confusion_matrix(y_test,y_pred)
cnf
```

Out[9]:
```
array([[10,  9],
       [ 1, 39]], dtype=int64)
```

In [10]:
```
import seaborn as sns
import numpy as np
from matplotlib import pyplot
sns.set(style="white")
sns.set(style="whitegrid", color_codes=True)
class_names=[0,1] # name of classes
fig, ax = pyplot.subplots()
tick_marks = np.arange(len(class_names))
pyplot.xticks(tick_marks, class_names)
pyplot.yticks(tick_marks, class_names)
# create heatmap
sns.heatmap(pd.DataFrame(cnf), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
pyplot.tight_layout()
pyplot.title('Confusion matrix', y=1.1)
pyplot.ylabel('Actual label')
pyplot.xlabel('Predicted label')
pyplot.savefig("HeatMap")
```



In [11]:
```
y_pred_proba = clf.predict_proba(x_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test, y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
pyplot.plot(fpr,tpr,label="data 1, auc="+str(auc))
pyplot.legend(loc=4)
pyplot.show()
pyplot.savefig('ROC')
```



<Figure size 432x288 with 0 Axes>

In [ ]: