In [1]:
```
"""
ECGR 5105 - Intro to Machine Learning
Homework 2 - Part 1
Phillip Harmon
"""
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:
```
#Import Dataset
csvData = pd.read_csv('diabetes.csv')
csvData
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunctic
0	6	148	72	35	0	33.6	0.6
1	1	85	66	29	0	26.6	0.3
2	8	183	64	0	0	23.3	0.6
3	1	89	66	23	94	28.1	0.1
4	0	137	40	35	168	43.1	2.2
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.1
764	2	122	70	27	0	36.8	0.3
765	5	121	72	23	112	26.2	0.2
766	1	126	60	0	0	30.1	0.3
767	1	93	70	31	0	30.4	0.3

768 rows × 9 columns

In [3]:
```
#Sort Dataset
x = csvData.iloc[:,0:-1].values
y = csvData.iloc[:,-1].values
```

In [4]:
```
#Train-Test Split
from sklearn.model_selection import train_test_split
xt, xv, yt, yv = train_test_split(x, y, train_size = 0.8, test_size = 0.2, ran
```

In [5]:
```
#Clean the Dataset
from sklearn.preprocessing import MinMaxScaler, StandardScaler
 # scaler = StandardScaler() #MinMaxScaler gave better results here
scaler = MinMaxScaler()
xt = scaler.fit_transform(xt)
xv = scaler.fit_transform(xv)
```

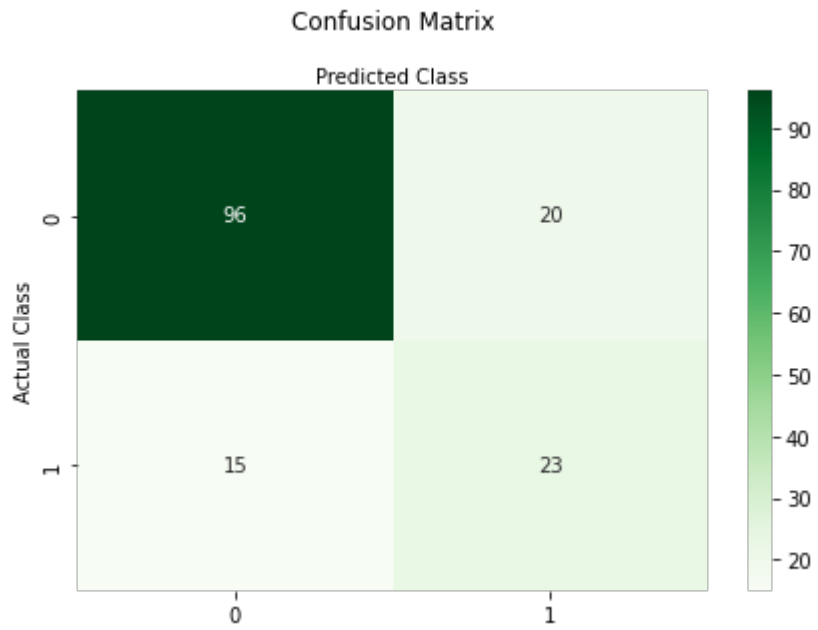
In [6]:
```
#Perform the Training
from sklearn.linear_model import LogisticRegression
training_montage = LogisticRegression(random_state=1337)
training_montage.fit(xt,yt);
```

In [7]:
```
#Test the Model
p = training_montage.predict(xv)
```

In [8]:
```
#Evaluate the model metrics
from sklearn import metrics
print("Model Accuracy:  {:.3f}%".format(metrics.accuracy_score(yv,p)*100))
print("Model Precision: {:.3f}%".format(metrics.precision_score(yv,p)*100))
print("Model Recall:    {:.3f}%".format(metrics.recall_score(yv,p)*100))
```

```
Model Accuracy:  77.273%
Model Precision: 53.488%
Model Recall:    60.526%
```

In [9]:
```
#Analyze using the Confusion Matrix
from sklearn.metrics import confusion_matrix
import seaborn as sns
classes = ['Not Diabetes','Diabetes']
figure, axis = plt.subplots()
ticks = np.arange(len(classes))
plt.xticks(ticks, classes)
plt.yticks(ticks, classes)
sns.heatmap(pd.DataFrame(confusion_matrix(yv, p)), annot=True, cmap="Greens",
axis.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion Matrix', y=1.1)
plt.ylabel('Actual Class')
plt.xlabel('Predicted Class');
```



In [ ]: