ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY

SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING – SITE

Department of MSC in Artificial Intelligence

Course Assignment

Course: Distributed Computing for AI

Name: Mintesnot Fikir

Section: Regular

IDs: GSR1669/15

Submitted to: Beakal Gizachew (PhD)

Submission date February 2024

Assignment 5 spark

Part 1 - Implement and Analyze Word Count

Part 1 A

For the first part of this homework, you must write a basic word count program that reads in a single text file, counts the number of occurrences of each word in that text file, and outputs the result back to a text file as a list of key-value pairs, where the key is the word, and the value is the number of times the word occurred in the text file. In addition to basic word counting your code must also do the following:

1. It must be case insensitive (see lower() in Python)
2. It must ignore all punctuation (see, for example, translate() in Python)
3. It must ignore stop words (see filter() in Spark)
4. The output must be sorted by count in descending order (see sortBy() in Spark) To accomplish this you will use a combination of basic Python, and RDD operations in PySpark

Part 1 B

Extending Your Application Once you have your basic word count working for a single text file as described above, you will now extend it to work for a list of text files, combining the results from each file into a single RDD of key-value pairs, where the key is the word and the

value is the number of times that word appeared across all passed in text files. The result should only be sorted and written to file once after all text files have been processed.

Note: This is different from the in class example where we combined results using join(). See instead union() in Spark.

Analysis

Answer the following questions based on your WordCount application.

1. In the PySpark REPL, run your basic word count program on a single text file. a. What are the 25 most common words? Include a screenshot of program output to back-up your claim. b. How many stages is execution broken up into? Explain why. Include a screenshot of the DAG visualization from Spark's WebUI to back-up your claim.
2. In the PySpark REPL, run your extended word count program on all 10 text files. a. What are the 25 most common words? Include a screenshot of program output to back-up your claim. b. How many stages is execution broken up into? Explain why. Include a screenshot of the DAG visualization from Spark's WebUI to back-up your claim.
3. Your WordCount application should compute the same results as your WordCount application from Homework #1. Answer the following based on your knowledge of both MapReduce and Spark: a. If you were running your WordCount programs in a large cluster or cloud environment, and one of the nodes you were running on died mid computation, how would your MapReduce and Spark programs handle this? b. Explain one concrete benefit you experienced when writing the Spark version of WordCount compared to the MapReduce version.

> .

>

>

>

>

.

Answers

In [4]:
```
!pip install pyspark
!pip install pandas
!pip install nltk
!pip install pyngrok
```

```
Collecting pyspark
  Downloading pyspark-3.5.0.tar.gz (316.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 316.9/316.9 MB 1.8 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-pa
ckages (from pyspark) (0.10.9.7)
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.0-py2.py3-none-any.whl size=3174
25345 sha256=a24570b353c6568b09e4ee1c129220bbd5622ab9e6ed92afe2159c078cc597bb
  Stored in directory: /root/.cache/pip/wheels/41/4e/10/c2cf2467f71c678cfc8a6b9ac9
241e5e44a01940da8fbb17fc
Successfully built pyspark
Installing collected packages: pyspark
Successfully installed pyspark-3.5.0
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.1
0/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-pack
ages (from pandas) (2023.4)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-pac
kages (from pandas) (1.25.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages
(from python-dateutil>=2.8.1->pandas) (1.16.0)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.
8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (f
rom nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages
(from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-p
ackages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (fr
om nltk) (4.66.1)
Collecting pyngrok
  Downloading pyngrok-7.1.2-py3-none-any.whl (22 kB)
Requirement already satisfied: PyYAML>=5.1 in /usr/local/lib/python3.10/dist-packa
ges (from pyngrok) (6.0.1)
Installing collected packages: pyngrok
Successfully installed pyngrok-7.1.2
```

In [5]:
```python
from pyspark.sql import SparkSession
from pyngrok import ngrok
import pandas as pd
```

Initialize SparkContext and set **Spark UI** access point

In [6]:
```python
spark = SparkSession.builder\
        .master("local[*]")\
        .appName("Spark UI")\
        .config("spark.ui.port", "4050")\
        .getOrCreate()

NGROK_TOKEN = "2cMzS23dCc40jCub1EmWsC7lLYF_6SvSu354ptk7Ri79vwJH"
ngrok.set_auth_token(NGROK_TOKEN)

ngrok_tunnel = ngrok.connect(4050, bind_tls=True)
print("Spark UI:", ngrok_tunnel.public_url)
```

```
Spark UI: https://0931-35-196-171-220.ngrok-free.app
```

In [1]:
```
# ngrok.disconnect(ngrok_tunnel.public_url)
# spark.stop()
```

In [7]:
```
sc = spark.sparkContext
```

In [8]:
```
sc
```

Out[8]: **SparkContext**

Spark UI

Version	`v3.5.0`
Master	`local[*]`
AppName	`Spark UI`

Answer

Part 1 A: Basic Word Count for a single text file

Read single book

In [9]:
```
readsinglebook = sc.textFile("/content/drive/MyDrive/MintesnotF-A5Spark-assignment/
```

Load and broadcast stopwords

In [10]:
```
import re
stopwords_path = "stopwords.txt"
with open(stopwords_path, 'r') as file:
    stopwords = set(file.read().split())
bcast = sc.broadcast(stopwords)
```

Remove panctuation

In [11]:
```
def rmsplitpunon(line):
    line = re.sub(r'[^a-zA-Z\s]', '', line)
    return line.split()
```

Perform Word Count

- Word count with filtering stopwords, and also removing punctuation, and numbers

Based on the assignment instractun in addition to basic word counting my code also do the following:

1. It must be case insensitive (see lower() in Python)
2. It must ignore all punctuation (see, for example, translate() in Python)
3. It must ignore stop words (see filter() in Spark)
4. The output must be sorted by count in descending order (see sortBy() in Spark)

In [12]:
```
wordcountsingle = readsinglebook.flatMap(lambda line: rmsplitpunon(line)) \
                                .filter(lambda word: word.lower() not in bcast.va
                                .map(lambda word: (word.lower(), 1)) \
                                .reduceByKey(lambda x, y: x + y) \
                                .sortBy(lambda x: x[1], ascending=False)
```

Saving in Default RDD Format and also as a text

In [13]:
```
savedir = "/outputs/wordcountsinglebook"

wordcountsingle.saveAsTextFile(savedir)

temp_output_dir = "/wordcountsinglebook"
wordcountsingle.coalesce(1).saveAsTextFile(temp_output_dir)
!mv $(find /content/wordcountsinglebook -type f -name 'part-00000') '/outputs/wordc
!rm -rf /content/wordcountsinglebook
```

Display Top Words

In [14]:
```
top25 = wordcountsingle.take(25)
dfcountsingle = pd.DataFrame(top25, columns=['Word', 'Count Without Stopwords'])
dfcountsingle.index = range(1, len(dfcountsingle) + 1)
dfcountsingle
```

Out[14]:

	Word	Count Without Stopwords
1	jo	841
2	meg	585
3	amy	385
4	dont	375
5	laurie	349
6	beth	315
7	mother	253
8	march	250
9	good	241
10	girls	217
11	time	195
12	asked	189
13	young	173
14	face	170
15	ill	165
16	dear	162
17	day	156
18	illustration	143
19	looked	137
20	eyes	131
21	cried	128
22	great	124
23	head	121
24	didnt	119
25	wont	117

Lets compere without stopword and with stop word

In [15]:
```
withsword = readsinglebook.flatMap(lambda line: rmsplitpunon(line)) \
                                    .map(lambda word: (word.lower(), 1)) \
                                    .reduceByKey(lambda x, y: x + y) \
                                    .sortBy(lambda x: x[1], ascending=False)


withoutsword = readsinglebook.flatMap(lambda line: rmsplitpunon(line)) \
                                   .filter(lambda word: word.lower() not in bcast.va
                                   .map(lambda word: (word.lower(), 1)) \
                                   .reduceByKey(lambda x, y: x + y) \
                                   .sortBy(lambda x: x[1], ascending=False)
```

Comparison of with stop word and without stop word

In [16]:
```
top25without = withoutsword.take(25)
top25with = withsword.take(25)


dfwithout = pd.DataFrame(top25without, columns=['Word', 'Count Without Stopwords'])
dfwith = pd.DataFrame(top25with, columns=['Word', 'Count With Stopwords'])

dfwithout.index = range(1, len(dfwithout) + 1)
dfwith.index = range(1, len(dfwith) + 1)

comparisondf = pd.concat([dfwithout, dfwith], axis=1)
comparisondf
```

Out[16]:

	Word	Count Without Stopwords	Word	Count With Stopwords
1	jo	841	and	5095
2	meg	585	the	4808
3	amy	385	to	3125
4	dont	375	a	2833
5	laurie	349	her	2235
6	beth	315	of	2117
7	mother	253	i	1886
8	march	250	in	1505
9	good	241	she	1446
10	girls	217	it	1444
11	time	195	you	1354
12	asked	189	for	1285
13	young	173	was	1268
14	face	170	as	1155
15	ill	165	with	1107
16	dear	162	that	1003
17	day	156	jo	841
18	illustration	143	but	819
19	looked	137	he	791
20	eyes	131	so	688
21	cried	128	at	670
22	great	124	had	668
23	head	121	be	630
24	didnt	119	on	598
25	wont	117	said	589

Save the Word Count Results

In [17]:
```
dir1 = "/outputs/compered"
outwithout1 = f"{dir1}/resultswithoutstopwords"
outwith1 = f"{dir1}/resultswithstopwords"
withoutsword.saveAsTextFile(outwithout1)
withsword.saveAsTextFile(outwith1)
```

Read Back the Results

In [18]:
```
with open('/outputs/compered/resultswithoutstopwords/part-00000', 'r') as file:
    lines = file.readlines()
top_25 = lines[:25]
for line in top_25:
    word, count = line.strip("()\n'").split(', ')
    word = word.strip("'")
    print(f"Word - {word}           -        Count - {count}")
```

```
Word - jo           -        Count - 841
Word - meg           -        Count - 585
Word - amy           -        Count - 385
Word - dont           -        Count - 375
Word - laurie           -        Count - 349
Word - beth           -        Count - 315
Word - mother           -        Count - 253
Word - march           -        Count - 250
Word - good           -        Count - 241
Word - girls           -        Count - 217
Word - time           -        Count - 195
Word - asked           -        Count - 189
Word - young           -        Count - 173
Word - face           -        Count - 170
Word - ill           -        Count - 165
Word - dear           -        Count - 162
Word - day           -        Count - 156
Word - illustration           -        Count - 143
Word - looked           -        Count - 137
Word - eyes           -        Count - 131
Word - cried           -        Count - 128
Word - great           -        Count - 124
Word - head           -        Count - 121
Word - didnt           -        Count - 119
Word - wont           -        Count - 117
```

In [19]:
```
import pandas as pd
dir2 = "/outputs/compered"

withoutdir = f"{dir2}/resultswithoutstopwords"
withdir = f"{dir2}/resultswithstopwords"

import pandas as pd
import ast

def loadandtopd(dir):

    results = []

    with open(f"{dir}/part-00000", 'r') as file:
        for line in file:

            try:
                word, count = ast.literal_eval(line)
                results.append((word, count))
            except ValueError:
```

```
                print(f"Skipping line: {line}")

    df = pd.DataFrame(results, columns=['Word', 'Count'])

    return df

dfwithoutstopwords = loadandtopd(withoutdir)
dfwithstopwords = loadandtopd(withdir)

dfwithoutstopwords = dfwithoutstopwords.head(25)
dfwithstopwordstop = dfwithstopwords.head(25)

finaldf = pd.concat([dfwithstopwordstop.reset_index(drop=True),
                             dfwithoutstopwords.reset_index(drop=True)],
                            axis=1)

finaldf.columns = ['Word With Stopwords', 'occurance ', 'Word Without Stopwords', '
finaldf
```

Out[19]:

	Word With Stopwords	occurance	Word Without Stopwords	occurance
0	and	5095	jo	841
1	the	4808	meg	585
2	to	3125	amy	385
3	a	2833	dont	375
4	her	2235	laurie	349
5	of	2117	beth	315
6	i	1886	mother	253
7	in	1505	march	250
8	she	1446	good	241
9	it	1444	girls	217
10	you	1354	time	195
11	for	1285	asked	189
12	was	1268	young	173
13	as	1155	face	170
14	with	1107	ill	165
15	that	1003	dear	162
16	jo	841	day	156
17	but	819	illustration	143
18	he	791	looked	137
19	so	688	eyes	131
20	at	670	cried	128
21	had	668	great	124
22	be	630	head	121
23	on	598	didnt	119
24	said	589	wont	117

Part 1 B: Basic Word Count from multiple text file(books)

From 10 Books

the above is for a single book now lets do it for 10 books

Load/Read books

In [20]:
```
import os
bookspath = "/content/drive/MyDrive/MintesnotF-A5Spark-assignment/books"
```

```
book_files = [f for f in os.listdir(bookspath) if f.endswith('.txt')]
book_paths = [os.path.join(bookspath, f) for f in book_files]
```

Load and broadcast stopwords

In [21]:
```
# Load and broadcast stopwords
stopwordspath = "/stopwords.txt"
with open(stopwordspath, 'r') as file:
    stopwords = set(file.read().split())
bcaststopwords = sc.broadcast(stopwords)
```

Remove panctuation

and split

In [22]:
```
def rmandsplitpanc(line):
    line = re.sub(r'[^a-zA-Z\s]', '', line)
    return line.split()
```

Perform Word Count

- Word count with filtering stopwords, and also removing punctuation, and numbers

In [23]:
```
read10books = sc.emptyRDD()
for book_path in book_paths:
    book_rdd = sc.textFile(book_path)
    read10books = read10books.union(book_rdd) if not read10books.isEmpty() else boo

word_counts_with_10books = read10books.flatMap(lambda line: rmandsplitpanc(line)) \
                                      .filter(lambda word: word.lower() not in bcas
                                      .map(lambda word: (word.lower(), 1)) \
                                      .reduceByKey(lambda x, y: x + y) \
                                      .sortBy(lambda x: x[1], ascending=False)
```

Saving in Default RDD Format and also as a text

In [24]:
```
# as RDD format
savedir10 = "/outputs/word_counts_with_10books"

word_counts_with_10books.saveAsTextFile(savedir10)


# as a text file
outdir = "/content/word_counts_with_10books"
word_counts_with_10books.coalesce(1).saveAsTextFile(outdir)
!mv $(find /content/word_counts_with_10books -type f -name 'part-00000') '/outputs/
!rm -rf /content/word_counts_with_10books
```

Display Top Words

In [25]:
```
top20word=word_counts_with_10books.take(25)
top20wordDF = pd.DataFrame(top20word, columns=['Word', '   Count Without Stopwords

top20wordDF.index = range(1, len(top20wordDF) + 1)
top20wordDF
```

Out[25]:

	Word	Count Without Stopwords from all 10 books
1	government	2723
2	united	2559
3	time	2488
4	great	2161
5	volume	1995
6	people	1610
7	general	1596
8	men	1579
9	man	1576
10	good	1395
11	work	1274
12	life	1175
13	thought	1162
14	years	1148
15	south	1132
16	war	1129
17	long	1099
18	british	1078
19	order	1054
20	day	1036
21	american	980
22	number	977
23	hand	955
24	president	950
25	country	942

Lets compere without stopword and with stop word

In [26]:
```
word_counts_with_stopwords = read10books.flatMap(lambda line: rmandsplitpanc(line))
                                        .map(lambda word: (word.lower(), 1)) \
                                        .reduceByKey(lambda x, y: x + y) \
                                        .sortBy(lambda x: x[1], ascending=False)

word_counts_without_stopwords = read10books.flatMap(lambda line: rmandsplitpanc(lir
                                    .filter(lambda word: word.lower() not in bcaststc
                                    .map(lambda word: (word.lower(), 1)) \
                                    .reduceByKey(lambda x, y: x + y) \
                                    .sortBy(lambda x: x[1], ascending=False)
```

Display comparison of with stop word and without stop word

In [27]:
```
top_25_without = word_counts_without_stopwords.take(25)
top_25_with = word_counts_with_stopwords.take(25)

df_without_stopwords = pd.DataFrame(top_25_without, columns=['Word', 'Count Without
df_with_stopwords = pd.DataFrame(top_25_with, columns=['Word', 'Count With Stopword

df_without_stopwords.index = range(1, len(df_without_stopwords) + 1)
df_with_stopwords.index = range(1, len(df_with_stopwords) + 1)

comparison_df = pd.concat([df_without_stopwords, df_with_stopwords], axis=1)

comparison_df
```

Out[27]:

	Word	Count Without Stopwords	Word	Count With Stopwords
1	government	2723	the	128046
2	united	2559	of	79598
3	time	2488	and	52442
4	great	2161	to	49129
5	volume	1995	in	36484
6	people	1610	a	35970
7	general	1596	that	20586
8	men	1579	is	16232
9	man	1576	it	16069
10	good	1395	as	15305
11	work	1274	was	14943
12	life	1175	for	13702
13	thought	1162	with	13671
14	years	1148	be	13627
15	south	1132	i	13118
16	war	1129	by	12471
17	long	1099	which	11385
18	british	1078	on	10351
19	order	1054	he	9929
20	day	1036	not	9851
21	american	980	this	9561
22	number	977	at	9555
23	hand	955	his	9111
24	president	950	her	9091
25	country	942	had	8508

Save the Word Count Results

In [28]:
```
# Directories to save the results
output_dir_without_stopwords = "/outputs/without_stopwords_from10books"
output_dir_with_stopwords = "/outputs/with_stopwords_from10books"

# Save the results
word_counts_without_stopwords.saveAsTextFile(output_dir_without_stopwords)
word_counts_with_stopwords.saveAsTextFile(output_dir_with_stopwords)
```

Read Back the Results

In [30]:
```
# To display top 25 words directly
top_25 = word_counts_without_stopwords.take(25)
```

```python
for word, count in top_25:
    print(f"Word - {word}           -        Count - {count}")
```

```
Word - government           -        Count - 2723
Word - united           -        Count - 2559
Word - time           -        Count - 2488
Word - great           -        Count - 2161
Word - volume           -        Count - 1995
Word - people           -        Count - 1610
Word - general           -        Count - 1596
Word - men           -        Count - 1579
Word - man           -        Count - 1576
Word - good           -        Count - 1395
Word - work           -        Count - 1274
Word - life           -        Count - 1175
Word - thought           -        Count - 1162
Word - years           -        Count - 1148
Word - south           -        Count - 1132
Word - war           -        Count - 1129
Word - long           -        Count - 1099
Word - british           -        Count - 1078
Word - order           -        Count - 1054
Word - day           -        Count - 1036
Word - american           -        Count - 980
Word - number           -        Count - 977
Word - hand           -        Count - 955
Word - president           -        Count - 950
Word - country           -        Count - 942
```