Teammates:

Sri Nikhitha Boddapati – sb4dz@umsystem.edu - 16322565

GitHub link: https://github.com/UMKC-APL-BigDataAnalytics/icp-1-Srinikhitha98

Sukumar Bodapati – sb5zh@umsystem.edu - 16326105

GitHub link: https://github.com/UMKC-APL-BigDataAnalytics/icp-1-sukumarbodapati

Video link: https://youtu.be/7zRazItx2nE

ICP2 - Big data App & Analytics - Spark

Outcomes of ICP2:

What is spark and how do we initialize:

> **Introduction to Spark:** Apache Spark is an open-source which is used for big data workloads and utilizes in-memory caching, optimized queries. It is mainly used for Real-time data processing and trivial operations like filters and joins. The spark application runs on master-slave operation, that uses the user's main function and execute the operations parallelly. Main abstraction of the Spark is RDD (resilient distributed dataset).
> **Initializing of Spark**: In order to access the cluster, we have to create Spark context object and for enabling this object we have to build SparkConf().

RDD (resilient distributed dataset):

RDD can be created in two ways: Can read the input file into rdd using 'sc.textfile(path)'. Convert the existing data frames into the rdd using 'rdd.dataframename'.

- RDD Operations: Transformations and Actions

Transformations used as part of this ICP:

- **Flatmap ():** It is used to map the input data to zero or many output items.
- **Distinct ():** This will return a new dataset with distinct element.
- **sortByKey ():** This is used to return the dataset either in ascending or descending order.
- **GroupBy ():** This is used to group the elements in the dataset.

Actions used as part of this ICP:

- **collect ():** This returns all the elements in the dataset**.**
- **Count ():** This returns number of elements in the dataset**.**

Objective:

To write a spark program to group words on the first letter.

Algorithm:

In order to group the words, first we need to remove the punctuation in the passage given and split the words in the passage and group then using the first letter.

Tasks:

1. Installed all the required software's like Java and Spark in the colab.

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
!wget -q https://downloads.apache.org/spark/spark-3.0.3/spark-3.0.3-bin-hadoop3.2.tgz
!tar xf spark-3.0.3-bin-hadoop3.2.tgz
!pip install -q findspark
```

2. Set the path to run the PySpark in the colab environment.

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.0.3-bin-hadoop3.2"
```

3. Imported Spark Session and Spark context from the libraries and Configure them to access the clusters.

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local[*]").appName("Big_Data_Application_ICP_2").getOrCreate()
sc=SparkContext.getOrCreate(conf=spark)
```

4. Opened a local file on Google colab and uploaded the text file using file.upload().

```
from google.colab import files
files.upload()
```

```
Choose Files icp.txt
• icp.txt(text/plain) - 1502 bytes, last modified: 1/28/2022 - 100% done
Saving icp.txt to icp (1).txt
{'icp.txt': b'As the Labor Day holiday nears, many people are planning travel and get-togethers to see family a
```

5. Read the input file into rdd (input_data) using spark context (sc.textfile(path)).

```
input_data=sc.textFile('icp.txt')
```

6. In order to see the number of elements in the input RDD, we used count action.

```
input_data.count()
```

```
1
```

7. To recheck the input_data rdd, we have use take action.

```
input_data.take(1)
```

```
['As the Labor Day holiday nears, many people are pl
```

8. In order to use the regular expressions, we have imported re module from the library and defined a function such that it removes all the punctuation and numeric.

```
import re
def RemovePunc(sentence):
     return re.sub(re.compile(r'[^a-zA-Z\s]'),"",sentence).lower().strip()
```

9. Converted them into lower using lower().
10. After removing the punctuations, we used flatmap() transformation for splitting the records by space .

```
cleaned_rdd=input_data.flatMap(lambda sentence: RemovePunc(sentence).split())
```

11. Each records are separated and there is repetitions of words like (a, the ,there) .In order to remove them, we have used distinct() which removes all the repetitive elements.

```
dist_data_rdd=cleaned_rdd.distinct()

dist_data_rdd.count()
```

```
149
```

12. Rechecked data using count() and take() actions

```
dist_data_rdd.take(5)
```

```
['as', 'holiday', 'are', 'planning', 'family']
```

13. As per question, we need to group elements using the first letter so we have used Groupby for grouping them and also used sortBYkey for displaying in descending order.

```
final_result = dist_data_rdd.groupBy(lambda x: (x[0])).sortByKey(False)
```

14. To write the output in a file, we have create a new text file using open command in colab and used for loop for displaying as well writing the data into the output file created.

```
textfile = open("ICP1_Output_File.txt", "w")
for (k,v) in final_result.collect():
  print(k,list(v))
  textfile.write(str(k)+' '+str(list(v))+'\n')
```

```
y ['year']
w ['we', 'were', 'weekend', 'what', 'want', 'with', 'wen', 'washington', 'well', 'who']
v ['very', 'visiting', 'vaccines', 'vaccinated']
u ['united', 'university', 'unvaccinated', 'us']
t ['this', 'than', 'think', 'take', 'these', 'the', 'travel', 'to', 'time', 'they', 'their', 'together', 'that'
s ['safe', 'steps', 'start', 'school', 'spoke', 'said', 'see', 'same', 'since', 'states', 'should', 'safety', '
r ['rates', 'reduce', 'risk', 'riskto', 'reason', 'report']
q ['questions']
p ['planning', 'policy', 'public', 'published', 'prevention', 'people', 'parts', 'physician', 'professor', 'prc
o ['of', 'officials', 'occurring', 'our', 'one']
n ['new', 'now', 'nears', 'navigate']
m ['members', 'medical', 'management', 'more', 'many', 'milken', 'main']
l ['levels', 'last', 'lifelines', 'likely', 'los', 'labor']
j ['january', 'journey']
```

Output File :

Notebook ICP1_Output_File.txt

```
y ['year']
w ['we', 'were', 'weekend', 'what', 'want', 'with', 'wen', 'washington', 'well', 'who']
v ['very', 'visiting', 'vaccines', 'vaccinated']
u ['united', 'university', 'unvaccinated', 'us']
t ['this', 'than', 'think', 'take', 'these', 'the', 'travel', 'to', 'time', 'they', 'their', 'together', 'that', 'time
s ['safe', 'steps', 'start', 'school', 'spoke', 'said', 'see', 'same', 'since', 'states', 'should', 'safety', 'stay',
r ['rates', 'reduce', 'risk', 'riskto', 'reason', 'report']
q ['questions']
p ['planning', 'policy', 'public', 'published', 'prevention', 'people', 'parts', 'physician', 'professor', 'protect']
o ['of', 'officials', 'occurring', 'our', 'one']
n ['new', 'now', 'nears', 'navigate']
m ['members', 'medical', 'management', 'more', 'many', 'milken', 'main']
l ['levels', 'last', 'lifelines', 'likely', 'los', 'labor']
j ['january', 'journey']
i ['is', 'infections', 'in', 'institute', 'it', 'if']
h ['holiday', 'higher', 'have', 'house', 'help', 'hospitalized', 'hospitalizations', 'highest', 'how', 'health', 'heal
g ['george', 'gettogethers']
f ['family', 'friendsunfortunately', 'fight', 'friends', 'for']
```

Challenges:

- As this is new programming language and new environment to work on, initially it was difficult to understand and get acquaintance with the Ide.
- We have knowledge on Dataframes but working on RDD is quite different like what operations can be performed.