1. import os

2. import random

3. import string

4. import nltk

5. nltk.download('punkt')

6. nltk.download('stopwords')

7. from nltk import word\_tokenize

8. from collections import defaultdict

9. from nltk import FreqDist

10. from nltk.corpus import stopwords

11. from sklearn.feature\_extraction.text import TfidfVectorizer

12. from sklearn.feature\_extraction.text import CountVectorizer

13. from sklearn.naive\_bayes import MultinomialNB

14. from sklearn import metrics

15. import pickle

16. stop\_words = set(stopwords.words('english'))

17. stop\_words.add('said')

18. stop\_words.add('mr')

19. BASE\_DIR = 'D:\\MYLEARNING\\THE\_JOURNEY\_IV\\COMPUTER\_SCIENCE\_PROJECT\_2\\PRACTICE\\bbc'

20. LABELS = ['business', 'entertainment', 'politics', 'sport', 'tech']

21. # def create\_data\_set():

22. # with open('data.txt', 'w', encoding='utf8') as outfile:

23. # for label in LABELS:

24. # dir = '%s/%s' % (BASE\_DIR, label)

25. # for filename in os.listdir(dir):

26. # fullfilename = '%s/%s' % (dir, filename)

27. # print(fullfilename)

28. # with open(fullfilename, 'rb') as file:

29. # text = file.read().decode(errors= 'replace').replace('\n', '')

30. # outfile.write('%s\t%s\t%s\n' % (label, filename, text))

31.

32. def setup\_docs():

33. docs = []

34. with open('data.txt', 'r', encoding='utf8') as datafile:

35. for row in datafile:

36. parts = row.split('\t')

37. doc = ( parts[0], parts[2].strip() )

38. docs.append(doc)

39. return docs

40.

41. def clean\_text(text):

42. text = text.translate(str.maketrans('', '', string.punctuation))

43. text = text.lower()

44. return text

45.

46.

47.

48. def get\_tokens(text):

49. tokens = word\_tokenize(text)

50. tokens = [t for t in tokens if not t in stop\_words]

51. return tokens

52.

53.

54. def print\_frequency\_dist(docs):

55. tokens = defaultdict(list)

56.

57. for doc in docs:

58. doc\_label = doc[0]

59. #doc\_text = doc[1]

60. doc\_text = clean\_text(doc[1])

61. #doc\_tokens = word\_tokenize(doc\_text)

62. doc\_tokens =get\_tokens(doc\_text)

63. tokens[doc\_label].extend(doc\_tokens)

64. for category\_label, category\_tokens in tokens.items():

65. print(category\_label)

66. fd = FreqDist(category\_tokens)

67. print(fd.most\_common(20))

68.

69. def get\_splits(docs):

70.

71. random.shuffle(docs)

72.

73. X\_train = []# training documents

74. y\_train = []# corresponding training labels

75. X\_test = []# test documents

76. y\_test = []# corresponding test labels

77. pivot = int(.80 \* len(docs))

78. for i in range(0, pivot):

79. X\_train.append(docs[i][1])

80. y\_train.append(docs[i][0])

81. for i in range(pivot, len(docs)):

82. X\_test.append(docs[i][1])

83. y\_test.append(docs[i][0])

84. return X\_train, X\_test, y\_train, y\_test

85.

86. def evaluate\_classifier(title, classifier, vectorizer, X\_test, y\_test):

87. X\_test\_tfidf = vectorizer.transform(X\_test)

88. y\_pred = classifier.predict(X\_test\_tfidf)

89.

90. precision = metrics.precision\_score(y\_test, y\_pred, average='macro')

91. recall = metrics.recall\_score(y\_test, y\_pred, average='macro')

92. f1 = metrics.f1\_score(y\_test, y\_pred, average='macro')

93.

94. print("%s\t%f\t%f\t%f\n" % (title, precision, recall, f1))

96. def train\_classifier(docs):

97. X\_train, X\_test, y\_train, y\_test = get\_splits(docs)

98.

99. vectorizer = CountVectorizer(stop\_words='english', ngram\_range=(1, 3), min\_df=3, analyzer='word')

100.

101. dtm = vectorizer.fit\_transform(X\_train)

102.

103. naive\_bayes\_classifier = MultinomialNB().fit(dtm, y\_train)

104.

105.

106. evaluate\_classifier("Naive Bayes\tTRAIN\t", naive\_bayes\_classifier, vectorizer, X\_train, y\_train)

107.

108. evaluate\_classifier("Naive Bayes\tTEST\t", naive\_bayes\_classifier, vectorizer, X\_test, y\_test)

109.

110.

111. clf\_filename = 'naive\_bayes\_classifier.pkl'

112. pickle.dump(naive\_bayes\_classifier, open(clf\_filename, 'wb'))

113.

114.

115. vec\_filename = 'count\_vectorizer.pkl'

116. pickle.dump(vectorizer, open(vec\_filename, 'wb'))

117.

118. def classify(text):

119.

120. clf\_filename = 'D:\\MYLEARNING\\THE\_JOURNEY\_IV\\COMPUTER\_SCIENCE\_PROJECT\_2\\naive\_bayes\_classifier.pkl'

121. nb\_clf = pickle.load(open(clf\_filename, 'rb'))

122.

123.

124. vec\_filename = 'D:\\MYLEARNING\\THE\_JOURNEY\_IV\\COMPUTER\_SCIENCE\_PROJECT\_2\\count\_vectorizer.pkl'

125. vectorizer = pickle.load(open(vec\_filename, 'rb'))

126.

127.

128. pred = nb\_clf.predict(vectorizer.transform([text]))

129.

130. print(f"\n\nThe topic classified for the text is: {str(pred[0]).upper()}\n\n")

131. if \_\_name\_\_ == '\_\_main\_\_':

132. create\_data\_set()

133. docs = setup\_docs()

134. print\_frequency\_dist(docs)

135. train\_classifier(docs)

136. new\_doc = """ It’s a good day for Apple deals. Not only can you get

137. $150 off the Apple Vision Pro for the first time, but the new MacBook Air

138. , released earlier this month, is already $100 off at Amazon and B&H Photo.

139. That’s a new all-time low available on the midnight base models, which

140. come with 8GB of memory and 256GB of storage. Note that Amazon is temporarily

141. out of stock, but you can still place an order, and it will ship when it’s available."""

142. train\_classifier(docs)

143. classify(new\_doc)

144. print("Done")