Analyzing data with PySpark

About the Data:

The data I selected for analysis was the text file of the book "**The Hound of the Baskervilles**" by Sir Arthur Conan Doyle, as This is one of my favorites when it comes to Sherlock Homes Character, moreover this book being crime mystery would be great analysis of the words and parts of speech of the words in the book and see if we find any trend in the words used by the author.

MapReduce and RDD:

Resilient Distributed Datasets are used to parallelize our Big Data in Spark (RDDs). RDDs are the most fundamental abstraction in Apache Spark, and they partition our original data into many clusters. The best thing about RRDs is that they are fault tolerant, meaning they can recover lost data if one or more workers fail. While working with the data we also create many RDDs and either by filtering the data or using other kind of transformations. Transformations take RDDs and change them into new datasets, returning an RDD therefore (eg. map, filter and reduce by key operations). All transformations are lazy that means they only run once when an action is invoked (they are placed in an execution map and then performed when an Action is called).

Findings and Trend:

After Analysis of the text from the book, we find the frequencies of the words in the book and here we notice that the word 'of' has the highest word count of 1702 considering this also a stop word we will need to do some data processing and then check which words has the highest occurrence. To do that we make the corpus into a RDD. We pass the book RDD into function called "func" which basically converts the corpus data into all lowercase and then splits them by spaces, now we apply RDD functions Map () and FlatMap() to store all the words of the corpus as an array.

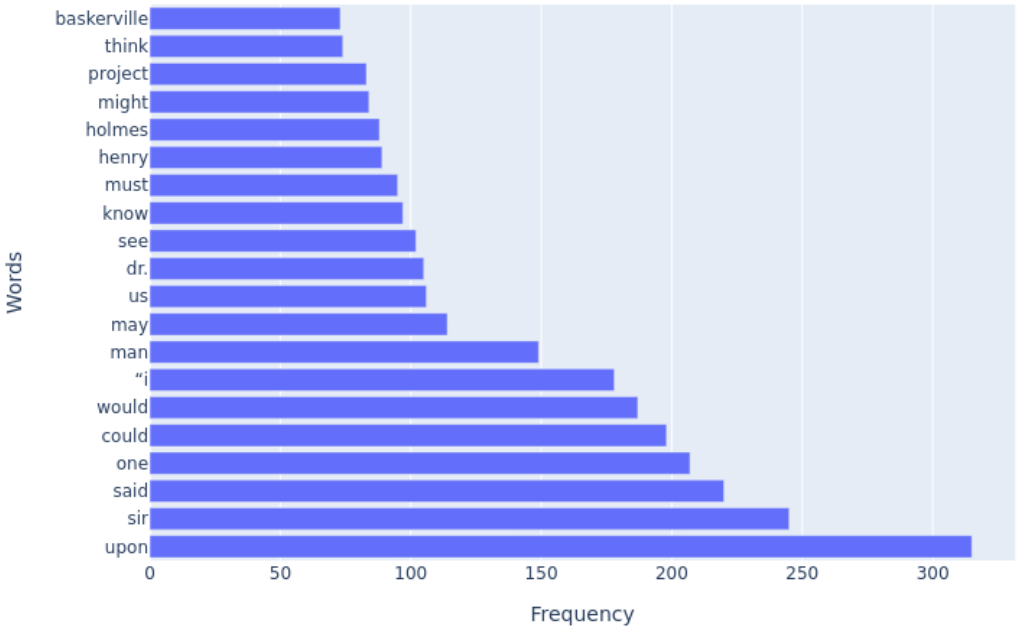
Now I made a list of all the stop words (this list is taken from one of my previous projects on document classification), now using the filter and lambda function I remove stop words form the RDD. To also learn about group by I tried to group by the words with same starting two letters in the words and the results can be seen in the notebook/html and. since the title of the book is The Hound of the Baskervilles, I also was intrigued to see how many times the word Hound was used in the book and to my surprise it was very less with the word count of 44

Now move towards Finding the words count all the words in the corpus after removing the stop words and visualize the top 20 words in the corpus (excluding the stop words). This time for finding the frequency of words we use a different and simple approach by using the group by key , map ,sort by key and map values rdd functions to find the frequency of words and initiated a spark session and convert the rdd into spark data frame of TOP 20 words used in the corpus we find out that the words 'Upon' is the most used words in the book which is a preposition.

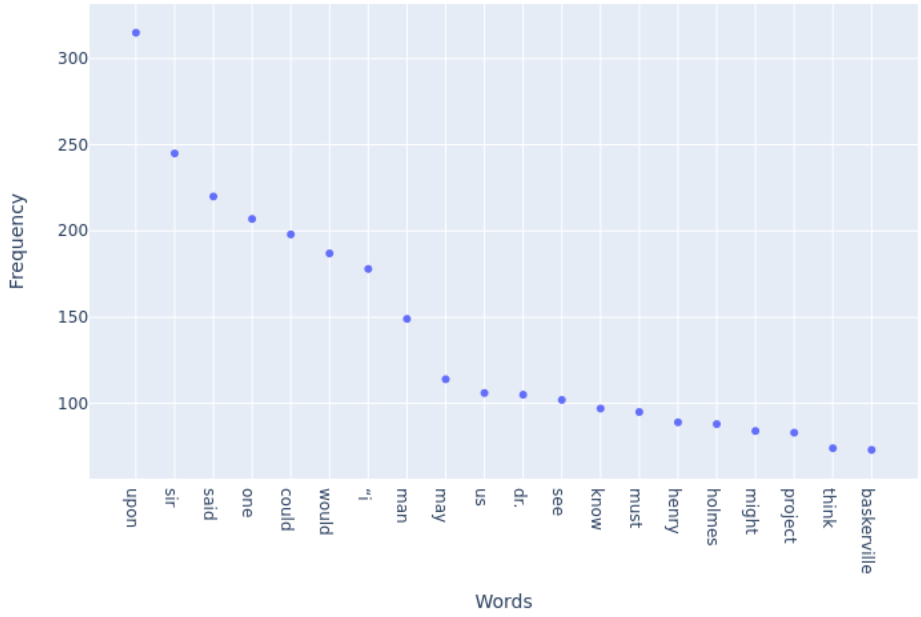
```
root
 |-- Frequency: long (nullable = true)
 |-- Words: string (nullable = true)

+---------+-----------+
|Frequency|Words      |
+---------+-----------+
|315      |upon       |
|245      |sir        |
|220      |said       |
|207      |one        |
|198      |could      |
|187      |would      |
|178      |“i         |
|149      |man        |
|114      |may        |
|106      |us         |
|105      |dr.        |
|102      |see        |
|97       |know       |
|95       |must       |
|89       |henry      |
|88       |holmes     |
|84       |might      |
|83       |project    |
|74       |think      |
|73       |baskerville|
+---------+-----------+
only showing top 20 rows
```

Spark data frame of top 20 words of the book



Words Vs Frequency Bar Plot of the top 20 words in the book



Frequency Vs words Scatter Plot of the top 20 words in the book

Both graph show “Baskerville” is 20 most used word in the book which is a Proper Noun with a word count of 73 and highest count word is “upon” with a frequency of 315.

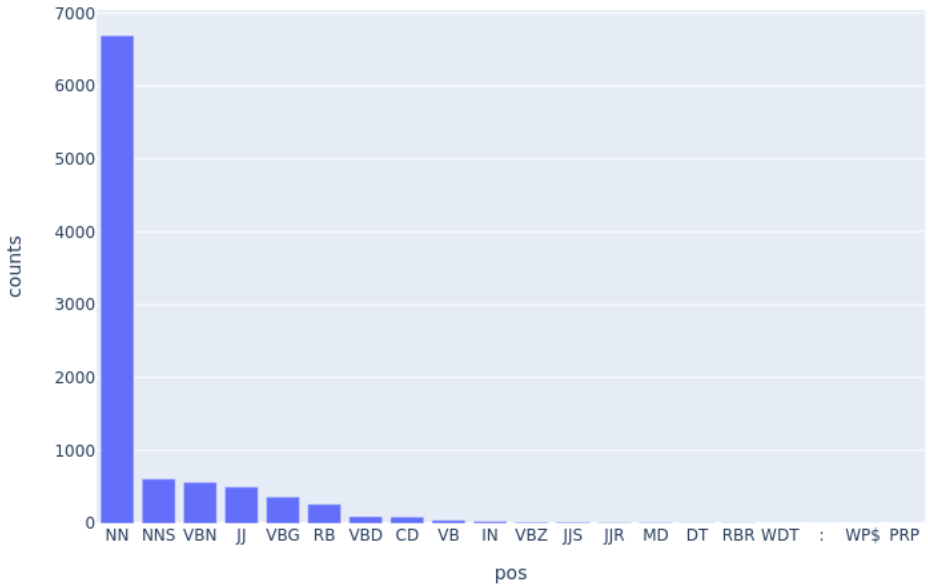
I have also done some Parts of speech tagging to our word frequency data frame , for this task I have used natural language processing library NLTK and used the function pos_tag() , usually this is used after we tokenized the text data but due our previous procession that has already ben taken care of , once the pos are tagged we then then create a data frame of the pos and the count of the words with respect to the words in our corpus

	Frequency	Words	pos
0	315	upon	IN
1	245	sir	NN
2	220	said	VBD
3	207	one	CD
4	198	could	MD

The above data frame is the word frequency with tagged parts of speech which says “upon” is preposition and “sir” is a common noun.

	pos	counts
0	NN	6690
1	NNS	608
2	VBN	564
3	JJ	498
4	VBG	361

The above is the data frame of the pos counts of the words in the corpus the highest is the NN which is the tag for Common noun and NNS is the second highest which the tag for Noun plural.



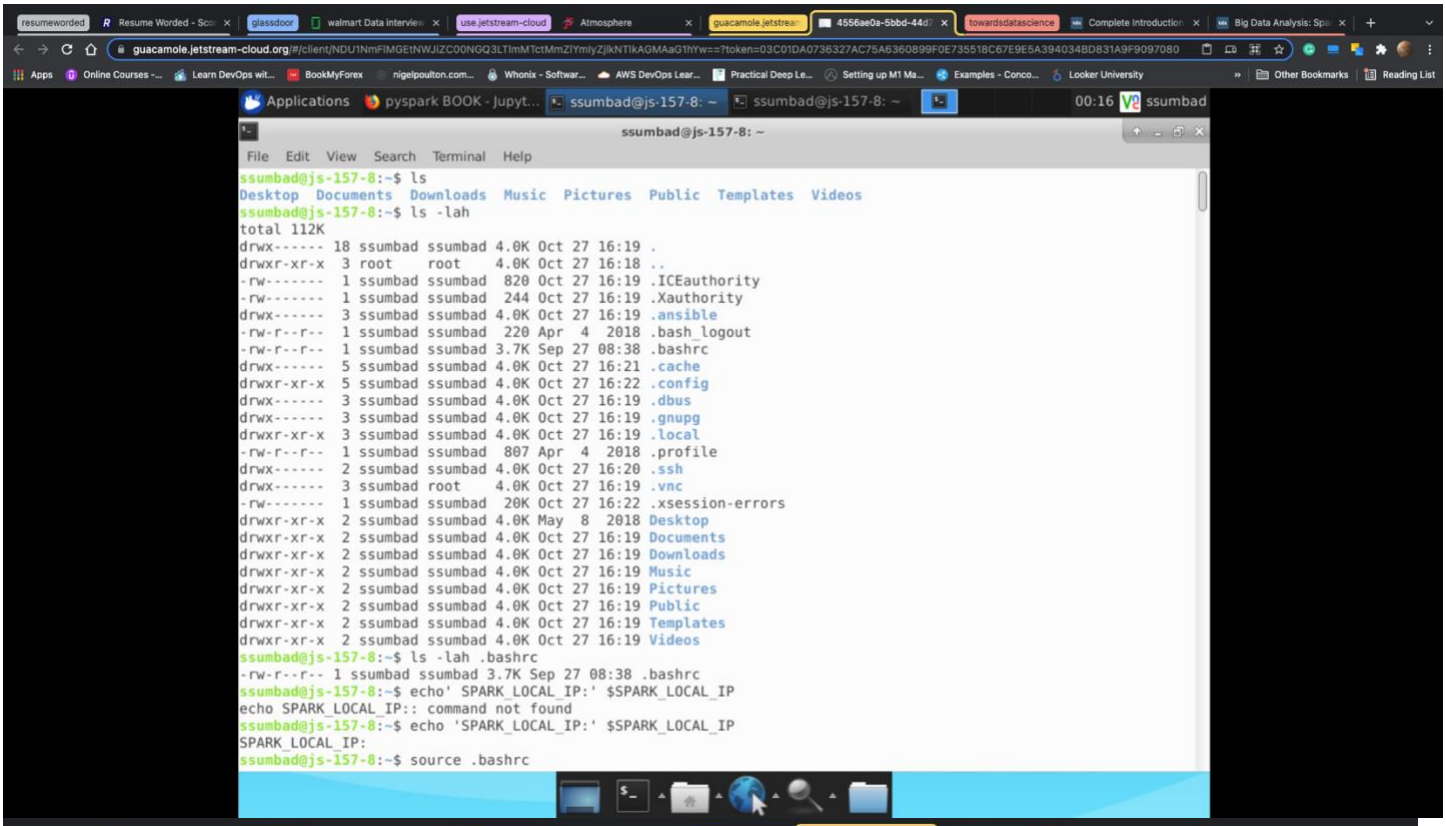
Bar Chart of the parts of speech in the corpus

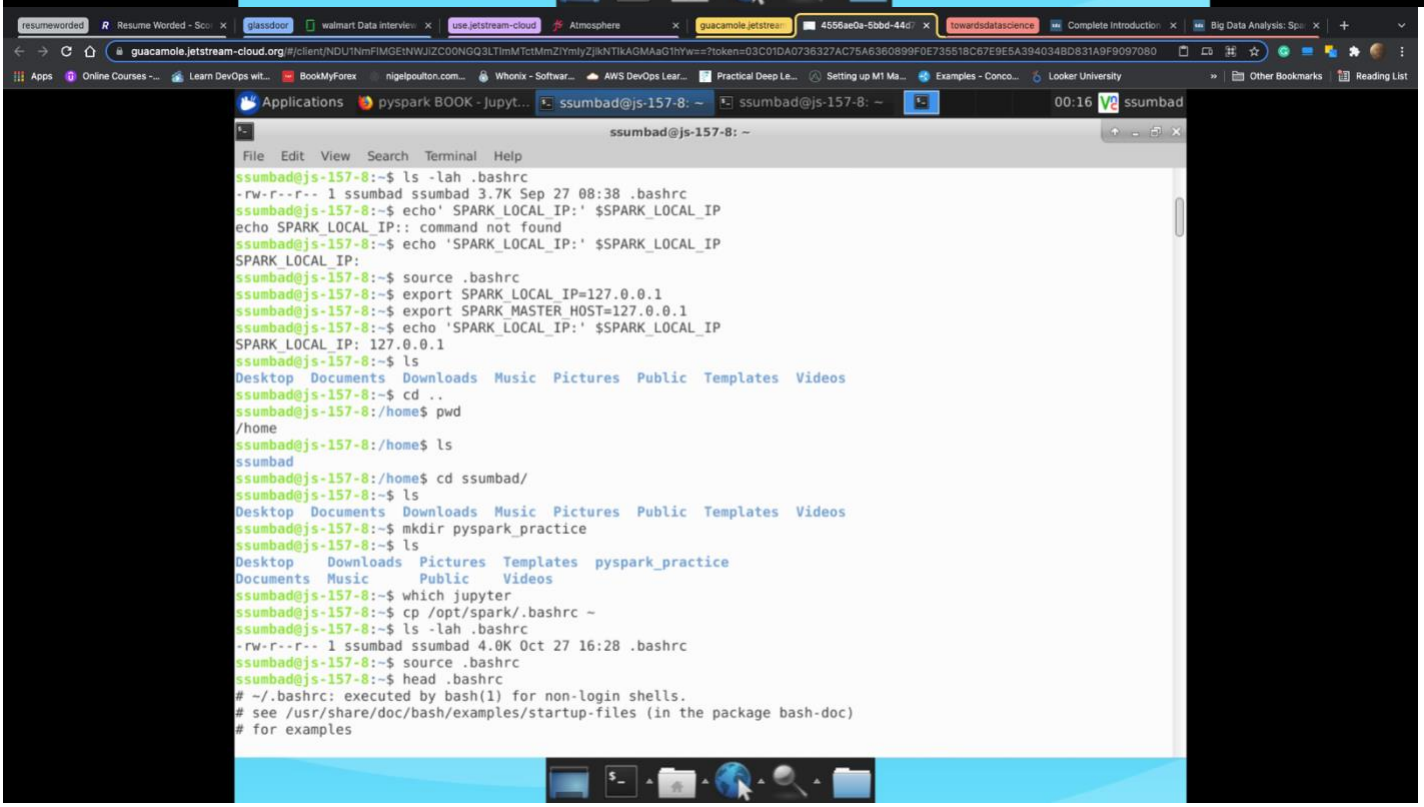
The trends that can be observed in the graph is that NN common nouns have a huge usage of the in this book compared to any other parts of speech, even the second highest NNS plural nouns have a usage lower than the half of the common nouns the corpus.

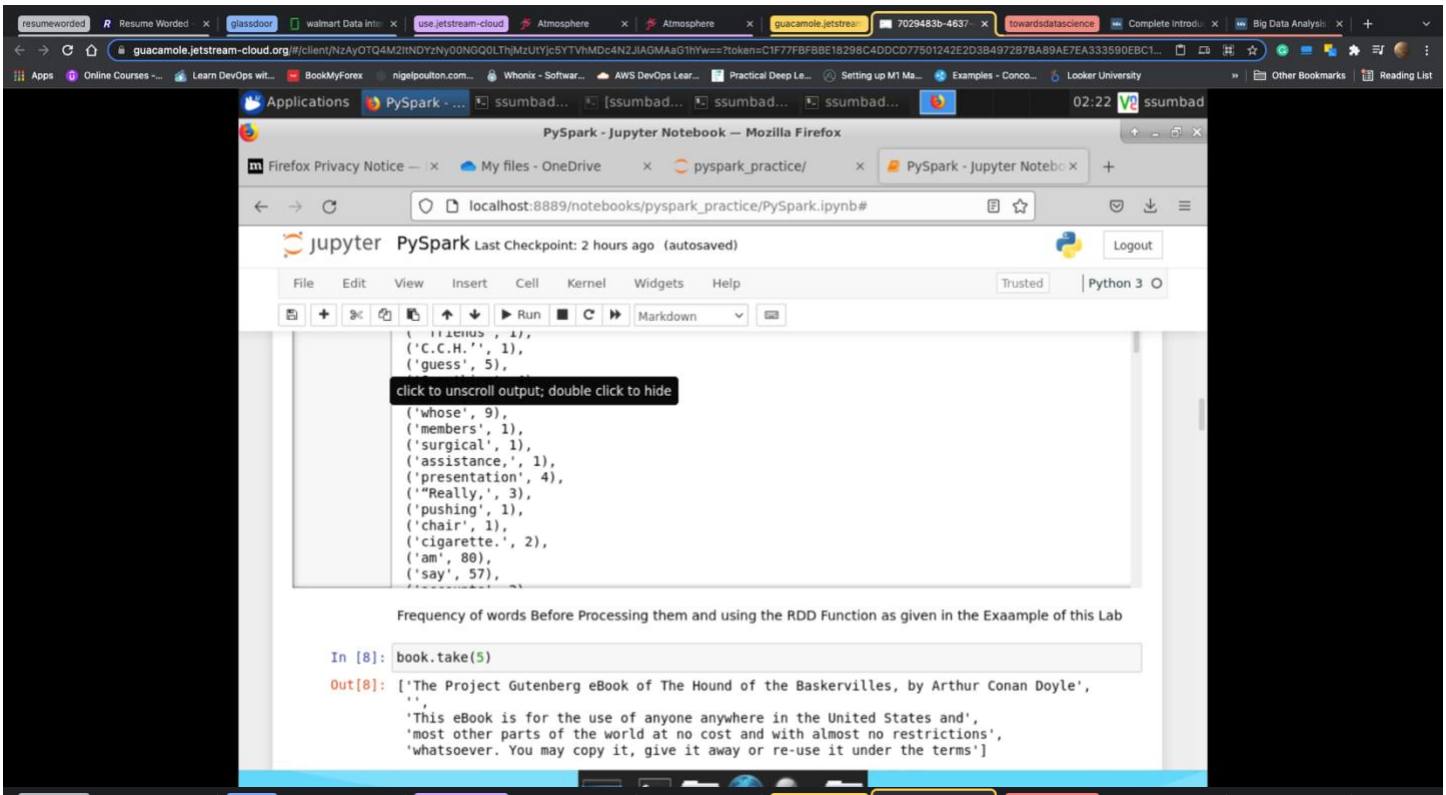
Conclusion:

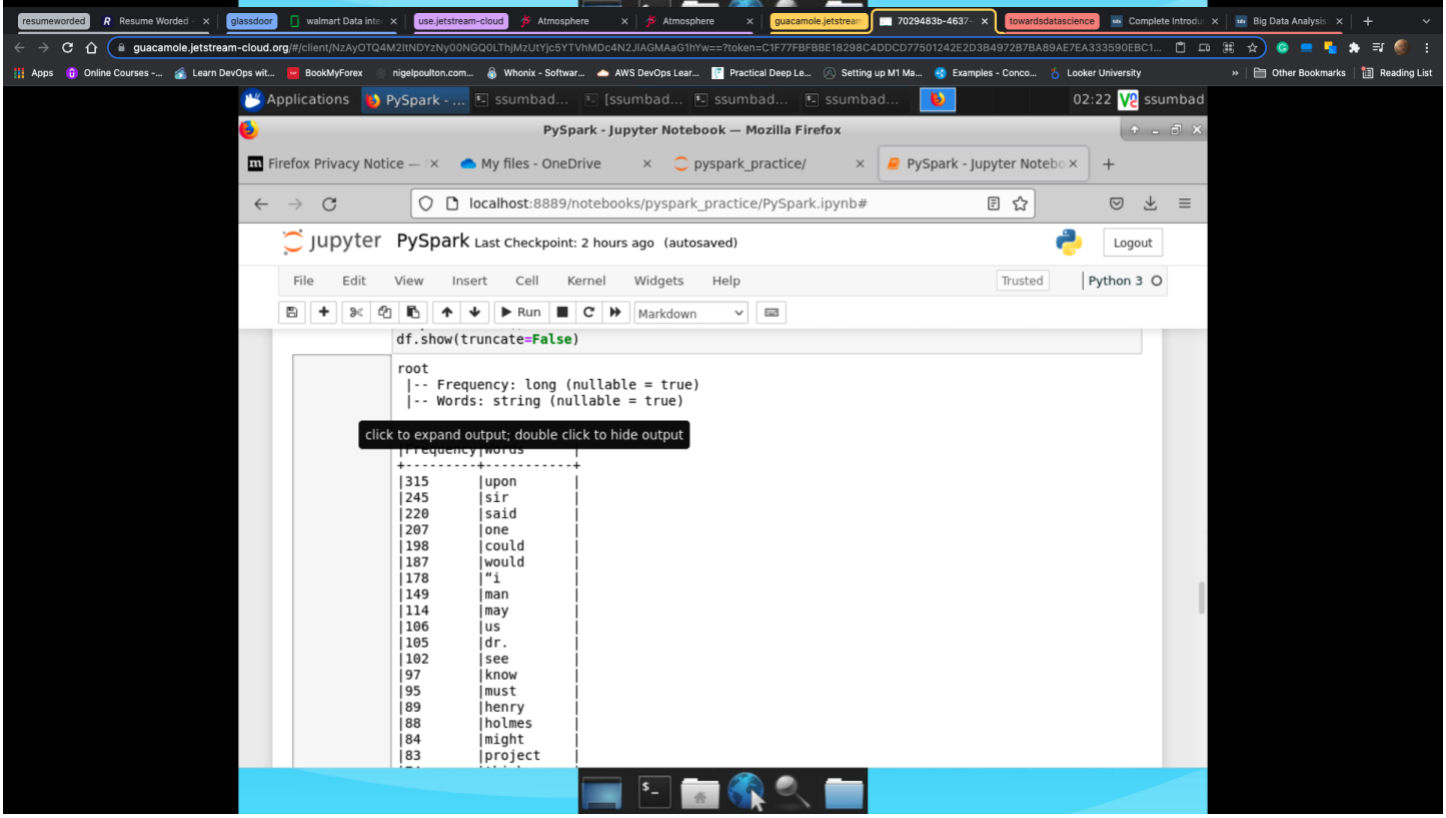
There was a total of 9284 words used by the author excluding the 163 stop words. “upon” a preposition being the most used word in the corpus and NN common nouns words are most used parts of speech used the corpus with a very high majority.

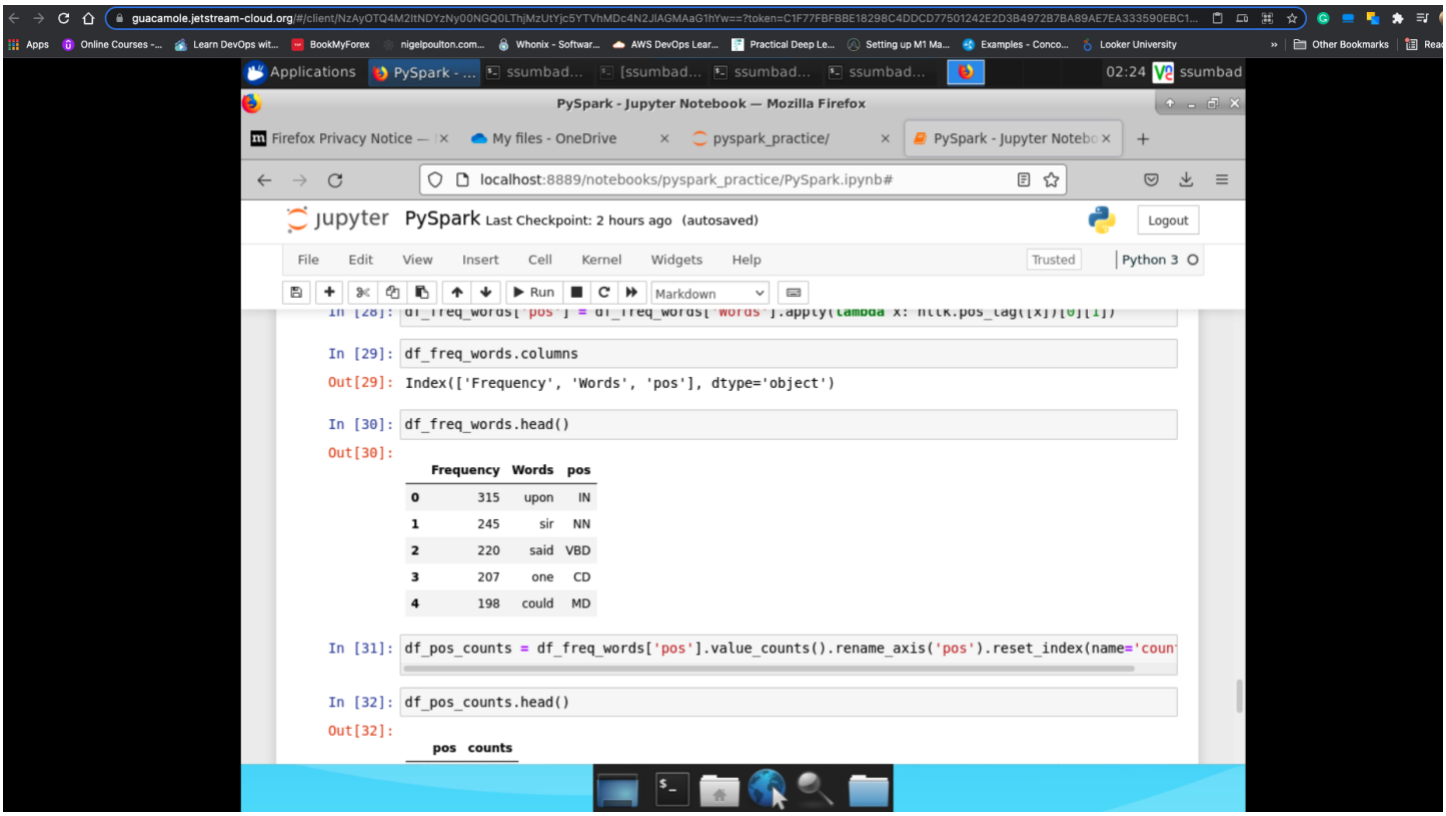
Screenshots of working on the instance :






PySpark - Jupyter Notebook — Mozilla Firefox
localhost:8889/notebooks/pyspark_practice/PySpark.ipynb#
jupyter PySpark Last Checkpoint: 2 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown
('C.C.H.'', 1),
('guess', 5),
click to unscroll output; double click to hide
('whose', 9),
('members', 1),
('surgical', 1),
('assistance,', 1),
('presentation', 4),
('"Really,', 3),
('pushing', 1),
('chair', 1),
('cigarette.', 2),
('am', 80),
('say', 57),
Frequency of words Before Processing them and using the RDD Function as given in the Exaample of this Lab
In [8]: book.take(5)
Out[8]: ['The Project Gutenberg eBook of The Hound of the Baskervilles, by Arthur Conan Doyle',
''
'This eBook is for the use of anyone anywhere in the United States and',
'most other parts of the world at no cost and with almost no restrictions',
'whatsoever. You may copy it, give it away or re-use it under the terms']


PySpark - Jupyter Notebook — Mozilla Firefox
localhost:8889/notebooks/pyspark_practice/PySpark.ipynb#
jupyter PySpark Last Checkpoint: 2 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown
df.show(truncate=False)
root
|-- Frequency: long (nullable = true)
|-- Words: string (nullable = true)
click to expand output; double click to hide output
+---------+-----------+
315	upon
245	sir
220	said
207	one
198	could
187	would
178	"i
149	man
114	may
106	us
105	dr.
102	see
97	know
95	must
89	henry
88	holmes
84	might
83	project



Sources:

- [1]https://www.analyticsvidhya.com/blog/2016/10/using-pyspark-to-perform-transformations-and-actions-on-rdd/
- [2]https://sparkbyexamples.com/pyspark/convert-pyspark-rdd-to-dataframe/
- [3]https://sparkbyexamples.com/spark/spark-map-vs-flatmap-with-examples/
- [4]https://spark.apache.org/docs/latest/rdd-programming-guide.html#actions
- [5] https://towardsdatascience.com/big-data-analysis-spark-and-hadoop-a11ba591c057