In [81]:
```python
import pandas as pd
```

In [82]:
```python
import os
```

In [83]:
```python
#!pip install nltk
```

In [84]:
```python
import nltk
```

In [85]:
```python
from nltk import FreqDist
```

In [86]:
```python
#!pip install matplotlib
```

In [87]:
```python
#!pip install wordcloud
```

In [88]:
```python
import matplotlib.pyplot as plt
```

In [89]:
```python
folder_path = r'C:/Users/fpate/Desktop/AIT-580'
os.chdir(folder_path)
```

In [90]:
```python
os.getcwd()
```

Out[90]:
```
'C:\\Users\\fpate\\Desktop\\AIT-580'
```

In [91]:
```python
text = open("C:\\Users\\fpate\\Desktop\\AIT-580\\WABA-crash copy.txt","r
mytext = text.read()
```

In [92]:
```python
mytext
```

Out[92]:
```
',6810,,,,Crash Description,"Turned right from 38th and Nebraska into a g
ap in traffic. Car approached from behind and tried to pass too fast and
too close, and car\'s wing mirror hit my handlebars. Lost control of the
bike and crashed on the pavement, hitting my head but sustaining no other
injuries. Driver stopped, another witness stopped, both offered aid. Driv
er offered his name and phone number but I failed to ask for insurance in
fo. Didn\'t get witness\'s name or phone number. Also, didn\'t call the p
olice as I didn\'t think I was seriously injured at the time. Only later
discovered I had a concussion and went to the ER. Filed a police report a
fter the fact. Now stuck in an insurance fight to cover the medical treat
ment - I didn\'t realize that car insurance is assumed to be the payer of
first resort, not sure if my medical insurance will ultimately cover the
treatment. Tried to contact the driver on multiple occasions but thus far
getting no response. Would prefer not to go the lawyer route, but I\'m no
t sure how else to find out his car insurance info. Any WABA advice welco
me!"\n11310384,6810,WASHINGTON,DC,20009,Crash Description,"While recoveri
ng from a turn, hit a parallel rut in the road and went down."\n11310789,
6810,Washington,DC,20001,Crash Description,"I was southbound on 3rd Stree
t and saw a northbound motorist with his left signal on. I knew he didn
```

In [95]:
```
#1.Calculate the word frequency counts (eliminating stop words)
```

In [96]:
```
tokenize_wd = nltk.word_tokenize(mytext)
print("count of words :- ",len(tokenize_wd))
tokenize_wd[:10]
```

```
count of words :-  119452
```

Out[96]:
```
[',6810', ',', ',', ',', ',Crash', 'Description', ',', "''", 'Turned', 'rig
ht']
```

In [97]:
```
#Removing extra special characters like , space etc.
tokenizer = nltk.RegexpTokenizer(r"\w+")
remove_s_word = tokenizer.tokenize(mytext)
remove_s_word[:10]
```

Out[97]:
```
['6810',
 'Crash',
 'Description',
 'Turned',
 'right',
 'from',
 '38th',
 'and',
 'Nebraska',
 'into']
```

In [98]:
```
print("count of word :- ",len(remove_s_word))
```

```
count of word :-  106313
```

In [99]:
```
#converting text into the lower character for NLP procedure:
```

In [100]:
```
from nltk.tokenize import RegexpTokenizer
```

In [101]:
```python
tokenizer = RegexpTokenizer(r'\w+')
Token_lower = tokenizer.tokenize(mytext.lower())
print(Token_lower)
```

```
['6810', 'crash', 'description', 'turned', 'right', 'from', '38th', 'an
d', 'nebraska', 'into', 'a', 'gap', 'in', 'traffic', 'car', 'approached',
'from', 'behind', 'and', 'tried', 'to', 'pass', 'too', 'fast', 'and', 'to
o', 'close', 'and', 'car', 's', 'wing', 'mirror', 'hit', 'my', 'handlebar
s', 'lost', 'control', 'of', 'the', 'bike', 'and', 'crashed', 'on', 'th
e', 'pavement', 'hitting', 'my', 'head', 'but', 'sustaining', 'no', 'othe
r', 'injuries', 'driver', 'stopped', 'another', 'witness', 'stopped', 'bo
th', 'offered', 'aid', 'driver', 'offered', 'his', 'name', 'and', 'phon
e', 'number', 'but', 'i', 'failed', 'to', 'ask', 'for', 'insurance', 'inf
o', 'didn', 't', 'get', 'witness', 's', 'name', 'or', 'phone', 'number',
'also', 'didn', 't', 'call', 'the', 'police', 'as', 'i', 'didn', 't', 'th
ink', 'i', 'was', 'seriously', 'injured', 'at', 'the', 'time', 'only', 'l
ater', 'discovered', 'i', 'had', 'a', 'concussion', 'and', 'went', 'to',
'the', 'er', 'filed', 'a', 'police', 'report', 'after', 'the', 'fact', 'n
ow', 'stuck', 'in', 'an', 'insurance', 'fight', 'to', 'cover', 'the', 'me
dical', 'treatment', 'i', 'didn', 't', 'realize', 'that', 'car', 'insuran
ce', 'is', 'assumed', 'to', 'be', 'the', 'payer', 'of', 'first', 'resor
t', 'not', 'sure', 'if', 'my', 'medical', 'insurance', 'will', 'ultimatel
y', 'cover', 'the', 'treatment', 'tried', 'to', 'contact', 'the', 'drive
```

In [102]:
```python
remove_word_lower = list(map(lambda word: word.lower(),remove_s_word))
```

In [103]:
```python
from nltk.corpus import stopwords
```

In [104]:
```python
#Using Filter for tokenzing word by removing stopwords i.e. one of the i
filtered_words = [word for word in remove_word_lower if word not in stop
#filtered_words[:10]
print(filtered_words)
```

```
['6810', 'crash', 'description', 'turned', 'right', '38th', 'nebraska',
'gap', 'traffic', 'car', 'approached', 'behind', 'tried', 'pass', 'fast',
'close', 'car', 'wing', 'mirror', 'hit', 'handlebars', 'lost', 'control',
'bike', 'crashed', 'pavement', 'hitting', 'head', 'sustaining', 'injurie
s', 'driver', 'stopped', 'another', 'witness', 'stopped', 'offered', 'ai
d', 'driver', 'offered', 'name', 'phone', 'number', 'failed', 'ask', 'ins
urance', 'info', 'get', 'witness', 'name', 'phone', 'number', 'also', 'ca
ll', 'police', 'think', 'seriously', 'injured', 'time', 'later', 'discove
red', 'concussion', 'went', 'er', 'filed', 'police', 'report', 'fact', 's
tuck', 'insurance', 'fight', 'cover', 'medical', 'treatment', 'realize',
'car', 'insurance', 'assumed', 'payer', 'first', 'resort', 'sure', 'medic
al', 'insurance', 'ultimately', 'cover', 'treatment', 'tried', 'contact',
'driver', 'multiple', 'occasions', 'thus', 'far', 'getting', 'response',
'would', 'prefer', 'go', 'lawyer', 'route', 'sure', 'else', 'find', 'ca
r', 'insurance', 'info', 'waba', 'advice', 'welcome', '11310384', '6810',
'washington', 'dc', '20009', 'crash', 'description', 'recovering', 'tur
n', 'hit', 'parallel', 'rut', 'road', 'went', '11310789', '6810', 'washin
gton', 'dc', '20001', 'crash', 'description', 'southbound', '3rd', 'stree
t', 'saw', 'northbound', 'motorist', 'left', 'signal', 'knew', 'time', 'm
```

In [105]:
```python
print("count of word :- ",len(filtered_words))
```

```
count of word :-  55689
```

In [106]:
```python
from nltk.stem import WordNetLemmatizer
```

In [107]:
```python
#Lemmatizer is a process of grouping same meaning words together.
lemmatizer = WordNetLemmatizer()
rsp_word_lower_lem = list(map(lambda word:lemmatizer.lemmatize(word),fil
rsp_word_lower_lem[:10]
```

Out[107]:
```
['6810',
 'crash',
 'description',
 'turned',
 'right',
 '38th',
 'nebraska',
 'gap',
 'traffic',
 'car']
```

In [108]:
```python
frequency_dist_words = FreqDist(rsp_word_lower_lem)
frequency_dist_words.most_common(10)
```

Out[108]:
```
[('car', 1191),
 ('bike', 1074),
 ('crash', 949),
 ('lane', 877),
 ('description', 847),
 ('6810', 841),
 ('right', 790),
 ('left', 711),
 ('driver', 680),
 ('dc', 602)]
```

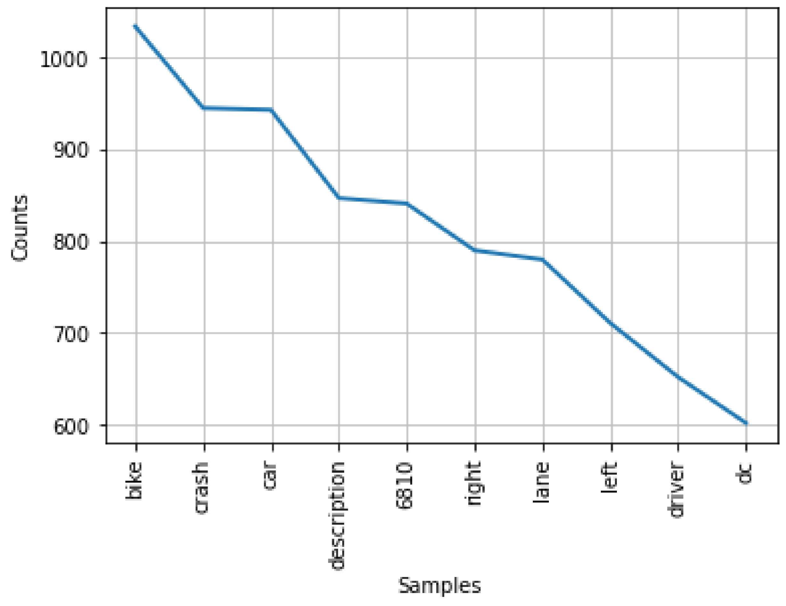
In [109]:
```python
#Plot top 10-word frequency (eliminating stop words)
```

In [110]:
```python
import matplotlib.pyplot as plt
```

In [111]:
```
freq_dist = nltk.FreqDist(filtered_words)
freq_dist

print(freq_dist)
print(freq_dist.most_common(10))
freq_dist.plot(10)
```

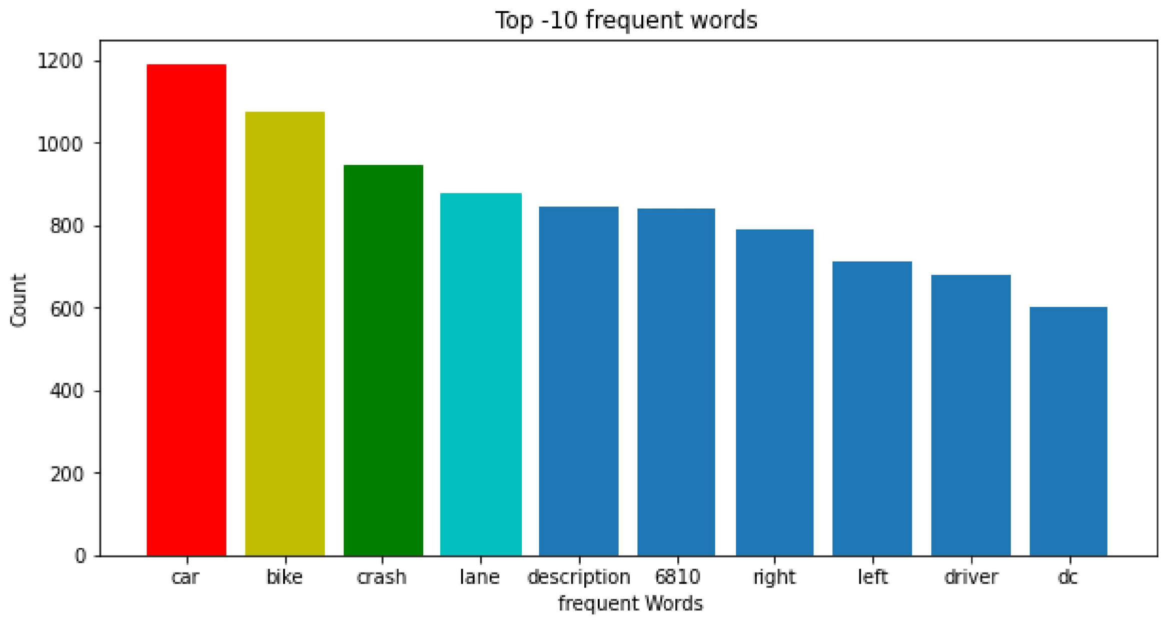
```
<FreqDist with 6347 samples and 55689 outcomes>
[('bike', 1034), ('crash', 945), ('car', 943), ('description', 847), ('681
0', 841), ('right', 790), ('lane', 780), ('left', 711), ('driver', 652),
('dc', 602)]
```



Out[111]:
```
<AxesSubplot:xlabel='Samples', ylabel='Counts'>
```

In [112]:

```python
plt.figure(figsize=(10,5))
barlist=plt.bar(*zip(*frequency_dist_words.most_common(10)))
barlist[0].set_color('r')
barlist[1].set_color('y')
barlist[2].set_color('g')
barlist[3].set_color('c')
plt.title("Top -10 frequent words")
plt.xlabel(" frequent Words")
plt.ylabel("Count")
plt.show()
```



In [113]:

```python
#Interpret the result; what can you learn from the word frequency analys
```

In [114]:

```python
#car, bike, crash, lane, 6810 are the most frequently used words in text
#cyran color in the bar graph.

# Through this word frequency analysis we can get the worthy data from th
# For example, when some organization wants to get some specific kind of
# frequency analysis.

#So, basically the frequency analysis means getting the proper insights f
```

In [115]:

```python
#1.(b)  Extract and display the sentences that include the word ‘police’
```

In [116]:
```
sentences = nltk.sent_tokenize(mytext)
```

In [117]:
```
police=[]
for s in sentences:
    if "police" in s:
        police.append(s)
print(police)
```

```
["Also, didn't call the police as I didn't think I was seriously injured
at the time.", 'Filed a police report after the fact.', 'Since this occur
red just feet from the FBI parking garage entrance, I was immediately sur
rounded by police (I believe FBI police) who must have contacted police a
nd an ambulance.', 'It took about 15 minutes for EMS to arrive—police fol
lowed shortly afterwards.', '(I did give a statement to the police.)"',
'I got a scrape on my arm but nothing significant and my bike wasn\'t tou
ched, so I continued home and then filed a police report."', 'The driver
remained at the scene and appeared to give a statement to the police.',
'As I was still just realizing what all was happening I noticed a police
car stopped ahead at the Vermont intersection, the officer was out of the
car but I was able to yell to him as we stopped at the light.', 'A police
officer was near by but not close enough to see the entire accident.', 'H
e can file a traffic ticket but that would require that I and a witness g
o to police department to complete it.', 'I asked if I need to contact hi
m and he basically said that any police officer can do it.', 'Following m
y returning home from hospital, I call the Montgomery PD to see which pol
ice depart would be most convenient for me and the witness that I had con
tacted to go to.', 'I described to him what happen at the accident and th
e officer came to my hospital room to take the report and how he had advi
```

In [118]:
```
police_sentence= len(police)
```

In [119]:
```
freq_dist = nltk.FreqDist(filtered_words)
freq_dist
```

Out[119]:
```
FreqDist({'bike': 1034, 'crash': 945, 'car': 943, 'description': 847, '681
0': 841, 'right': 790, 'lane': 780, 'left': 711, 'driver': 652, 'dc': 602,
...})
```

In [120]:
```
police_word=freq_dist['police']
```

In [121]:
```
print("count :",police_sentence)
```

```
count : 284
```

In [122]:
```
print("count :",police_word)
```

```
count : 339
```

In [123]:
```
#3.Determine and interpret the general sentiment of the comments about th
```

In [124]:
```
from nltk.sentiment import SentimentIntensityAnalyzer
```

In [125]:
```
sia = SEIA()
```

In [126]:
```
mydata = sia.polarity_scores('.'.join(police))
```

In [127]:
```
mydata
```

Out[127]:
```
{'neg': 0.083, 'neu': 0.879, 'pos': 0.038, 'compound': -0.9998}
```

In [128]:
```
# Through the general sentiment of the word police we get to know that th
#rather than the police in the data i .e. on adding upto 1 it says negat
# of 0.879/1. And the compound scores means to calculate all the lexicon
# case the compound value -0.998 is less than -0.05 so it is negative sen

# Hence, it is NEGATIVE SENTIMENT ANALYSIS.
```