Expt no: 2 Implement of Logistic Regression

Coding:

```
#import pandas
import pandas as pd

col_names = ['pregnant', 'glucose', 'bp', 'skin', 'insulin', 'bmi', 'pedigree', 'age', 'label']

# load dataset
pima = pd.read_csv("data/diabetes.csv", header=1, names=col_names)


pima.head()
```

	pregnant	glucose	bp	skin	insulin	bmi	pedigree	age	label
0	1	85	66	29	0	26.6	0.351	31	0
1	8	183	64	0	0	23.3	0.672	32	1
2	1	89	66	23	94	28.1	0.167	21	0
3	0	137	40	35	168	43.1	2.288	33	1
4	5	116	74	0	0	25.6	0.201	30	0

5 rows

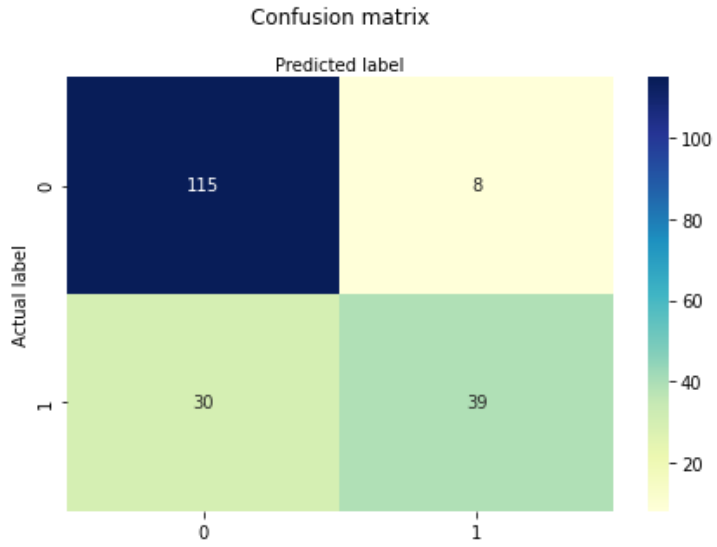
```
#split dataset in features and target variable
feature_cols = ['pregnant', 'insulin', 'bmi', 'age','glucose','bp','pedigree']
X = pima[feature_cols] # Features
y = pima.label # Target variable
# split X and y into training and testing sets
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=16)

X.head()
```

	pregnant	insulin	bmi	age	glucose	bp	pedigree
0	1	0	26.6	31	85	66	0.351
1	8	0	23.3	32	183	64	0.672
2	1	94	28.1	21	89	66	0.167
3	0	168	43.1	33	137	40	2.288
4	5	0	25.6	30	116	74	0.201

5 rows

```
# import the class
from sklearn.linear_model import LogisticRegression
```

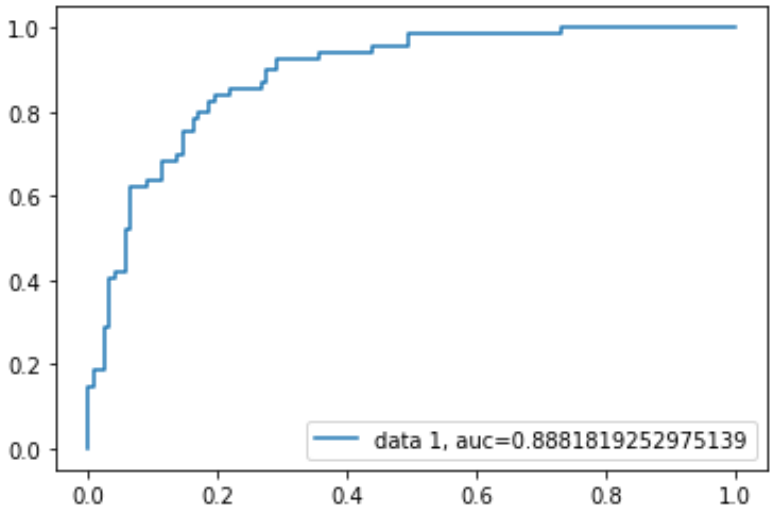
```
# instantiate the model (using the default parameters)
logreg = LogisticRegression(random_state=16)

# fit the model with data
logreg.fit(X_train, y_train)

y_pred = logreg.predict(X_test)


# import the metrics class
from sklearn import metrics

cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
array([[115,   8],
       [ 30,  39]])
# import required modules
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


class_names = [0,1] # name  of classes
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)

# create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label');
```




```
from sklearn.metrics import classification_report
target_names = ['without diabetes', 'with diabetes']
print(classification_report(y_test, y_pred, target_names=target_names))
```

	precision	recall	f1-score	support
without diabetes	0.79	0.93	0.86	123
with diabetes	0.83	0.57	0.67	69
accuracy			0.80	192
macro avg	0.81	0.75	0.77	192
weighted avg	0.81	0.80	0.79	192

```
y_pred_proba = logreg.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)

plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()
```


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
data 1, auc=0.8881819252975139