**CA02: Spam Email Detection Using Naïve Bayes Classification Algorithm**

*Vania Revelina*

* Link to Github Repository: <https://github.com/vrevelina/BSAN6070-CA-Revelina/tree/master/CA02>
* Link to Google Drive – Colab folder share
* Results

# import necessary packages

import os

import numpy as np

from collections import Counter

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score

def MakeCounterList(root\_dir):

  """Extracts 3000 most repeated words

  Creates a list of every word in every email,

  takes only alphabetical words with more than 1 character,

  and return 3000 words which occured the most.

  Parameter:

    root\_dir (str): path to a folder which contains the emails

  Returns:

    word\_count\_list (list): a list of tuples containing the 3000 most common words

                            with its corresponding number of occurences

  """

  all\_words = []

  # create a list containing all the path to each of the text file (.txt)

# containing an email

  emails = [os.path.join(root\_dir,f) for f in os.listdir(root\_dir)]

  for mail in emails: # for every path in the 'emails' list

    with open(mail) as m: # open the file at the path selected

      for line in m: # for every line in the file

        words\_list = line.split() # split the line into a list of words

        # put these lists of words into 1 list of all the words in the email,

# we ignore empty lists

        # note: the elements of all\_words are only words NOT LISTS

        all\_words += words\_list

  # counter is a subclass of dictionaries, it counts the number of occurences

# in iterables

  # sample output = Counter({'eggs':2,'milk':3}) meaning the word 'eggs' was

# repeated 2 times, 'milk' was repeated 3 times

  # create a counter subclass containing the number of occurences of each

# word in all\_words list

  word\_count\_list = Counter(all\_words)

  # when you make a dict to a list, it'll only take the keys, not the values

# so words\_to\_remove only contains the list of all words in the body

# of the email.

  words\_to\_remove = list(word\_count\_list)

  for word in words\_to\_remove: # for every word in the words\_to\_remove list

    if word.isalpha() == False: # if word have non-alphabetical characters

      del word\_count\_list[word] # delete the word from the counter dictionary

    elif len(word) == 1: # if word ONLY CONTAINS alphabetical characters AND

# only has 1 character

      del word\_count\_list[word] # delete the word from the counter dictionary

    # otherwise, leave it in the counter dictionary

  # only take the 3000 words with the most occurences in the

# counter dictionary

  # then turn it into a list of tuples ('word',# of occurence)

  word\_count\_list = word\_count\_list.most\_common(3000)

  # return word counter list

  return word\_count\_list

def extract\_features(mail\_dir):

  """Marks most common word occurence and spam emails

  Checks the number of occurences of most common words in each email and

marks it in the features\_matrix

  Checks if the name of the text file containing the email indicates that it

is a spam email

  Parameter:

    mail\_dir (str): path to a folder containing all emails

  Returns:

    features\_matrix (np.array): a matrix containing the number of occurences

of each most common word in each email

    email\_labels (np.array): a matrix indicating which emails are spam (1)

and not spam (0)

  """

  # create a list of all the path to every email in the folder

  files = [os.path.join(mail\_dir,fi) for fi in os.listdir(mail\_dir)]

  # create a numpy array of zeros to put the number of occurences of

  # the 3000 most common words in each of our email.

  features\_matrix = np.zeros((len(files),3000))

  # create an empty array to put our labels (spam/not spam) in

  email\_labels = np.zeros(len(files))

  # fileNUM indicates the order of the file/email we are in

  fileNUM = 0

  for file\_ in files: # for every path in the files list

    with open(file\_) as f: # open the file at the path selected

      for i, line in enumerate(f):

        # i is the order number of the line inside the text file

        # example: subject line is line 0 in the text file

        # so i=0 for all subject lines

        # i=1 is an empty line

        # i=2 is the line containing the body of the email

        if i == 2: # if it's the body of the email

          words = line.split() # create a list of all the words in the body

          for word in words: # for every word in the body

            wordID = 0

            # REMINDER: words\_used\_dict is a list of tuples of the 3000 most

# common words

            for j, wtup in enumerate(common\_words\_list): # for j=index and

wtup=tuple of (word, number of occurence) in words\_used\_dict

              if wtup[0]==word: # if the most common word selected is the

# same as the word in the body selected

                wordID = j # set wordID to be the index of the most

# common word

                features\_matrix[fileNUM,wordID] = words.count(word) # mark

# the (count of occurences of the most common word in

# the email body) in the features matrix

      filepathTokens = file\_.split('/')

      # REMINDER: the name of the file that starts with "spmsg" indicates

# that the email is a spam

      lastToken = filepathTokens[-1] # get only the name of the

# file (without the path)

      if lastToken.startswith("spmsg"): # if email is a spam

        email\_labels[fileNUM] = 1 # set label to 1

      fileNUM = fileNUM + 1 # go to the next file

  return features\_matrix, email\_labels

"""The section is the main Program that calls the above two functions and gets executed first. First it "trains" the model using model.fit function and Training Dataset. After that it scores the Test Data set by running the Trained Model with the Test Data set. At the end it prints the model performance in terms of accuracy score."""

# specify the train and test directory/path

TRAIN\_DIR = '/content/drive/My Drive/MSBA\_Colab\_2020/ML\_Algorithms/CA02/Data/train-mails'

TEST\_DIR = '/content/drive/My Drive/MSBA\_Colab\_2020/ML\_Algorithms/CA02/Data/test-mails'

# create a list of the most common words and the number of occurences

common\_words\_list = MakeCounterList(TRAIN\_DIR)

print ("reading and processing emails from TRAIN and TEST folders")

# create a features matrix and labels for train and test emails sets

features\_matrix, train\_labels = extract\_features(TRAIN\_DIR)

test\_features\_matrix, test\_labels = extract\_features(TEST\_DIR)

# instantiate the Naive Bayes Classification Model

model = GaussianNB()

print ("Training Model using Gaussian Naibe Bayes algorithm .....")

model.fit(features\_matrix, train\_labels) # train model

print ("Training completed")

print ("testing trained model to predict Test Data labels")

predicted\_labels = model.predict(test\_features\_matrix) # test model

print ("Completed classification of the Test Data .... now printing Accuracy Score by comparing the Predicted Labels with the Test Labels:")

print (accuracy\_score(test\_labels, predicted\_labels)) # calculate the accuracy of the model

"""======================= END OF PROGRAM ========================="""

**OUTPUT**

Reading and processing emails from TRAIN and TEST folders

Training Model using Gaussian Naibe Bayes algorithm .....

Training completed

testing trained model to predict Test Data labels

Completed classification of the Test Data .... now printing Accuracy Score by comparing the Predicted Labels with the Test Labels:

0.9653846153846154

'======================= END OF PROGRAM ========================='