Name: Parthvi Shah NetID: pss434

In [ ]:
```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
```

Question 1:

In [ ]:
```
train_data = pd.read_csv('sample_data/spam_train (1).txt', header= None)
test_data = pd.read_csv('sample_data/spam_test.txt', header= None)
```

In [ ]:
```
trainn = train_data.iloc[:,0]
label = trainn.str.split(' ').str[0]
email = trainn.str.split(' ').str[1:]
```

In [33]:
```
train_size = int(len(train_data)*0.8)
X_train = email[:train_size]
X_val = email[train_size:]
Y_train = label[:train_size]
Y_val = label[train_size:]
X_train.shape
```

Out[33]:
```
(4000,)
```

In [ ]:
```
test = test_data.iloc[:,0]
label_test = test.str.split(' ').str[0]
email_test = test.str.split(' ').str[1:]
```

If we had not created Validation Set, Our model could risk overfitting. We need to hypertune the parameters, in this case, the number of iterations. After finding the best model and itertions, we will fit our model to the test.

Question 2:

In [ ]:
```
def preprocess(data):
  vocab = {}
  data = data.to_dict()

  for key,values in data.items():
      temp = set()
      for v in values:
          temp.add(v)
      for t in temp:
          if t in vocab.keys():
              vocab[t] += 1
          else:
              vocab[t] = 1
      temp.clear()
  for key,value in list(vocab.items()):
    if (vocab[key] < 30):
      vocab.pop(key)
  return vocab
```

In [ ]:
```
vocab_train = preprocess(X_train)
```

In [ ]:

```
def featurevector1(data, vocab):
  vocablist = list(vocab)
  sorted_items = sorted(vocablist)
  data = data.to_dict()
  results = []
  for item in data.values():
    featurevector = [0 for i in range(0, len(vocablist))]
    for i in item:
      if i in vocablist:
        one_hot_index = sorted_items.index(i)
        featurevector[one_hot_index] = 1
    results.append(featurevector)
  return np.asarray(results)
```

In [ ]:

```
fvarr_train = featurevector1(X_train, vocab_train)
```

Question 3:

In [ ]:

```
def perceptron_train(data, fvarr, vocab):
  data = data.astype(int)
  data  = list(data.replace([0,1],[-1,1]))
  vocablist = list(vocab)
  w = np.zeros(len(fvarr[0]))
  iteration = 0
  mistake = 0
  All_True = False
  while not All_True:
    iteration += 1
    k =0
    for i,x in enumerate(fvarr):
      if (int(np.dot(fvarr[i], w)*float(data[i]))) <= 0:
        k += 1
        w += np.dot(data[i], fvarr[i])
        mistake += 1
    if k==1:
      All_True = True
    else:
      All_True = False

  return w,mistake,iteration
```

In [ ]:

```
def perceptron_test(w, vector, label, n):

  label = label.astype(int)
  label  = list(label.replace([0,1],[-1,1]))
  errors = []
  error = 0
  for i,x in enumerate(vector):
    if (int(np.dot(vector[i], w)*float(label[i]))) < 0:
      error += 1
  fraction = error/len(vector)


  return fraction,error
```

Question 4:

In [ ]:

```
w,k,iteration = perceptron_train(Y_train, fvarr_train, vocab_train)
fraction,error = perceptron_test(w, fvarr_train, Y_train, iteration)
```

In [48]:

```
print('The number of mistakes in training set:',k,'& the number of iterations to converge:'
,iteration)
```

```
The number of mistakes in training set: 448 & the number of iterations to converge: 11
```

In [49]:

```
print('Fraction of error with training set', fraction)
```

```
Fraction of error with training set 0.0
```

In [ ]:

```
vocab_val = preprocess(X_val)
fvarr_val = featurevector1(X_val,vocab_train)
```

In [ ]:

```
fraction,errors = perceptron_test(w, fvarr_val, Y_val, iteration)
```

In [52]:

```
print('Fraction of error with validation set', fraction)
```

```
Fraction of error with validation set 0.013
```

Question 5:

In [ ]:

```
vtk = list(sorted(vocab_train.keys()))
wl = list(w)
s = dict(zip(vtk, wl))
```

In [ ]:

```
sorted_x = sorted(s.items(), key=lambda kv: kv[1], reverse=False)
```

In [ ]:

```
mostnegative = []
mostpositive = []
for i in range(1,16):
  mostnegative.append(sorted_x[i])
  mostpositive.append(sorted_x[-i])
```

In [56]:

```
mostnegative , mostpositive
```

Out[56]:

```
([('reserv', -15.0),
  ('prefer', -14.0),
  ('copyright', -13.0),
  ('i', -12.0),
  ('still', -12.0),
  ('technolog', -12.0),
  ('but', -11.0),
  ('comput', -11.0),
  ('recipi', -11.0),
  ('someth', -11.0),
  ('which', -11.0),
  ('coupl', -10.0),
  ('date', -10.0),
  ('url', -10.0),
  ('execut', -9.0)],
 [('sight', 22.0),
  ('click', 18.0),
  ('these', 16.0),
  ('remov', 16.0),
  ('market', 16.0),
  ('our', 15.0),
  ('deathtospamdeathtospamdeathtospam', 14.0),
  ('most', 13.0),
  ('yourself', 12.0),
  ('present', 12.0),
  ('parti', 12.0),
  ('ever', 12.0),
  ('pleas', 11.0),
  ('guarante', 11.0),
  ('check', 11.0)])
```

Question 6:

In [ ]:

```
def average_perceptron(data, fvarr, vocab):
  data = data.astype(int)
  data  = list(data.replace([0,1],[-1,1]))
  vocablist = list(vocab)
  w = np.zeros(len(fvarr[0]))
  #Number of mistakes
  k = 0
  mistake = 0
  #Number of passes through the data
  iteration = 0
  All_True = False
  weightstotal = 0
  while not All_True:
    iteration += 1
    k =0
    for i,x in enumerate(fvarr):
      if (int(np.dot(fvarr[i], w)*float(data[i]))) <= 0:
        k += 1
        w += np.dot(data[i], fvarr[i])
        weightstotal+= w
        mistake += 1

      else:
        w=w
        weightstotal += w

    if k==1:
      All_True = True
    else:
      All_True = False
  w = weightstotal/(iteration*len(fvarr))
  return w,mistake,iteration
```

In [ ]:

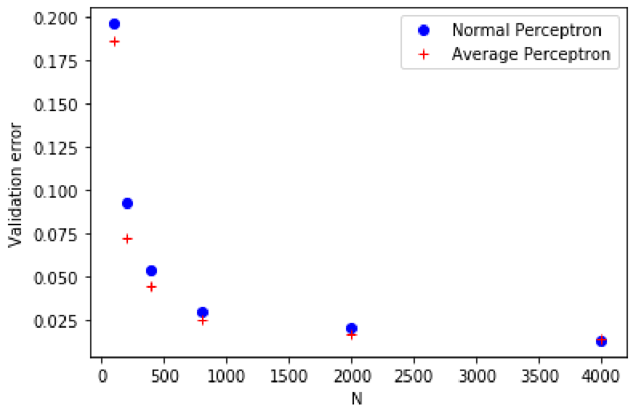
```
w,k,iteration = average_perceptron(Y_train, fvarr_train, vocab_train)
```

Question 7:

In [ ]:

```
N = [100, 200, 400, 800, 2000, 4000]
```

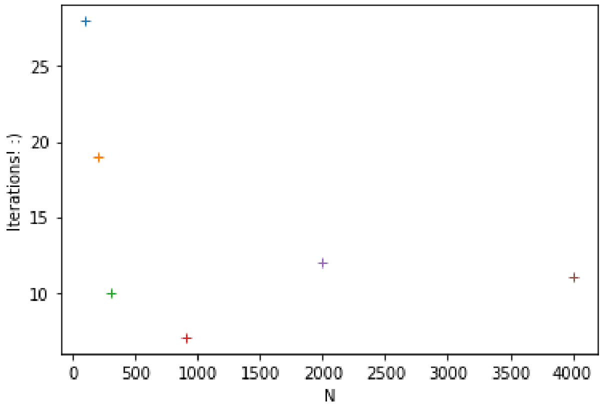
In [64]:
```
for i in N:

  vocab_trainn1 = preprocess(X_train[:i])
  fvarr_trainn1 = featurevector1(X_train[:i], vocab_trainn1)
  w,k,iteration = perceptron_train(Y_train[:i], fvarr_trainn1[:i], vocab_trainn1)
  wa,ka,iterationa = average_perceptron(Y_train[:i], fvarr_trainn1[:i], vocab_trainn1)
  fvarr_val = featurevector1(X_val,vocab_trainn1)

  fractionnormal,errorsnormal = perceptron_test(w, fvarr_val, Y_val, iteration)
  fractionaverage,errorsaveraged = perceptron_test(wa, fvarr_val, Y_val, iterationa)

  plt.plot(i,fractionnormal, 'bo', color='blue', label="Normal Perceptron" if i == 100 else
'')

  plt.plot(i, fractionaverage, 'r+', color='red', label = 'Average Perceptron' if i == 100
else '')

plt.xlabel('N')
plt.ylabel('Validation error')
plt.legend()
plt.show()
```



Question 8:

In [ ]:
```
N = [100, 200,400,800,2000, 4000]
```

In [90]:
```
for i in N:
  vocab_trainn1 = preprocess(X_train[:i])
  fvarr_trainn1 = featurevector1(X_train[:i], vocab_trainn1)
  w,k,iteration = perceptron_train(Y_train[:i], fvarr_trainn1, vocab_trainn1)
  plt.plot(i,iteration, marker = '+', linestyle = '', label='0')
plt.xlabel('N')
plt.ylabel('Iterations! :)')
plt.show()
```



In [ ]:

```
def tryperceptron_train(data, fvarr, vocab, iteration):
  data = data.astype(int)
  data  = list(data.replace([0,1],[-1,1]))
  vocablist = list(vocab)
  w = np.zeros(len(fvarr[0]))
  itera = 0
  mistake = 0
  All_True = False
  while not All_True and itera < iteration :
    itera += 1
    k =0
    for i,x in enumerate(fvarr):
      if (int(np.dot(fvarr[i], w)*float(data[i]))) <= 0:
        k += 1
        w += np.dot(data[i], fvarr[i])
        mistake += 1
    if k==0:
      All_True = True
    else:
      All_True = False

  return w,mistake,iteration
```

Question 9:

In [ ]:

```
def tryaverage_perceptron(data, fvarr, vocab, iteration):
  data = data.astype(int)
  data  = list(data.replace([0,1],[-1,1]))
  vocablist = list(vocab)
  w = np.zeros(len(fvarr[0]))
  #Number of mistakes
  k = 0
  itera = 0
  #Number of passes through the data
  All_True = False
  weightstotal = 0
  while not All_True and itera < iteration:
      itera += 1
      for i,x in enumerate(fvarr):
        if (np.dot(fvarr[i] ,w)*data[i]) > 0:
          w = w
          weightstotal += w
        else:
          k += 1
          w += np.dot(data[i], fvarr[i])
          weightstotal+= w
      if k==0:
        All_True = True
      else:
        All_True = False
  w = weightstotal/(itera*len(fvarr))
  return w,k,iteration
```

In [93]:

```
N = [10,15,20,50,100]
for i in N:
  wa,ka,iterationa = tryaverage_perceptron(Y_train, fvarr_train, vocab_train, i)
  fractiontrya,errortrya = perceptron_test(wa, fvarr_val, Y_val, iteration)
  print('Validation error on Average peceptron with iteration',iterationa,'is:',fractiontry
a)
```

```
Validation error on Average peceptron with iteration 10 is: 0.015
Validation error on Average peceptron with iteration 15 is: 0.011
Validation error on Average peceptron with iteration 20 is: 0.012
Validation error on Average peceptron with iteration 50 is: 0.012
Validation error on Average peceptron with iteration 100 is: 0.012
```

```
In [94]: for i in N:
           wa,ka,iterationa = tryperceptron_train(Y_train, fvarr_train, vocab_train, i)
           fractiontrya,errortrya = perceptron_test(wa, fvarr_val, Y_val, iteration)
           print('Validation error on Normal peceptron with iteration',iterationa,'is:',fractiontrya
         )
```

```
Validation error on Normal peceptron with iteration 10 is: 0.013
Validation error on Normal peceptron with iteration 15 is: 0.013
Validation error on Normal peceptron with iteration 20 is: 0.013
Validation error on Normal peceptron with iteration 50 is: 0.013
Validation error on Normal peceptron with iteration 100 is: 0.013
```

So According to me, With the right number of iterations, both the algorithm work similarly at least for this data set! But I believe normal perceptron works a little better since it uses lesser number of iterations to get such a less validation error. Average perceptron works better when the number of iterations is 15.

I would use average perceptron with 15 iterations as that gives the least validation error.

Question 11:

```
In [ ]: vocab_train = preprocess(email)
        fvarr_train = featurevector1(email, vocab_train)
        w,k,iteration = tryaverage_perceptron(label, fvarr_train, vocab_train, 15)
        fvarr_test = featurevector1(email_test, vocab_train)
        fraction,error = perceptron_test(w, fvarr_test, label_test, iteration)
```

```
In [96]: print('Test set error on Average peceptron with 15 iterations is ',fraction)
```

```
Test set error on Average peceptron with 15 iterations is  0.016
```

Question 10: Optional

```
In [ ]: def tryingxpreprocess(data):
          vocab = {}
          data = data.to_dict()

          for key,values in data.items():
              temp = set()
              for v in values:
                  temp.add(v)
              for t in temp:
                  if t in vocab.keys():
                      vocab[t] += 1
                  else:
                      vocab[t] = 1
              temp.clear()
          for key,value in list(vocab.items()):
            if (vocab[key] < 15):
              vocab.pop(key)
          return vocab
```

```
In [ ]: vocab_train = tryingxpreprocess(email)
        fvarr_train = featurevector1(email, vocab_train)
        w,k,iteration = tryaverage_perceptron(label, fvarr_train, vocab_train, 11)
        fvarr_test = featurevector1(email_test, vocab_train)
        fraction,error = perceptron_test(w, fvarr_test, label_test, iteration)
```

```
In [99]: print('Test set error on changong X and applying Average peceptron with 11 iterations is ',
         fraction)
```

```
Test set error on changong X and applying Average peceptron with 11 iterations is  0.018
```

If I change my X from 30 to 10, I see an increase in the error rate.