INF553 Foundations and Applications of Data Mining

Spring 2020

Assignment 1

Deadline: Feb 10, 3:00 PM PST

1. Overview of the Assignment

In assignment 1, you will complete three tasks. The goal of these tasks is to let you be familiar with Spark operation types (e.g., transformations and actions) and perform data exploration tasks on the Yelp dataset (https://www.yelp.com/dataset).

If you have questions about the assignment, please ask on the Piazza, which will also help other students.

2. Requirements

2.1 Programming Requirements

a. You must use Python to implement all tasks. You can only use standard python libraries (i.e., external libraries like numpy or pandas are not allowed). There will be **10% bonus** for each task if you also submit a Scala implementation and both your Python and Scala implementations are correct.

b. You are required to only use Spark RDD in order to understand Spark operations more deeply. You will not get any point if you use Spark DataFrame or DataSet.

2.2 Programming Environment

Python 3.6, JDK 1.8, Scala 2.11, and Spark 2.4.4
We will use these library versions to compile and test your code. There will be a 20% penalty if we cannot run your code due to the library version inconsistency.

2.3 Write your own code

Do not share code with other students!!

For this assignment to be an effective learning experience, you must write your own code! We emphasize this point because you will be able to find Python implementations of some of the required functions on the web. Please do not look for or at any such code!

TAs will combine all the code we can find from the web (e.g., Github) as well as other students' code from this and other (previous) sections for plagiarism detection. We will report all detected plagiarism.

2.4 What you need to turn in

We will grade all submissions on Vocareum and the submissions on the blackboard will be ignored.

a. three Python scripts, named: (all lowercase)

task1.py, task2.py, task3.py

b1. [OPTIONAL] three Scala scripts, named: (all lowercase)

task1.scala, task2.scala, task3.scala

c. You don't need to include your results or the datasets. We will grade on your code with our testing data (data will be in the same format).

3. Yelp Data

In this assignment, you will explore the Yelp dataset. You need to download the original JSON files HERE (https://www.yelp.com/dataset). You are going to use the entire review.json and business.json for assignment 1.

You can find the metadata of the datasets here (https://www.yelp.com/dataset/documentation/main).

4. Tasks

4.1 Task1: Data Exploration (3 points)

You will explore the dataset, review.json, containing review information for this task, and you need to write a program to automatically answer the following questions:

A. The total number of reviews (0.5 point)

B. The number of reviews in 2018 (0.5 point)

C. The number of distinct users who wrote reviews (0.5 point)

D. The top 10 users who wrote the largest numbers of reviews and the number of reviews they wrote (0.5 point)

E. The number of distinct businesses that have been reviewed (0.5 point)

F. The top 10 businesses that had the largest numbers of reviews and the number of reviews they had (0.5 point)

Input format: (we will use the following command to execute your code)

Spark-submit task1.py <review_filepath> <output_filepath>

Output format:

IMPORTANT: Please strictly follow the output format since your code will be graded automatically.

a. The output for Questions A/B/C/E will be a number. The output for Questions D/F will be a list, which is sorted by the number of reviews in the descending order. If two user_ids/business_ids have the same number of reviews, please sort the user_ids /business_ids in the alphabetical order.

b. You need to write the results in the JSON format file. You must use **exactly the same tags** (see the red boxes in Figure 2) for answering each question.

```
{
    "n_review": 11111,
    "n_review_2018": 11111,
    "n_user": 11111,
    "top10_user": [["ABCDEFGHIJKLMNOPQ", 1111], ..., ["BCDEFGHIJKLMNOPQR", 111]],
    "n_business": 11111,
    "top10_business": [["QPONMLKJIHGFEDCBA", 1111], ..., ["RQPONMLKJIHGFEDCB", 111]]
}
```

Figure 1: JSON output structure for task1

4.2 Task2: Partition (2 points)

Since processing large volumes of data requires performance decisions, properly partitioning the data for processing is imperative.

In this task, you will show the number of partitions for the RDD used for **Task 1 Question F** and the number of items per partition.

Then you need to **use a customized partition function to improve the performance** of map and reduce tasks. A time duration (for executing Task 1 Question F) comparison between **the default partition** and **the customized partition** (RDD built using the partition function) should also be shown in your results.

Hint:

Certain operations within Spark trigger an event known as the shuffle. The shuffle is Spark's mechanism for re-distributing data so that it's grouped differently across partitions. This typically involves copying data across executors and machines, making the shuffle a complex and costly operation. So, trying to design a partition function to avoid the shuffle will improve the performance a lot.

Input format: (we will use the following command to execute your code)

Spark-submit task2.py <review_filepath> <output_filepath> <n_partition>

Output format:

A. The output for the number of partition and execution time will be a number. The output for the number of items per partition will be a list of numbers.

B. You need to write the results in a JSON file. You must use **exactly the same tags**.

```
{
    "default":{
        "n_partition": 2,
        "n_items":[1111,2222],
        "exe_time": 0
    },
    "customized":{
        "n_partition":2,
        "n_items":[3332,1],
        "exe_time":0
    }
}
```

Figure 3: JSON output structure for task2

4.3 Task3: Exploration on Multiple Datasets (2 points)

In task3, you are asked to explore two datasets together containing review information (review.json) and business information (business.json) and write a program to answer the following questions:

A. What is the average stars for each city? (**DO NOT** use the stars information in the business file) (1 point)

B. You are required to use two ways to **print** top 10 cities with highest stars.

   Method1: Collect all the data, sort in python, and then print the first 10 cities

   Method2: Sort in Spark, take the first 10 cities, and then print these 10 cities

Input format: (we will use the following command to execute your code)

Spark-submit task3.py <review_filepath> <business_filepath> <output_filepath_question_a> <output_filepath_question_b>

Output format:

a. You need to write the results for Question A as a file,The header (first line) of the file is "city,stars". The outputs should be sorted by the average stars in descending order. If two cities have the same stars, please sort the cities in the alphabetical order. (see Figure 3 left)

b. You also need to write the answer for Question B in a JSON file. You must use **exactly the same tags** for the task.

```
city,stars        {
abcd,5.0              "m1": 0,
bcde,5.0              "m2": 0
cdef,5.0          }
defg,5.0
abce,4.9
```

Figure 3: Question A output file structure (left) and JSON output structure (right) for task3

5. Grading Criteria

(% penalty = % penalty of possible points you get)

1. You can use your free 5-day extension separately or together.

2. There will be 10% bonus if you use both Scala and Python.

3. We will combine all the code we can find from the web (e.g., Github) as well as other students' code from this and other (previous) sections for plagiarism detection. If a plagiarism is detected, there will be no point for the entire assignment and we will report all detected plagiarism.

4. All submissions will be graded on the Vocareum. Please strictly follow the format provided, otherwise you can't get the point even though the answer is correct.

5. If the outputs of your program are unsorted or partially sorted, there will be 50% penalty.

6. if you use Spark DataFrame, DataSet, sparksql, there will be 20% penalty.

7. We can regrade on your assignments within seven days once the scores are released. No argue after one week. There will be 20% penalty if our grading is correct.

8. There will be 20% penalty for late submission within a week and no point after a week.

9. There will be no point if the total execution time exceeds 8 minutes.

10. Only when your results from Python are correct, the bonus of using Scala will be calculated. There is no partially point for Scala. See the example below:

Example situations

Task	Score for Python	Score for Scala (10% of previous column if correct)	Total
Task1	Correct: 3 points	Correct: 3 * 10%	3.3
Task1	Wrong: 0 point	Correct: 0 * 10%	0.0
Task1	Partially correct: 1.5 points	Correct: 1.5 * 10%	1.65
Task1	Partially correct: 1.5 points	Wrong: 0	1.5

6. Common problems causing fail submission on Vocareum/FAQ

(If your program runs seems successfully on your local machine but fail on Vocareum, please check these)

1. Try your program on Vocareum terminal. Remember to set python version as python3.6,

```
export PYSPARK_PYTHON=python3.6
```

And use the latest Spark

```
/home/local/spark/latest/bin/spark-submit
```

2. Check the input commend line format.

3. Check the output format, for example, the header, tag, typo.

4. Check the requirements of sorting the results.

5. Your program scripts should be named as task1.py task2.py etc.

6. Check whether your local environment fit the assignment description, i.e. version, configuration.

7. If you implement the core part in python instead of spark, or implement it in a high time complexity way (e.g. search an element in a list instead of a set), your program may be killed on the Vocareum because it runs too slow.

8. You are required to only use Spark RDD in order to understand Spark operations more deeply. You will not get any point if you use Spark DataFrame or DataSet. Don't import sparksql.