BUCHAREST UNIVERSITY OF ECONOMIC STUDIES
FACULTY OF BUSINESS ADMINISTRATION IN FOREIGN LANGUAGES
International Master of Business Administration

Research Methods for Business Administration Final Project:

“Simona Halep’s interview after receiving a four-year ban from The International Tennis Integrity Agency (ITIA)”

Students:
Ecaterina Geavela
Iulia Hașcu
Ion-Andrei Alexandru

Bucharest
2024

Table of Contents

1. Abstract

The project works on analyzing an interview by tennis player Simona Halep after a doping scandal. It does analysis of sentiment carefully scraping responses given by Halep in an Euronews interview using advanced techniques of web scraping and natural language processing (NLP). The methods include stop word removal, text cleaning, lemmatization, and frequency analysis of words. The study, therefore, shows findings on Halep's statements on doping that are presented through different visualizations like word clouds and bar graphs. The sentiment analysis reveals a predominantly positive polarity and neutral-objective subjectivity in Halep's responses. This study unquestionably demonstrates the power of web scraping and NLP to infer deep insights from textual data, especially to decide where the stakeholders stand on the issues in contention.

Keywords: *Simona Halep, Doping Scandal, Web Scraping, Natural Language Processing (NLP), Sentiment Analysis, Data Visualization.*

2. Introduction

Speech Sentiment Analysis is an area of the recent burgeoning filed in the umbrella branch of natural language processing (NLP) that involve the use of advanced algorithms and machine learning techniques to decode the emotional tone ingrained within the spoken form of human language (Lian et al., 2023). In an era where humongous amounts of audio data are being generated every day, it is of highest relevance to recognize the sentiments expressed in spoken words for a broad range of topical areas that go from customer feedback analysis and market research to virtual assistants and monitoring of mental health (Cambria et al., 2013).

Sentiment analysis of speech goes beyond its mere transcription, it touches into the realm of nuanced feelings capturing the underlying sentiment, mood or attitude conveyed through spoken words. Using the strategies of computational linguistics and machine learning models, this technology is geared towards ascertaining whether a speaker's tone is positive, negative, neutral, or even modulations like joy, anger, sadness, or surprise (Balazs and Velasquez, 2016).

Encrypted with precise details of human voice speech and crafted to account for the variants in sentiment intended by speakers, sophisticated algorithms built on labeled and available datasets have equipped businesses and researchers to finally provide effective sentiment analysis models useful to spoken language. All these models do also support to enhance our understanding of human communication and are useful in practice for improving customer experiences, gauging public opinion, and helping diagnostics for mental health (Haddi et al., 2013).

The act of doping in sports involves the use of various prohibited substances or methods by athletes to improve either their performance or recovery from exercise. Such substances can include performance-enhancing drugs (PEDs) or even blood doping methods. Use of such substances is not fair from the moral aspect and compromises the spirit associated with fair play in any sport discipline (Waddington and Smith, 2009). It should be noted, however, that whilst doping may exist and some athletes do or support doping practices, attitudes towards doping possibly vary between the athletes and especially according to level of competition. Moreover, anti-doping measures including education measures, testing, and sanctions are in continuous development with a view both to keep sports clean and to protect the health and well-being of the athletes (Overbye, 2016).

This insightful interview of two-time Grand Slam winner Simona Halep, ranging from discussing her perspective on the doping scandal, has been carefully analyzed using web scraping and advanced natural language processing (NLP) techniques to give us insight about the attitude of Simona towards doping. The main corpus for our exhaustive analysis was an interview which can be accessed from this link. A bar chart for frequency distribution of various important words along with the top 10 lemmatized words has been created for easy visualization. Added insights were extended into the development of generic word clouds and personalized with a mask exhibiting Simona Halep holding her Grand Slam trophy.

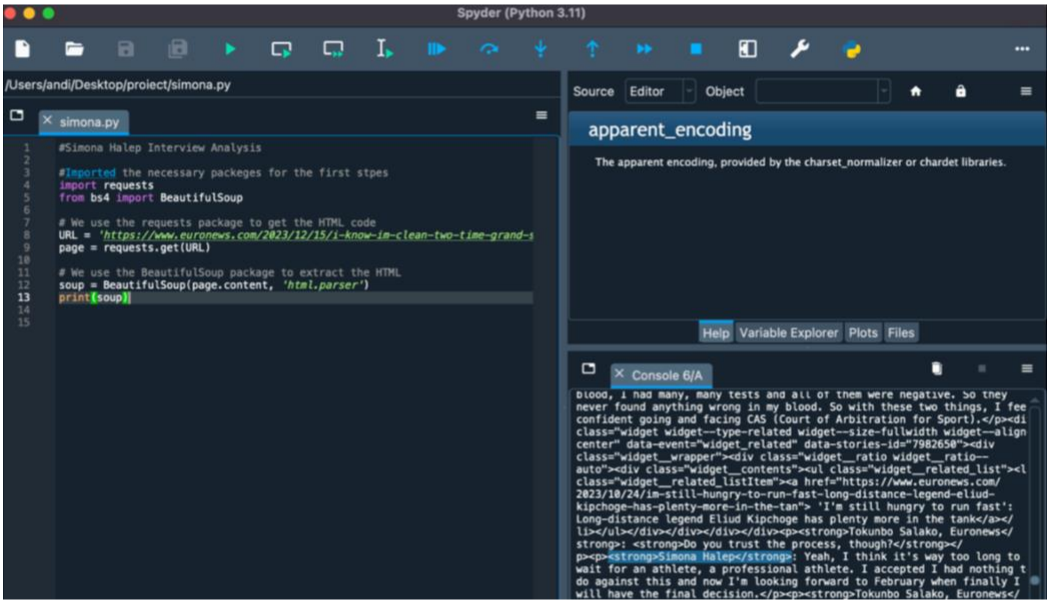
This analysis provides an elaborate overview of the Simona Halep interview that utilizes advanced techniques that could assist in extracting the insights and sentiments that lie within the textual data. The following sub-sections provide results for specific features in the analysis and these are text processing output, most frequent words, visualizations, and sentiment scores.

The interview is available online, here: https://www.euronews.com/2023/12/15/i-know-im-clean-two-time-grand-slam-winner-simona-halep-opens-up-about-doping-scandal.

3. Methods

In order to automatically extract data from the website, we used web scraping. Web scraping helped us collect and export the data into a format which is more useful for our analysis. To interpret the results, we are going to break down the code and explain its functions.

First, we had to import the HTML code from the website to Spyder. For this, we used the requests package, getting the code from the URL. After that, we extracted the HTML content to identify the specific content, most exactly the answers provided by Simona Halep in this interview.



Looking through the HTML code we identified that within the code, Simona Halep's answers were every time placed right after the Simona Halep . The name Simona Halep must be specified because the interviewer's name was also between strong tags.

Next, we had to find all the <strong> tags with the text "Simona Halep". So, we used the following line to search the parsed HTML content for all <strong> tags where the text exactly matches 'Simona Halep'. We also used the *find_all*, to obtain a list of all matching tags.

```
# We find all <strong> tags with the text 'Simona Halep'
halep_strong_tags = soup.find_all('strong', string='Simona Halep')
```

Afterwards, we initialized an empty list to hold Halep's answers before setting up the function to find all of them.

```
# We make an empty list to hold Simona's answers
halep_answers = []

```

Setting up the function, we began by iterating over each found <strong> tag. We also used *next_sibling*, a BeautifulSoup method that moves to the next element at the same level in the HTML tree. In our case, the loop continues until it encounters another <strong>, <h2>, or <h3> tag, which we assumed might indicate the start of a new question or a new section. If the next sibling has text (*next_sibling.string*), this text is added to the *halep_answers* list.

```
# Iterating over each found <strong> tag
for strong_tag in halep_strong_tags:
    # Looking for the answers
    next_sibling = strong_tag.next_sibling
    while next_sibling and next_sibling.name not in ['strong', 'h2', 'h3']:
        if next_sibling.string:
            halep_answers.append(next_sibling.string.strip())
        next_sibling = next_sibling.next_sibling

print(halep_answers)

```

```
[": Well, it's been actually\xa0more than one year already,\xa0and ever
day\xa0it felt very painful,\xa0very emotional, hurtful,\xa0because I
know\xa0I didn't do anything wrong\xa0and I know I'm clean.\xa0So it wa
a shock\xa0when I received the letter\xa0that my urine test,\xa0only th
urine test\xa0came out positive,\xa0with actually an extremely low
quantity\xa0of substance, banned substance.\xa0I've been always against
doping\xa0and you know,\xa0I've been loud as well about this,\xa0so it
didn't even cross my mind\xa0in my whole life\xa0to do something like
this.\xa0So it was a shock.\xa0I struggled with the emotional
part\xa0because it's been\xa0very heavy on my shoulders\xa0and seeing
this\xa0so much in the public,\xa0it was really affecting\xa0my mental
health, for sure.", ": Well, the support has been amazing.\xa0The fans
who are\xa0supporting me unconditionally,\xa0which means a lot.\xa0It
means a huge amount\xa0to see the people,\xa0even if I'm facing the
```

We observed that the answers were indeed extracted to some extent but needed further cleaning for obtaining the final text. Therefore, we searched for a function that goes through each item in *halep_answers* and performs a cleanup.

This lead us to *' '.join(answer.split())*, a common Python idiom for removing extra whitespace within a string. It splits the string into words (on whitespace) and then joins them back together with a single space. Also, the *if answer and not answer.isspace()* ensures that only non-empty, non-whitespace strings are included.

After running the new list, *cleaned_answers*, we managed to obtain Simona's answers in an almost clean format:

```
# We make an empty list to hold Simona's answers
halep_answers = []
for strong_tag in halep_strong_tags:
    # Looking for the answers
    next_sibling = strong_tag.next_sibling
    while next_sibling and next_sibling.name not in ['strong', 'h2', 'h3']:
        if next_sibling.string:
            halep_answers.append(next_sibling.string.strip())
        next_sibling = next_sibling.next_sibling

# Cleaning the extracted the answers
cleaned_answers = [' '.join(answer.split()) for answer in halep_answers if
cleaned_answers
```

```
Console 8/A

In [3]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
Out[3]:
[": Well, it's been actually more than one year already, and every day
it felt very painful, very emotional, hurtful, because I know I didn't
do anything wrong and I know I'm clean. So it was a shock when I
received the letter that my urine test, only the urine test came out
positive, with actually an extremely low quantity of substance, banned
substance. I've been always against doping and you know, I've been loud
as well about this, so it didn't even cross my mind in my whole life to
do something like this. So it was a shock. I struggled with the
emotional part because it's been very heavy on my shoulders and seeing
this so much in the public, it was really affecting my mental health,
for sure.",
```

The only thing that remained there were some unwanted characters like *"[:", ":", and "]",* which we wanted to remove too. We did this by using the *replace()* method, after, we iterated over the list of cleaned responses and added each one to a string variable *halep_speech_str*.

```
# Cleaning the text completely
halep_speech_str = ""
for answer in halep_answers:
    cleaned_answer = answer.replace('[:', '').replace(':",', '').replace(':
    halep_speech_str += " " + cleaned_answer

print(halep_speech_str)
```

```
In [17]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
 Well, it's been actually more than one year already, and every day it
felt very painful, very emotional, hurtful, because I know I didn't do
anything wrong and I know I'm clean. So it was a shock when I received
the letter that my urine test, only the urine test came out
positive, with actually an extremely low quantity of substance, banned
substance. I've been always against doping and you know, I've been loud
as well about this, so it didn't even cross my mind in my whole life to
do something like this. So it was a shock. I struggled with the
```

Now that we have the final text clean, the extracted answers will allow running NLP techniques and computing analysis based on the frequency of words. We do this using the NLTK python package and some other features, including stop word elimination. We proceeded by importing the package.

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('omw-1.4')
```

Next, we want to create the list containing only the important words, without "and", "the", "an/a" or others which are frequently used. Thus, we must remove them. The first step we did was to convert all to lowercase, so they are treated the same way. Then, we used NLTK's predefined list of stop words. Also, the *isalpha()* check ensured that we only kept words (not numbers or punctuation).

```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords

# Define English stop words, tokanize the answers and convert, filter stop w
stop_words = stopwords.words('english')
words = word_tokenize(halep_speech_str.lower())
filtered_speech = [w for w in words if w.isalpha() and not w in stop_words]
print(filtered_speech)
```

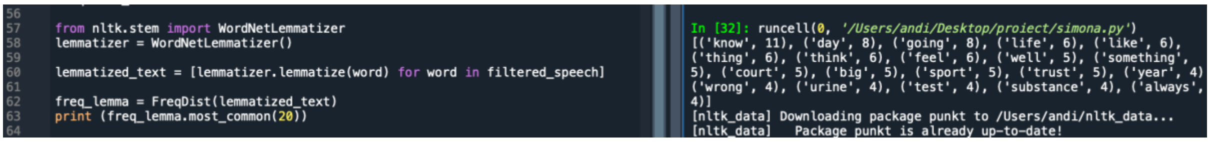
```
In [25]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
['well', 'actually', 'one', 'year', 'already', 'every', 'day', 'felt',
'painful', 'emotional', 'hurtful', 'know', 'anything', 'wrong', 'know',
'clean', 'shock', 'received', 'letter', 'urine', 'test', 'urine', 'test'
'came', 'positive', 'actually', 'extremely', 'low', 'quantity',
'substance', 'banned', 'substance', 'always', 'doping', 'know', 'loud',
'well', 'even', 'cross', 'mind', 'whole', 'life', 'something', 'like',
'shock', 'struggled', 'emotional', 'part', 'heavy', 'shoulders',
```

As our next goal was to find out which words were used the most by Simona Halep, and how many times each, we used the FreqDist tool from NLTK, asking for a display of top 20 most frequently used words with *freq.most_common(20)*.

```
#Analyze the frequency
from nltk import FreqDist
freq = FreqDist(filtered_speech)
print (freq.most_common(20))
```

```
In [30]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
[('know', 11), ('going', 8), ('life', 6), ('like', 6), ('think', 6),
('feel', 6), ('well', 5), ('something', 5), ('court', 5), ('big', 5),
('sport', 5), ('trust', 5), ('day', 4), ('wrong', 4), ('urine', 4),
('substance', 4), ('always', 4), ('support', 4), ('means', 4), ('lot'
4)]
```

After we observed the most frequently used words, we used the *WordNetLemmatizer()* tool from NLTK to lemmatize our list. It basically combines words with very similar meaning into one single word.



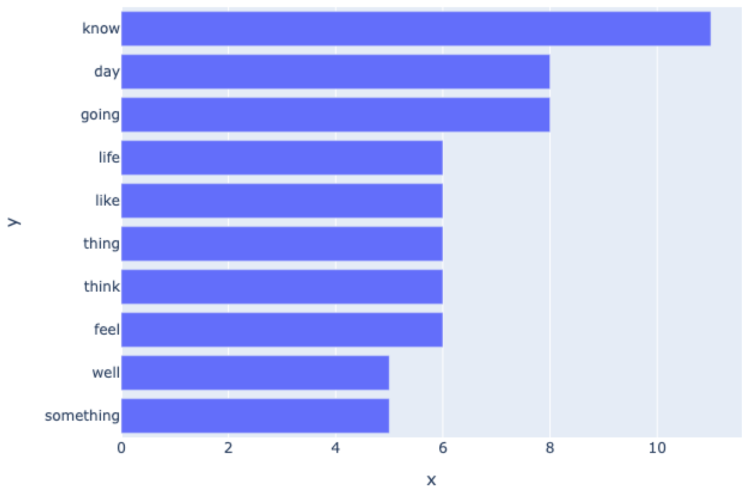
```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

lemmatized_text = [lemmatizer.lemmatize(word) for word in filtered_speech]

freq_lemma = FreqDist(lemmatized_text)
print (freq_lemma.most_common(20))
```

```
In [32]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
[('know', 11), ('day', 8), ('going', 8), ('life', 6), ('like', 6),
('thing', 6), ('think', 6), ('feel', 6), ('well', 5), ('something',
5), ('court', 5), ('big', 5), ('sport', 5), ('trust', 5), ('year', 4)
('wrong', 4), ('urine', 4), ('test', 4), ('substance', 4), ('always',
4)]
[nltk_data] Downloading package punkt to /Users/andi/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

For better visualization, we decided to create a bar graph, starting with the highest value at the top, including 10 of the most used (and lemmatized) words. We decided to save the file directly to our work folder so we can easily upload it in the document.

```
#Creating the bar graph
import plotly.express as px
common_words, common_freqs = zip(*freq_lemma.most_common(10))

fig = px.bar(x = common_freqs, y = common_words, orientation = 'h')
fig.update_layout(yaxis = dict(autorange="reversed"))

#Download bar graph to our folder
file_path = '/Users/andi/Desktop/proiect/filename.png'
fig.write_image(file_path)
print(f"Figure saved as {file_path}")
```

```
[nltk_data] Downloading package punkt to /Users/andi/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /Users/andi/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /Users/andi/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /Users/andi/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
Figure saved as /Users/andi/Desktop/proiect/filename.png
```



We now remove any unwanted characters from our speech string using a regular expression. This will remove all characters (*such as !, #, and @*) that are not lowercase or uppercase English letters, numbers or spaces. The cleaned speech string was then written into an Output.txt file.

```python
import re

# Make the Regex cleaning
expression = "[^a-zA-Z0-9 ]"  # keep only letters, numbers, and whitespace
cleantextCAP = re.sub(expression, ' ', halep_speech_str)  #apply
cleantext = cleantextCAP.lower() # lower case


# Save for wordcloud
with open("Output.txt", "w") as text_file:
    text_file.write(cleantext)

# Count and create dictionary
words = cleantext.split(" ")
print(len(words))

dict1 = {}
for word in words:
    if word:
        dict1[word] = words.count(word)

print(cleantext)
```

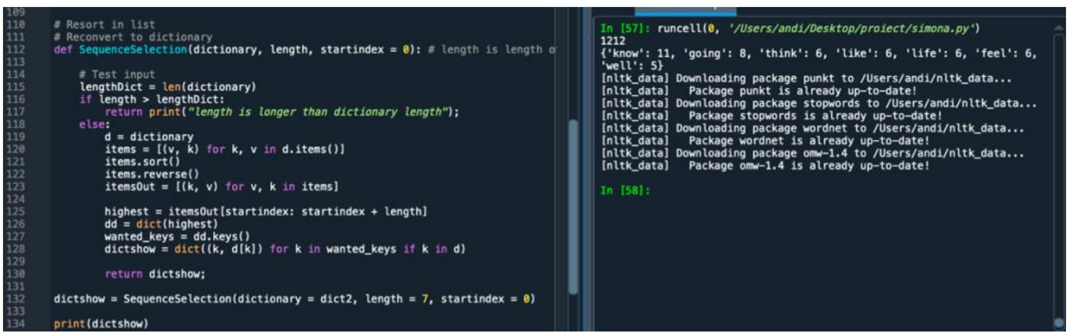
```
In [46]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
Figure saved as /Users/andi/Desktop/proiect/filename.png
1212
 well  it s been actually more than one year already  and every day it
felt very painful  very emotional  hurtful  because i know i didn t do
anything wrong and i know i m clean  so it was a shock when i received
the letter that my urine test  only the urine test came out positive
with actually an extremely low quantity of substance  banned substance
i ve been always against doping and you know  i ve been loud as well
about this  so it didn t even cross my mind in my whole life to do
something like this  so it was a shock  i struggled with the emotional
part because it s been very heavy on my shoulders and seeing this so
much in the public  it was really affecting my mental health  for sure
well  the support has been amazing  the fans who are supporting me
unconditionally  which means a lot  it means a huge amount to see the
people  even if i m facing the worst moment in my life as an athlete i
know i m clean  i received tonnes of messages  good messages  and the
biggest thing is that i ve never faced a person who told me something
negative  so all of them were positive  and this gave me the strength t
keep fighting every day  to clear my name and to show that i didn t do
anything wrong  the players also  which are opponents  they also showed
```

After we got all the dictionary's keys for each of our terms, we have chosen just the words that weren't stop words in English. Using this, we made a second dictionary where the values represent the words' number of appearances and the keys represent the filtered words, matching our first dictionary.

```python
from  nltk.corpus import stopwords

# Unsorted speech constituents in dictionary as dict1
keys = list(dict1)
filtered_words = [word for word in keys if word not in stopwords.words('english
dict2 = dict((k, dict1[k]) for k in filtered_words)

print(filtered_words)
```

```
In [53]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
1212
['well', 'actually', 'one', 'year', 'already', 'every', 'day', 'felt',
'painful', 'emotional', 'hurtful', 'know', 'anything', 'wrong', 'clean'
'shock', 'received', 'letter', 'urine', 'test', 'came', 'positive',
'extremely', 'low', 'quantity', 'substance', 'banned', 'always',
'doping', 'loud', 'even', 'cross', 'mind', 'whole', 'life', 'something'
```

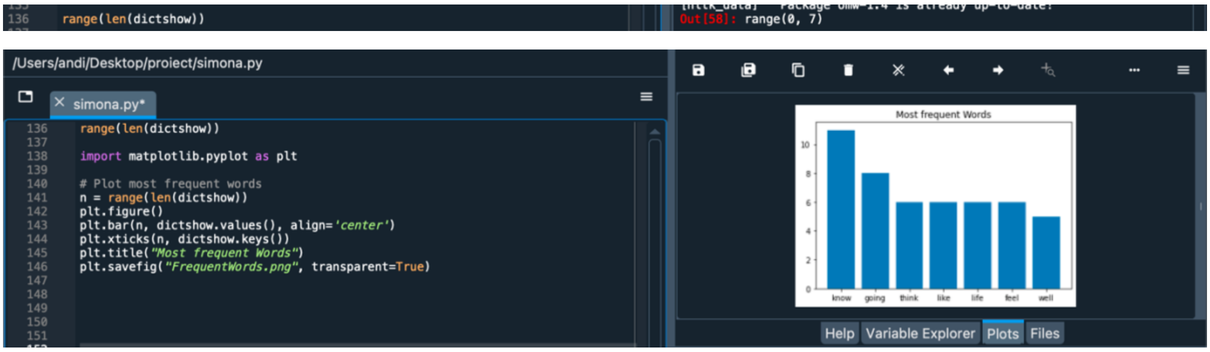
Next, we developed the Sequence Selection function, which accepts a length and start index as well as a dictionary containing word keys and frequency values. By taking all the key-value pairs in the dictionary and producing a list of value-key pairs in its place, this will sort our input dictionary in descending order based on the frequency values. Using this sub-list, we build a dictionary from our sorted key-value pair list by sorting it according to start index and length.



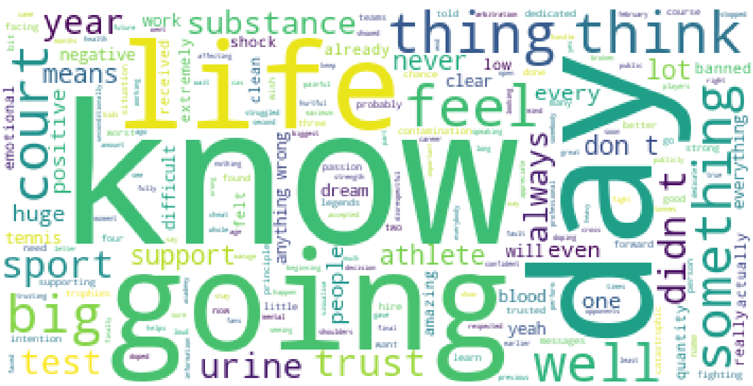
```python
# Resort in list
# Reconvert to dictionary
def SequenceSelection(dictionary, length, startindex = 0): # length is length o

    # Test input
    lengthDict = len(dictionary)
    if length > lengthDict:
        return print("length is longer than dictionary length");
    else:
        d = dictionary
        items = [(v, k) for k, v in d.items()]
        items.sort()
        items.reverse()
        itemsOut = [(k, v) for v, k in items]

        highest = itemsOut[startindex: startindex + length]
        dd = dict(highest)
        wanted_keys = dd.keys()
        dictshow = dict((k, d[k]) for k in wanted_keys if k in d)

        return dictshow;

dictshow = SequenceSelection(dictionary = dict2, length = 7, startindex = 0)

print(dictshow)
```

```
In [57]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
1212
{'know': 11, 'going': 8, 'think': 6, 'like': 6, 'life': 6, 'feel': 6,
'well': 5}
[nltk_data] Downloading package punkt to /Users/andi/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /Users/andi/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /Users/andi/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /Users/andi/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!

In [58]:
```

The following step was to print the range of numbers, from 0 to the number of key-value pairs in our dictionary. The outcome of our sequence selection was also presented as a bar graph, with the words' names on the horizontal line and their number of occurrences on the vertical line representing our dict. This was displayed in Python and also automatically saved to the folder.

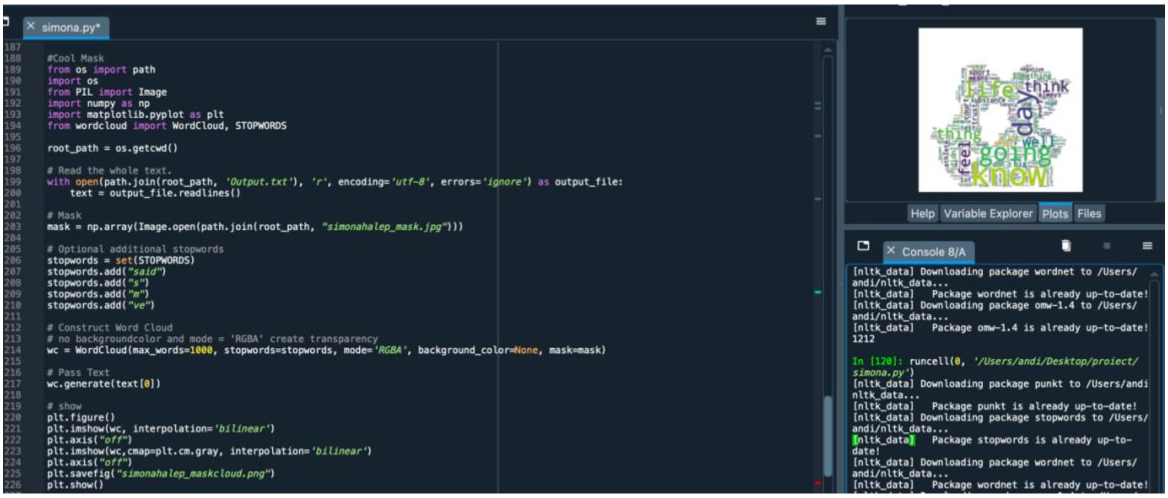


After importing other tools and packages that we required, we read the entire document. We next added optionally more stop words from Wordcloud, using the previous Output.txt as a dictionary for the word key frequency values that need to be plotted. After creating a suitable Wordcloud for our case study beforehand, we utilized matplotlib to bring everything together. The most often used words were finally displayed in a graphic that we could see, based on how many times they appear in Simona Halep's interview answers.



```python
from os import path
import os
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

root_path = os.getcwd()

# Read the whole text.
with open(path.join(root_path, 'Output.txt'), 'r', encoding='utf-8', errors='ignore') as output_file:
    text = output_file.readlines()


# Optional additional stopwords
stopwords = set(STOPWORDS)
stopwords.add("said")
stopwords.add("s")
stopwords.add("m")
stopwords.add("ve")

# Construct Word Cloud
# no backgroundcolor and mode = 'RGBA' create transparency
wc = WordCloud(max_words=1000, stopwords=stopwords, mode='RGBA', background_color=None)

# Pass Text
wc.generate(text[0])

# store to file
wc.to_file(path.join(root_path, "simonahalep.png"))

# show
plt.figure()
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.imshow(wc, cmap=plt.cm.gray, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Next, we generated another word cloud using the same procedure as before, but this time we used a mask. Unlike the previous rectangular shape, the mask—a photo of Simona Halep holding her first Grand Slam trophy—was utilized as the background shape on which the word cloud was generated.



Finally, we performed sentiment analysis on this using the TextBlob package and our cleaned interview text. Subjectivity is a score that ranges from 0.0 to 1.0, with 0 being very objective and 1 representing very subjective. Polarity is a number that ranges from -1.0 to 1.0, with -1 representing very negative and 1 representing very positive.

```
from textblob import TextBlob

sentiment = TextBlob(cleantext)
print("Polarity Score: ", sentiment.sentiment.polarity)

print("Subjectivity Score: ", sentiment.sentiment.subjectivity)
```

```
Polarity Score:
0.05288862652499018
Subjectivity Score:
0.5451435622458349

In [126]:
```

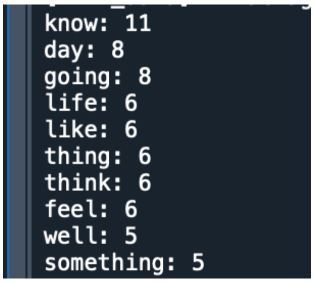
We obtained a polarity score of 0. 05288862652499018 which is pretty much positive and a subjectivity score of 0.5451435622458349, which is consider neutral objective.

4. Results

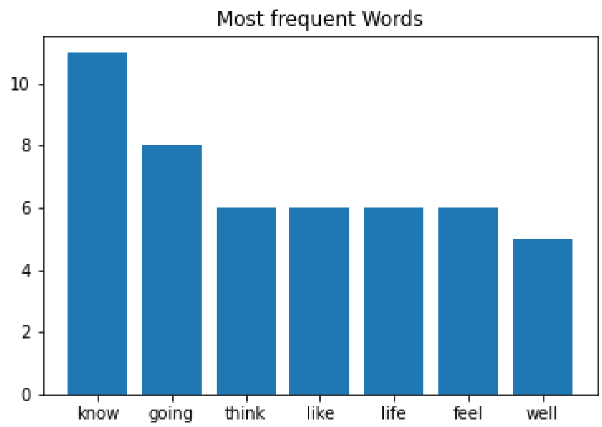
a. Text processing

```
In [94]: runcell(0, '/Users/andi/Desktop/proiect/simona.py')
1212
['', 'well', 'actually', 'one', 'year', 'already', 'every', 'day', 'felt', 'painful', 'emotional', 'hurtful', 'know', 'anything', 'wrong', 'clean', 'shock', 'received',
'letter', 'urine', 'test', 'came', 'positive', 'extremely', 'low', 'quantity', 'substance', 'banned', 'always', 'doping', 'loud', 'even', 'cross', 'mind', 'whole',
'life', 'something', 'like', 'struggled', 'part', 'heavy', 'shoulders', 'seeing', 'much', 'public', 'really', 'affecting', 'mental', 'health', 'sure', 'support',
'amazing', 'fans', 'supporting', 'unconditionally', 'means', 'lot', 'huge', 'amount', 'see', 'people', 'facing', 'worst', 'moment', 'athlete', 'tonnes', 'messages',
'good', 'biggest', 'thing', 'never', 'faced', 'person', 'told', 'negative', 'gave', 'strength', 'keep', 'fighting', 'clear', 'name', 'show', 'players', 'also',
'opponents', 'showed', 'appreciate', 'court', 'situation', 'legends', 'big', 'tennis', 'publicly', 'speaking', 'fully', 'great', 'everything', 'helps', 'stay', 'strong',
'difficult', 'times', 'fight', 'yeah', 'said', 'contamination', 'three', 'days', 'blood', 'beginning', 'could', 'doped', 'intention', 'disrespectful', 'sport',
'respected', 'dedicated', 'principles', 'think', 'cheat', 'two', 'things', 'second', 'many', 'tests', 'found', 'feel', 'confident', 'going', 'cas', 'arbitration', 'way',
'long', 'wait', 'professional', 'accepted', 'nothing', 'looking', 'forward', 'february', 'finally', 'final', 'decision', 'true', 'went', 'wish', 'done', 'little', 'bit',
'earlier', 'stopped', 'working', 'academy', 'manage', 'trusted', 'teams', 'previous', 'everybody', 'work', 'trusting', 'better', 'chance', 'perform', 'maximum', 'open',
'learn', 'hire', 'need', 'information', 'trust', 'broken', 'right', 'future', 'probably', 'principle', 'somebody', 'soon', 'months', 'ago', 'four', 'years', 'age',
'least', '25', 'catastrophic', 'handle', 'end', 'career', 'yes', 'fault', 'kids', 'say', 'dream', 'important', 'visualise', 'trophies', 'course', 'happen', 'dedicate',
'disciplined', 'hard', 'passion', 'able', 'lift', 'share', 'courage', 'go', 'tired', 'exhausted', 'depressed', 'sometimes', 'push', 'step', 'luck', 'confidence',
'inside', 'chances', 'dreaming', 'paris', 'city', 'roland', 'garros', 'junior', 'started', 'early', 'back', 'matter', 'want', 'belong', 'thank', 'listening', 'talking']
```

b. Ten most frequently used word


```
know: 11
day: 8
going: 8
life: 6
like: 6
thing: 6
think: 6
feel: 6
well: 5
something: 5
```

c. Most frequent words chart



d. Word cloud

e. Mask

f. Sentiment scores

Polarity Score: 0.05288862652499018

Subjectivity Score: 0.5451435622458349

5. Discussion and Conclusion

The meticulous process began with the extraction of relevant data through web scraping, followed by an intricate breakdown of code functions to precisely identify Simona Halep's responses within the HTML structure. Cleaning and refining the extracted text ensured a polished dataset for subsequent analysis.

The NLP techniques, including the removal of stop words and lemmatization, paved the way for a detailed exploration of the most frequently used words. The ensuing bar graph, showcasing the top lemmatized terms, offered a clear visual of the interview's prominent themes.

Further enriching the analysis, the emotional ratings were derived through the TextBlob package, indicating a predominantly positive polarity and a neutral-objective subjectivity in Simona Halep's responses.

The results, presented in a structured format covering text processing, the ten most frequently used words, a visual representation of word frequency, word clouds, and sentiment scores, collectively offer a comprehensive overview of the interview's key insights and emotional undertones.

This analytical journey not only reveals the technical prowess of web scraping and NLP but also underscores the depth of understanding that can be extracted from textual data. Simona Halep's interview, scrutinized through these advanced techniques, opens a window into her thoughts and feelings on a matter of significant importance in the sporting world.

6. References

Balazs, J.A. and Velásquez, J.D. (2016). Opinion Mining and Information Fusion: A survey. *Information Fusion*, 27, pp.95–110. doi:https://doi.org/10.1016/j.inffus.2015.06.002.

Cambria, E., Schuller, B., Xia, Y. and Havasi, C. (2013). New Avenues in Opinion Mining and Sentiment Analysis. *IEEE Intelligent Systems*, 28(2), pp.15–21. doi:https://doi.org/10.1109/mis.2013.30.

Haddi, E., Liu, X. and Shi, Y. (2013). The Role of Text Pre-processing in Sentiment Analysis. *Procedia Computer Science*, 17, pp.26–32. doi:https://doi.org/10.1016/j.procs.2013.05.005.

Lian, H., Lu, C., Li, S., Zhao, Y., Tang, C. and Zong, Y. (2023). A Survey of Deep Learning-Based Multimodal Emotion Recognition: Speech, Text, and Face. *Entropy*, [online] 25(10), p.1440. doi:https://doi.org/10.3390/e25101440.

Overbye, M. (2016). Doping control in sport: An investigation of how elite athletes perceive and trust the functioning of the doping testing system in their sport. *Sport Management Review*, 19(1), pp.6–22. doi:https://doi.org/10.1016/j.smr.2015.10.002.

Waddington, I. (2009). *An Introduction to Drugs in Sport*. [online] Available at: http://ndl.ethernet.edu.et/bitstream/123456789/24847/1/13.pdf.pdf.