CISC5950 – Big Data Programming – Spring 2022

Professor: Ying Mao
Lab 2
Due date: 4/27/2022
Student: Yuliya Akchurina

Lab 2. Spark. Questions 2 and 3

In this lab, please, based on your previous code, implement the K-Means algorithm, you can use any spark related library package.

2. Please redo Project 1 Bonus Question 1(Part 2).

3. Please redo Project 1 Part 1 Question 1 with different levels of parallelism, 2, 3, 4, 5 with (Part 3, just change one line in the test.sh file, add –conf spark.default.parallelism=2 after spark-submit to set parallelism level to 2).

Question 2.

The goal of this task is to use Spark on Hadoop HDFS in order to calculate the probability of getting a ticket give a black car that is parked in one of the three specified street codes: 34510, 10030, 34050.

The data is provided in a csv file and the file is located at /mapreduce-test/mapreduce-test-python/project1/Parking_Violations_Issued_-_Fiscal_Year_2021.csv

The data is read from the csv file. Only relevant eight columns are selected to create a dataframe:

Street Code1, Street Code2, Street Code3, Vehicle Color, Violation Description, No Standing or Stopping Violation, Hydrant Violation, Double Parking Violation.

The data is preprocessed the following way:

- The rows that have missing values in column Vehicle Color are dropped.
- All the string values in the dataframe are converted to lowercase.
- The Vehicle Color column data is filtered out to include only the black color vehicles. The assumption made from the dataset is that black color can be written the following way black, blk, bk, b.
- All the columns are renamed for convenience, from Street Code1 to street_code1, Vehicle Color to vehicle_color, and so on.

To count all black cars that received tickets in the 3 street codes, the dataframe is filtered to only include those values in the respective columns.

Second value is a count of all black cars that received tickets for parking or standing that was specified on the ticket in one of the columns: Violation Description, No Standing or Stopping Violation, Hydrant Violation, Double Parking Violation. The dataframe is filtered to include substrings “park” or “stand” on those columns. As well as vehicle color black and three street codes.

The probability is calculated by dividing the count of all black cars that received tickets for parking or standing that was specified on the ticket by the count of all black cars that received tickets in the 3 street codes.

The resulting output:

The probability is 39.54%

```
2022-04-29 16:00:31,591 INFO scheduler.DAGScheduler: Job 11
pl.java:0, took 0.123142 s

Probability of getting a parking ticket for a black car
             in street codes 34510, 10030, 34050 is 39.54%
```

```
Size of DF with black cars in 3 street codes 34510, 10030, 34050
(9259, 8)
Size of DF for parking tickets of black cars in 3 street codes 34510, 10030, 34050
(3661, 8)
Probability of getting a ticket is 39.53990711739929%
```

Question3

Use Spark on Hadoop HDFS to find out the date when the most tickets are being issued and rerun eth code using different levels of parallelism, 2, 3, 4, 5.

The data is provided in a csv file and the file is located at /mapreduce-test/mapreduce-test-python/project1/Parking_Violations_Issued_-_Fiscal_Year_2021.csv

The column Issue Date is used for this problem. The rows with missing values are dropped. The dataframe is grouped by the Issue Date column and the values are counted. The resulting count column is ordered in descending order and top 5 rows are printed as a result to show the days that received top five highest ticket counts. Also, the first lien is printed as a row to see the date that received the most tickets.

Top 5 dates with the most tickets being issued are 11/27/2020, 08/20/2020, 09/04/2020, 10/15/2020, 09/17/2020.

```
Count of Tickets_per_date
The Top 5 dates with the most tickets issused
+----------+----------------+
|Issue Date|tickets_per_date|
+----------+----------------+
|11/27/2020|           71101|
|08/20/2020|           69759|
|09/04/2020|           69296|
|10/15/2020|           68822|
|09/17/2020|           68666|
+----------+----------------+
only showing top 5 rows
```

The date with the most tickets issued is 11/27/2020 with 71101 tickets issued

```
The date with the most tickets issued: Row(Issue Date='11/27/2020', tickets_per_date=71101)
```

The use of parallelism, 2, 3, 4, 5 yields the following results:

conf spark.default.parallelism=2

```
2022-04-29 14:57:19,849 INFO codegen.CodeGenerator: Code generated in 21.916937 ms
2022-04-29 14:57:19,925 INFO codegen.CodeGenerator: Code generated in 54.723696 ms
+----------+---------------+
|Issue Date|tickets_per_date|
+----------+---------------+
|11/27/2020|          71101|
|08/20/2020|          69759|
|09/04/2020|          69296|
|10/15/2020|          68822|
|09/17/2020|          68666|
+----------+---------------+
only showing top 5 rows
```

```
2022-04-29 14:57:40,213 INFO storage.BlockManagerInfo: Removed broadcast_4_piece0 on instance-1.c.project5950.
internal:33001 in memory (size: 18.1 KiB, free: 434.3 MiB)
The date with the most tickets issued: Row(Issue Date='11/27/2020', tickets_per_date=71101)
```

conf spark.default.parallelism=3

```
2022-04-29 14:27:15,636 INFO codegen.CodeGenerator: Code generated in 61.229479 ms
2022-04-29 14:27:15,697 INFO codegen.CodeGenerator: Code generated in 35.072709 ms
+----------+---------------+
|Issue Date|tickets_per_date|
+----------+---------------+
|11/27/2020|          71101|
|08/20/2020|          69759|
|09/04/2020|          69296|
|10/15/2020|          68822|
|09/17/2020|          68666|
+----------+---------------+
only showing top 5 rows
```

```
2022-04-29 14:27:34,936 INFO scheduler.DAGScheduler: Job 4 finished: first at /spark-examples/Lab2/l2q3/lab2_q
3.py:46, took 0.157025 s
The date with the most tickets issued: Row(Issue Date='11/27/2020', tickets_per_date=71101)
```

conf spark.default.parallelism=4

```
2022-04-29 14:34:29,177 INFO codegen.CodeGenerator: Code generated in 30.411884 ms
2022-04-29 14:34:29,236 INFO codegen.CodeGenerator: Code generated in 32.659847 ms
+----------+---------------+
|Issue Date|tickets_per_date|
+----------+---------------+
|11/27/2020|          71101|
|08/20/2020|          69759|
|09/04/2020|          69296|
|10/15/2020|          68822|
|09/17/2020|          68666|
+----------+---------------+
only showing top 5 rows
```

```
2022-04-29 14:34:47,239 INFO scheduler.DAGScheduler: Job 4 finished: first at /spark-examples/Lab2/l2q3/lab2_q
3.py:46, took 0.152861 s
The date with the most tickets issued: Row(Issue Date='11/27/2020', tickets_per_date=71101)
```

conf spark.default.parallelism=5

```
2022-04-29 14:51:38,341 INFO codegen.CodeGenerator: Code generated in 57.868437 ms
2022-04-29 14:51:38,390 INFO codegen.CodeGenerator: Code generated in 33.787278 ms
+----------+---------------+
|Issue Date|tickets_per_date|
+----------+---------------+
|11/27/2020|          71101|
|08/20/2020|          69759|
|09/04/2020|          69296|
|10/15/2020|          68822|
|09/17/2020|          68666|
+----------+---------------+
only showing top 5 rows
```

```
2022-04-29 14:51:57,127 INFO scheduler.DAGScheduler: Job 4 finished: first at /spark-examples/Lab2/l2q3/lab2_q
3.py:46, took 0.554954 s
The date with the most tickets issued: Row(Issue Date='11/27/2020', tickets_per_date=71101)
```