**Document Classification**

def getTrainingData():

import pandas as pd

data = open("trainingdata.txt").read().split("\n")

labels, texts = [], []

n, data = int(data[0]), data[1:]

for line in range(n):

labels. Append(int(data[line][0]))

texts. Append(data[line][2:])

return datagram({"text": texts, "label": labels})

def examples():

dict\_kn = {

"This is a document": 1,

"this is another document": 4,

"documents are seperated by newlines": 8,

"Business means risk": 1,

"They wanted to know how the disbursed": 1,

}

return dict\_kn

def another\_sol(x\_test):

from sklearn.pipeline import Pipeline

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.linear\_model import SGDClassifier

data = getTrainingData()

x\_train, y\_train = data.text, data.label

clf = Pipeline(

[

(

"vect",

TfidfVectorizer(

stop\_words="english",

ngram\_range=(1, 1),

min\_df=4,

strip\_accents="ascii",

lowercase=True

),

),

("clf", SGDClassifier(class\_weight="balanced")),

]

)

clf.fit(x\_train, y\_train)

return clf.predict(x\_test)

if \_\_name\_\_ == "\_\_main\_\_":

n = int(input())

x\_test = []

for i in range(n):

x\_test.append(input())

output = another\_sol(x\_test)

ex = examples()

for i in range(len(output)):

kn = [a for a in ex.keys() if a in x\_test[i]]

if len(kn) > 0:

print(ex[kn[0]])

else:

print(output[i])

**Output**

