Technical University of Moldova

Special Mathematics

Laboratory No.3

Author:
st. Polina Gore
gr. FAF-161

Supervisor:
Victor Țurcanu

April 5, 2017

Problem No.1

Write a program that prints the first 10 most frequently used **words**, and the number of times it was mentioned.

Ex:

```
the 352
a 235
at 120
. . .
```

Solution

To process the tweets, we have to extract them from a *json* file.
Then split every tweet into words, but first we better lower the letters.
Now, using the function FreqDist from *nltk* module, we find the frequency of every encountered word.
The only thing left is finding the first 10 words that are used the most, for this I used the built-in function most_common.
By printing the results, we notice the following output:

Most common words:

Nr	Word	Frequency
1	the	1322
2	i	920
3	a	903
4	is	838
5	to	815
6	of	671
7	rt	548
8	in	500
9	and	497
10	you	487

Looks like the words *the*, *I*, *a* and so on, are the most used words, which is obvious, because they're used widely in every day life, being adverbs, conjunctions or pronouns.

Problem No.2

Write a program that prints the first 10 most frequently used **nouns**, and the number of times it was mentioned.

Solution

We'll do the same as in the previous problem, by extracting the tweets from the *json* file and splitting them into words.
But now we need to know the word class of each word.
Luckily, we have a function that does this for us - pos_tag, it tags the words with its appropriate class.
So, all we have to do now, is to find all the nouns, save them into a list, and find the frequency of their appearances.
The results are as follows:

Most common nouns:

Nr	Noun	Frequency
1	i	471
2	rt	304
3	time	83
4	thing	47
5	year	39
6	way	36
7	kind	36
8	lot	34
9	game	33
10	man	33

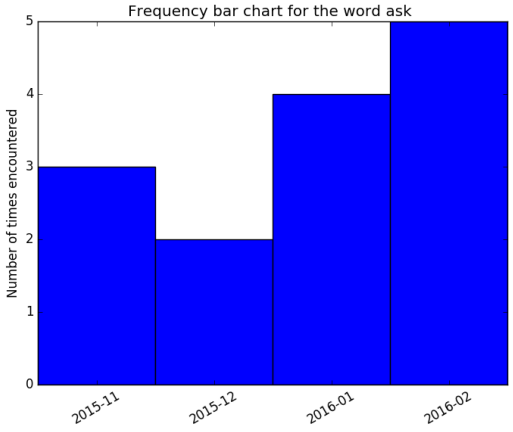
We observe that *I* is the most used noun, so egocentric.

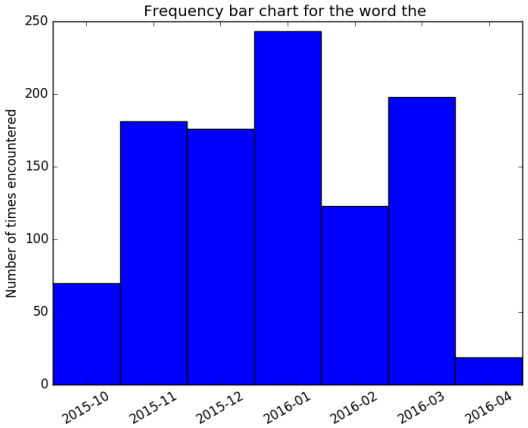
Problem No.3

Write a program that receives a word as an input and draws a frequency bar chart.
Every bar should represent the period of 1 month.

Solution

Again, we extract the tweets from the file, but now, we also save the dates when they were posted online.
Then, we have to compute the number of times we encountered the entered word in each month.
And now, we plot the bar chart for this word.
Examples:





Problem No.4

In our dataset we also have the number of likes and retweets for every message. This can give us some insight about the tweet popularity.
Hence we can compute some sort of rating.
The popularity of nouns is computed by the following formula:
frequency * (1.4 + normRetweet) * (1.2 + normLikes)
The values normRetweet and normLikes are the normalized values of retweets and likes for every word.
To compute the number of likes and retweets for every word you just cumulatively collect the numbers from every tweet that the word was mentioned.

Solution

To compute the popularity of nouns, we choose the same strategy as in the problem No.2, except that now, we also have to know the number of likes and retweets the words have in every tweet.
But, there is always a "but" there, if a word is repeated in a tweet, we have to eliminate the unnecessary summing of the same number of likes and retweets.
After using the formula above, we see the following output:

Most popular nouns:

Nr	Noun	Popularity
1	i	1064026943609
2	time	20386008819
3	thing	4889231141
4	work	741516848
5	way	640161507
6	one	579174359
7	like	535721587
8	do	451118807
9	lot	407723311
10	internet	358239580

Again, we notice the most favorite word of all humanity:

I

But also, we observe some strange things, looks like the 7-th and 8-th words don't really look like nouns, they're verbs.
That is a strange behaviour of the function pos_tag, since it's responsible for tagging them as nouns.

Problem No.5

Write a program that receives as input an uncompleted word and prints 3 word suggestions, followed by their frequency.
The suggestions should be based on the initial dataset and sorted by the word frequency, computed in the first problem.
The input can be any uncompleted word.
Ex. Input: `app`, Output: `application (324), apple (164), appreciate (53).`
Where `application` has the highest frequency, apple the second highest etc.
Ex. Input: `pro`, Output: `programming (196), product (176), program (103).`
Again `programming` has the highest frequency.

Solution

In this problem, we have to find the words that begin with the entered combination - the input.
As in the previous problems, we find the frequency of each word and then, after finding the words that start with the input, we find the frequency of each one, and print only the first 3 most used words.

Examples:

Input: *ju*

Nr	Word	Frequency
1	just	159
2	just	3
3	june	3

Input: *lo*

Nr	Word	Frequency
1	looks	40
2	look	35
3	lot	34

Input: *ap*

Nr	Word	Frequency
1	apple	27
2	app	21
3	apps	14

Input: *su*

Nr	Word	Frequency
1	super	49
2	sure	45
3	support	17

Bonus

Write a program that receives as input a word and prints 3 word suggestions, followed by the suggestion occurrences.
The suggestions should be selected in the following way.
You have to go through your tweets dataset and identify every occurrence of the input word.
At every occurrence collect the word that follows the input word.
That is the suggestion you are looking for.
The input can be any completed word.
Ex. Input: `love`, Output: `programming (5), cars (2), beer (2)`
Ex. Input: `awesome`, Output: `party (10), language (4), framework (2)`

Solution

This problem is similar to the previous one, except that we have to find the occurences of the words that follow the input word and count them.
The steps are pretty much the same, so let's head straight to the output.

Examples:

Input: *the*

Nr	Word	Frequency
1	best	30
2	first	18
3	same	18

Input: *I*

Nr	Word	Frequency
1	am	75
2	don't	54
3	have	53

Input: *no*

Nr	Word	Frequency
1	idea	7
2	longer	4
3	matter	3

Input: *Trump*

Nr	Word	Frequency
1	presidency	2
2	is	2
3	will	1