The user is able to consider special characters accepted as words:

**Expected result**: The user should not be consider with special characters

1. In this code the user is able to consider as single word without word break

import string  
from nltk import FreqDist  
from nltk import word\_tokenize  
from nltk.corpus import stopwords

text= \*input text\*

new\_string = text.translate(str.maketrans('', '', string.punctuation))  
stop\_list = set(stopwords.words('english') + list(new\_string))  
  
tokens = [token for token in word\_tokenize(new\_string) if token.lower() not in stop\_list]  
words = FreqDist(tokens).most\_common(10)  
print(words)

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| “cat-walk,,,” “cat-walk” @cat-walk “cat-@walk@” <<cat-walk>> ‘cat-walk’ | (‘catwalk’,6) |
| 50cror?e 50cror?e 50cror?e 50cror?e | (‘50crore’,4) |
| <ha@rsha> ha@rsha “ha@rsha” “ha@rsha” | (‘harsha’,4) |
| One!!two one!!two one!!two | (‘onetwo’,3) |
| ten’‘lakh ten’’lakh ten’’lakh | (‘tenlakh’,3) |
| [a@h4-a@] [ah@4-a@] [ah@4-a@] | (‘ah4a’,3) |
| [side’’by] [side’’by] [side’’by] | (‘sideby’,3) |
| Six{million} six{million} six{million} | (‘sixmillion’,3) |
| one;billion one;billion one;billion | (‘onebillion’,3) |
| hundred:rupees hundred:rupees | (‘hundredrupees’,2) |
| {per%centage} {per%centage} | (‘percentage’,2) |

**Description**:

In the above code I used **str.translate()** method.The **translate()** method typically takes a translation table,which we’ll do using the **.maketrans()** method

The **maketrans()** method here takes three arguments, the first two of which are empty strings and the third is the list of punctuation we want to remove. This tells the function to replace all punctuation with none.

2.In this code the user is able to consider with word break frequency

import string  
import re  
from nltk import FreqDist  
from nltk import word\_tokenize  
from nltk.corpus import stopwords

text = \*input text\*

stop\_words = set(stopwords.words("english"))  
  
specials\_chars = re.escape(string.punctuation)  
all\_words = [token for token in word\_tokenize(text)]  
all\_words = list(filter(lambda word: word, map(lambda word: re.sub(r'[' + specials\_chars + ']', '', word), all\_words)))  
  
filtered\_words = [word.lower() for word in all\_words if word.casefold() not in stop\_words]  
  
filtered\_words\_count = len(filtered\_words)  
  
common\_words = FreqDist(filtered\_words).most\_common(20)  
  
print(common\_words)

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| “cat@walk” “cat@walk” “cat@walk” “cat@walk” “cat@walk” “cat@walk” | (‘cat’,6) (‘walk’.6) |
| 50cror?e 50cror?e 50cror?e 50cror?e | (‘50cror’,4) (‘e’,4) |
| <ha@rsha> ha@rsha “ha@rsha” “ha@rsha” | (‘ha’,4) (‘rsha’,4) |
| one!!two one!!two one!!two | (‘one’,3)(‘two’,3) |
| ten’‘lakh ten’’lakh ten’’lakh | (‘ten’,3)(‘lakh’,3) |
| [a@h4-a@] [ah@4-a@] [ah@4-a@] | (‘ah’,2)(‘4a’,2) |
| [side’’by] [side’’by] [side’’by] | (‘side’,3) (‘by’,3) |
| Six{million} six{million} six{million} | (‘six’,3) (‘million’,3) |
| one;billion one;billion one;billion | (‘one’,3) (‘billion’,3) |
| hundred:rupees hundred:rupees | (‘hundred’,2)(‘rupees’,2) |
| {per%centage} {per%centage} | (‘per’,3)(‘centage’,2) |

**Description:**

In the above code I imported the regex package **re.escape** which is used to match a literal string and **re.sub** that returns a string after replacing the matched pattern in a string with a replacement.