Exercise 10: A spam filter using SVM:

UCI dataset: Spambase D6: https://archive.ics.uci.edu/ml/datasets/Spambase Build a spam filter using a pre-processed dataset. A spam filter to classify an email to be Ham or Spam, using the content of an email as features. Build a basic spam filter using SVM. You have to use libsvm https://github.com/cjlin1/libsvm/tree/master/python. libsvm accepts data in a libsvm format. Each data row in a libsvm format is given as

< label >< index1 >:< value1 >< index2 >:< value2 >...

Convert dataset D3 into a libsvm format. Follow the readme document given on the libsvm link to see how you can use it to solve your problem. You have to learn a spam classifier on train part of the dataset and evaluate it on test dataset. Also optimize the hyper parameter i.e. value of C. [hint: when choosing the range of hyperparameter its always useful to check a diverse range i.e. C ={1,2,3,4} is not a good range to check for optimal value, you might want to check a broader range going from 0.1 to 100 etc.]. Present your results in form of graphs and tables, listing details. You have to choose a quality criterion according to the given problem i.e. classification

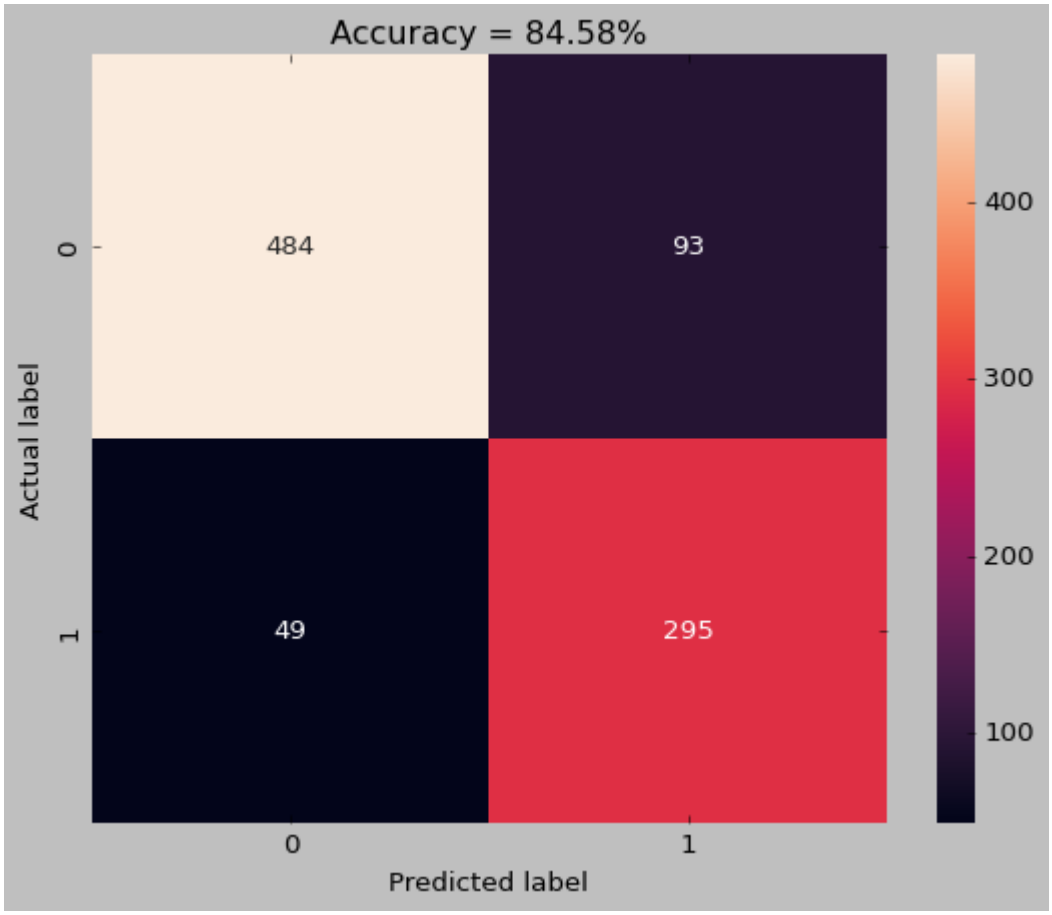
In [ ]:
```
import pandas as pd
from libsvm.svmutil import *
from sklearn.model_selection import train_test_split
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('classic')
import seaborn as sns
from sklearn import metrics
# Input: Dataset
data = pd.read_csv("spambase.data", sep=',', header= None)
# Separating the data into independent and dependent variables
X = data.iloc[:,:-1]
Y= data.iloc[:,-1].values
# Splitting the data into training and testing data
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2)
X_train = X_train.to_numpy()
X_test = X_test.to_numpy()
m = svm_train(Y_train, X_train, '-c 4')
p_label, p_acc, p_val = svm_predict(Y_test, X_test, m)
cm = metrics.confusion_matrix(Y_test, p_label)
```

```
....*..*
optimization finished, #iter = 6348
nu = 0.233709
obj = -2069.534770, rho = -0.147558
nSV = 2159, nBSV = 291
Total nSV = 2159
Accuracy = 84.582% (779/921) (classification)
```

Output: The Confuse Matrix and Classification Report

In [ ]:
```
cm_df = pd.DataFrame(cm)
sns.heatmap(cm_df, annot=True, fmt='g')
```

```
plt.title('Accuracy = {0:.2f}%'.format(p_acc[0]))
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
print("\nThe Classification Report for the Spam Filter\n\n" ,metrics.clas
```




```
The Classification Report for the Spam Filter

              precision    recall  f1-score   support

           0       0.91      0.84      0.87       577
           1       0.76      0.86      0.81       344

    accuracy                           0.85       921
   macro avg       0.83      0.85      0.84       921
weighted avg       0.85      0.85      0.85       921
```